WAPT Documentation

Version 2.3

Tranquil-IT

avr. 17, 2023

Contents

Bienvenue sur la documentation officielle de WAPT par Tranquil IT dernière version en date du 2023-04-17.

Cliquer ici pour une version PDF de la documentation complète.

WAPT est un outil de déploiement de logiciels et de configurations qui peut être comparé à Microsoft SCCM (System Center Configuration Management) (maintenant appelé MECM (Microsoft Endpoint Configuration Management)), Ivanti UIM (Unified Endpoint Manager), IBM Bigfix, Tanium, OPSI, PDQDeploy, ou Matrix42. WAPT existe en deux versions, *WAPT Discovery et WAPT Enterprise*.

Pour les Administrateurs Système:

- Installer des logiciels de manière silencieuse.
- Maintenir à jour les logiciel installés et leurs configurations.
- Configurer les logiciels et le système pour diminuer la charge sur les équipes de support.
- Désinstaller de manière silencieuse les logiciels ou configurations obsolètes.
- Réduire le besoin de support par les équipes informatiques, dont les temps de réaction sont parfois long du fait de leurs charges de travail.
- Réduire autant que possible la consommation de bande passante sur les sites distants afin de la préserver pour des usages productifs.

Pour les RSSI

- Faire converger les logiciels installés vers la norme de sécurité acceptable pour l'entreprise.
- Préparez votre entreprise à l'arrivée du RGPD et aider votre DPD, qui deviendra un proche collègue, à tenir à jour son registre des traitements.
- Ne plus tolérer le fonctionnement des machines en mode *Administrateur*.
- Ne plus tolérer que les utilisateurs téléchargent et exécutent des logiciels à partir de leur répertoire personnel.
- Commencer à appliquer les SRPs (Software Restriction Policies), également connues sous le nom de *Applocker* ou WDAC (Windows Defender Application Control) pour améliorer la sécurité informatique au niveau des applications.
- Réduire le niveau d'exposition aux vulnérabilités des logiciels et aux attaques par mouvement latéral .
- Faire remonter des indicateurs d'audit pour une meilleure connaissance de l'état des équipements informatiques installés et de leur niveau de sécurité global.

- Déployer immédiatement pour réagir à une menace type Wannacry ou notPetya.

Pour les utilisateurs finaux

- Avoir installé des logiciels configurés pour bien fonctionner dans le contexte de votre organisation et avoir confiance qu'ils fonctionneront correctement.
- Rendre les *Utilisateurs* plus autonomes pour installer des logiciels de manière sûre et sécurisée.
- Disposer de systèmes professionnels plus performants et plus prévisibles grâce à des configurations logicielles standard.

CHAPTER 1

Présentation des grands principes de WAPT

1.1 A quoi sert WAPT ?

WAPT installe, met à jour et supprime les logiciels et les configurations sur les appareils Windows, Linux et macOS. Le déploiement de logiciels (Firefox, MS Office, etc.) peut être effectué à partir d'un serveur central à l'aide d'une console graphique. WAPT reprend de nombreuses idées de l'outil de gestion de paquets apt Debian Linux, d'où son nom.

Des entreprises privées de toutes tailles, des collèges, des écoles, des universités, des laboratoires de recherche, des gouvernements locaux et nationaux, des hôpitaux, des mairies et des ministères d'État du monde entier utilisent avec succès **WAPT**.

WAPT existe en deux versions, **Discovery** et **Enterprise**, toutes deux propriétaires, la version **Community** ayant été amicalement *forkée* à la communauté Opensource.

WAPT est très efficace pour répondre aux **besoins récurrents de mise à jour de Firefox ou Chrome** et c'est souvent pour couvrir ce besoin de base que WAPT est initialement adopté ; il devient alors un outil de choix pour les tâches quotidiennes de l'administrateur système.

1.2 Certification de sécurité de l'ANSSI

Suite à sa certification CSPN du 14 février 2018, WAPT a obtenu le 15 mars 2018 la Qualification Elémentaire de l'ANSSI.

Figure1: Visa de sécurité de l'ANSSI du 14 février 2018 pour WAPT Enterprise Edition 1.5.0.13

1.3 Genèse WAPT

1.3.1 Notre constat après 15 ans d'infogérance

L'administration d'un large parc de PC sous Microsoft Windows est aujourd'hui une tâche difficile dans un environnement sécurisé:

- Les méthodes généralement utilisées (mastérisations type *ghost* ou *clonezilla*) sont efficaces si les parcs machines et les parcs applicatifs sont homogènes et que les profils utilisateurs sont itinérants.
- Les outils de télé-déploiement (*OCSInventory* ou *WPKG*) diffusent les logiciels mais ne permettent pas d'effectuer de manière simple les personnalisations qui évitent les demandes de support utilisateur.
- Les logiciels de petits éditeurs nécessitent souvent des droits *Administrateur Local* pour fonctionner correctement.
- Les solutions actuellement disponibles pour résoudre ces problèmes sont soit trop coûteuses, soit trop inefficaces, et elles sont dans tous les cas trop complexes.

1.3.2 Hypothèses et motivations du développement WAPT

Le développement de WAPT est animé par deux principes:

- Ce qui est **compliqué** doit être rendu **simple**.
- Ce qui est **simple** doit être rendu **trivial**.

WAPT s'appuie sur un jeu d'hypothèses fondamentales :

- Les adminsys doivent connaître un langage de script, et WAPT a choisi Python pour la profondeur et l'étendue de ses librairies.
- Les administrateurs système qui ont peu d'expérience avec les langages de script doivent s'inspirer d'exemples simples et efficaces qu'ils sauront adapter à leurs besoins.
- Les adminsys doivent pouvoir communiquer sur l'efficacité de leurs actions à leur direction et reporter les écarts de processus aux auditeurs internes ou externes.
- Les adminsys doivent pouvoir collaborer avec leur équipe informatique; ainsi les dépôts WAPT internes fournissent des paquets auxquels ils peuvent faire confiance pour les déployer sur leur réseau. Sinon, ils peuvent choisir des dépôts externes publics qui leur fournissent les garanties de sécurité qu'ils jugent suffisantes.
- Les administrateurs système sont conscients que les postes de travail des utilisateurs servent à des fins commerciales et que certaines personnalisations doivent être possibles. L'adaptation de l'infrastructure aux besoins de l'entreprise est facilitée par la notion de groupes et des:abbr:*OU (Organizational Units)*; ils permettent de sélectionner un grand nombre de machines pour personnaliser leur configuration.

CHAPTER 2

Principes fondamentaux

2.1 Principe de Dépôt

Les paquets sont stockés dans un répertoire web. Ils ne sont pas stockés dans une base de données.

Note: Le protocole de transport utilisé pour le déploiement des paquets est le **HTTPS**.

Les paquets WAPT sont servis par le serveur web **`Nginx`**, disponible sous Linux et Windows.

Le fichier d'index `Packages` est la seule chose nécessaire. Il liste les paquets disponibles sur les dépôts autorisés et quelques informations de base sur chaque paquet.

Ce mécanisme permet de mettre en place facilement un processus de réplication entre plusieurs dépôts.

Les grandes organisations avec des sites distants et des filiales nécessitent parfois que les services soient répliqués localement pour éviter la congestion de la bande passante (*Edge Computing*).

2.2 Dépôts répliqués

WAPT Enterprise offre la possibilité de mettre à niveau les agents distants pour servir de dépôt distants pouvant être gérés directement depuis la console WAPT. Tous les agents WAPT peuvent ensuite être configurés de manière centralisée pour sélectionner automatiquement le meilleur dépôts en fonction d'un ensemble de règles.

Lorsque WAPT est utilisé sur des sites distants à bande passante limitée, il est logique d'avoir un appareil local qui répliquera le dépôt WAPT principal pour réduire la bande passante réseau consommée lors du déploiement des mises à jour sur vos appareils distants.

Avec les dépôts distants, WAPT reste une solution à faible coût d'exploitation car vous n'avez pas besoin de mettre en place **des liaisons fibre haut débit** pour profiter de WAPT.

Cela fonctionne comme suit :

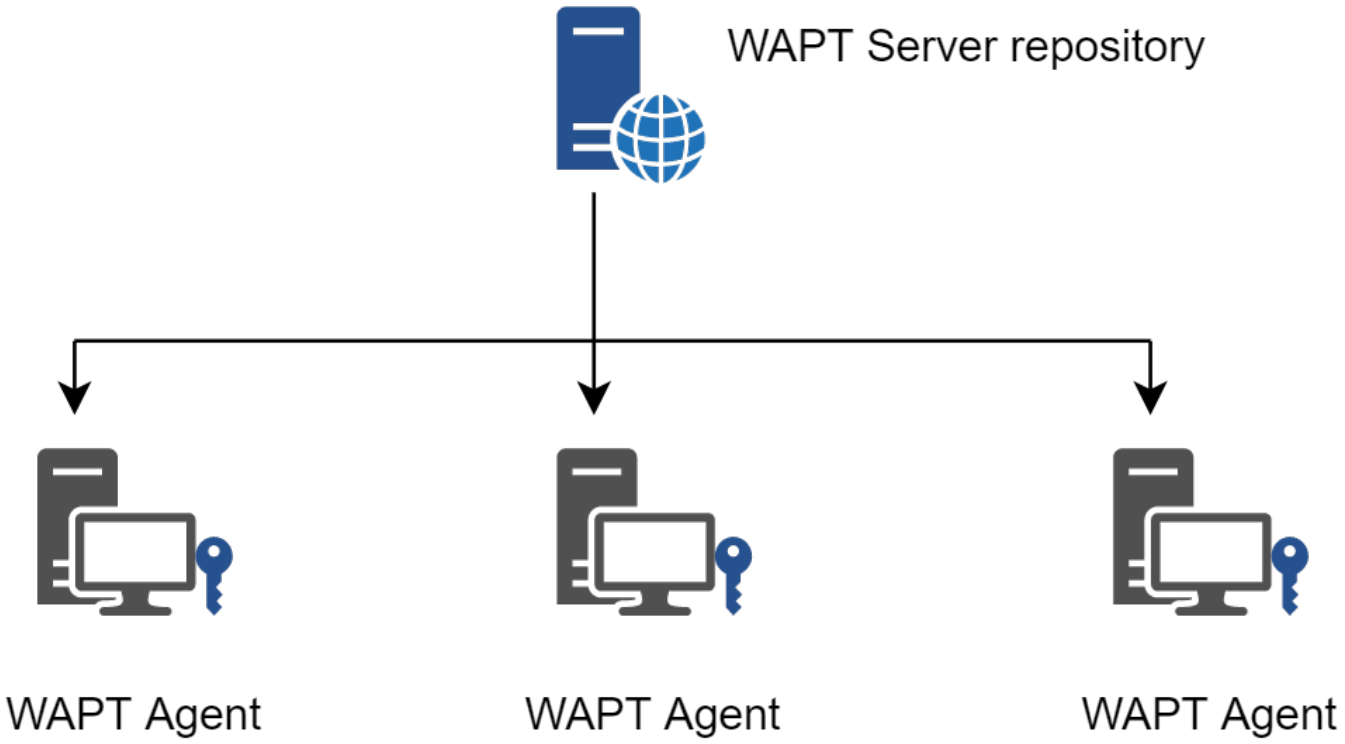


Figure1: Réplication et dépots multiples

- Une appliance de petite taille et sans maintenance jouant le rôle de dépôt secondaire est déployée sur le réseau local de chaque site distant ; un poste de travail peut également être utilisé, même s'il peut ne pas être opérationnel si vous souhaitez vous y connecter.
- Le dépôt distant réplique les paquets du dépôts principal.
- Les agents WAPT se connectent en priorité au dépôts le plus proche d'eux, le dépôt local.

Pour en savoir plus sur le dépôt répliqué, consultez la documentation sur *Réplication d'un dépot*.

2.3 Principe de Paquets

La structure d'un paquet WAPT est similaire à celle d'un paquet **.deb** de Debian Linux. Chaque paquet WAPT embarque avec lui les binaires qui seront exécutés et les autres fichiers dont il aura besoin.

Un paquet est transportable facilement.

Voici à quoi ressemble un package WAPT:

Pour en savoir plus sur la composition d'un paquet WAPT, consultez la documentation sur la *structure détaillée d'un paquet*.

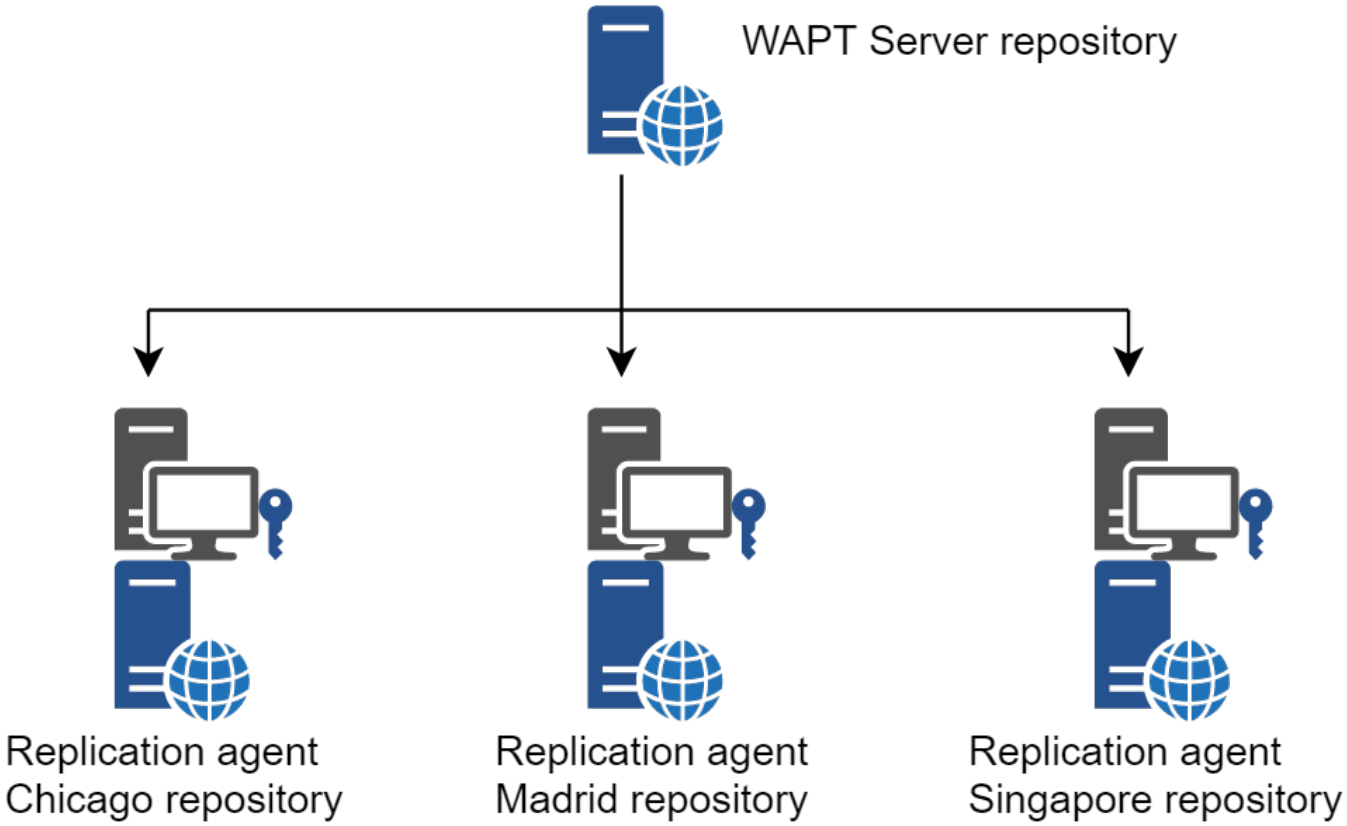


Figure2: Réplication des dépôts WAPT

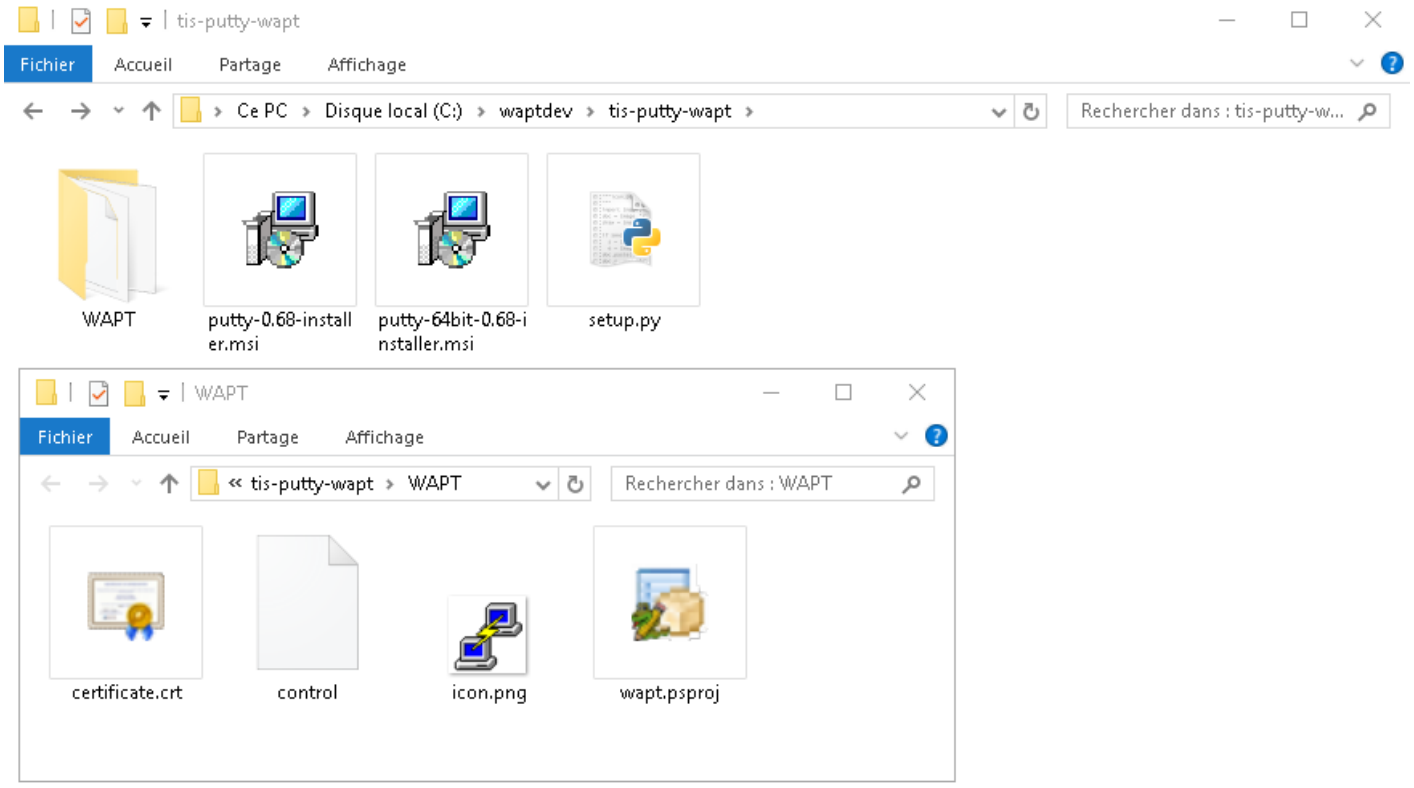


Figure3: Structure du packaging WAPT affichée dans l'explorateur Windows

2.3.1 Types de paquets WAPT

Il existe 8 types de paquets WAPT :

Figure4: Représentation d'un paquet WAPT simple

Les paquets *base*

Ce sont les paquets logiciels classiques.

Ils sont stockés dans le répertoire web https://srvwapt.mydomain.lan/wapt/.

Les paquets *group*

Ce sont des groupes de paquets.

Chaque groupe correspond souvent à:

- service dans l'entreprise (ex : **comptabilité**).
- une pièce, un bâtiment, etc.

Indication: Un client peut-être membre de plusieurs groupes.

Ils sont stockés dans le répertoire web https://srvwapt.mydomain.lan/wapt/.

Les paquets *host*

Les paquets « machines » portent le nom *UUID* Bios ou le *FQDN* de la machine.

Chaque client recherchera son paquet **host** pour connaître les packages qu'il doit installer (*dépendances*).

Les paquets **host** sont stockés dans le répertoire web https://srvwapt.mydomain.lan/wapt/.

les paquets *unit*

Les paquets « unit » portent le nom complet d'une OU (Unité Organisationnelle), exemple: **OU=piece1,OU=prod,OU=computers,DC=mydomain,DC=lan**.

Par défaut, chaque ordinateur recherche les paquets unit puis installe la liste des dépendances associées.

Les paquets **Unit** sont stockés dans le répertoire web https://srvwapt.mydomain.lan/wapt/.

les paquets *wsus*

Les paquets *wsus* contiennent le liste des mises à jour Windows autorisées et interdites.

Lorsque ce paquet est installé sur le terminal, la prochaine analyse de mise à jour effectuée par WAPT choisira les mises à jour Windows en fonction de ce filtrage.

Les paquets **wsus** sont stockés dans le répertoire web https://srvwapt.mydomain.lan/wapt/.

les paquets *self-service*

Les paquets *self-service* contiennent une liste de groupes ou d'utilisateurs (Active Directory ou local) et leurs listes associées de paquets que les utilisateurs seront autorisés à installer par eux-mêmes.

Les paquets **self-service** sont stockés dans le répertoire web https://srvwapt.mydomain.lan/wapt/.

les paquets *profile*

Les paquets *profile* sont similaires aux paquets *group*.

Cependant, les paquets **profile** fonctionnent un peu différemment et sont plus utiles lorsqu'un serveur Active Directory existe dans l'"*Organisation* :

- L'Agent WAPT dressera la liste des groupes Active Directory auxquels la machine appartient.
- Si un paquet *profile* porte le même nom qu'un groupe Active Directory, l'agent WAPT installera automatiquement le paquet *profile* pour le groupe Active Directory dont il est membre.
- Si la machine n'est plus membre de son groupe Active Directory, le paquet *profile* correspondant sera désinstallé.

Les paquets *profile* sont stockés dans le répertoire web https://srvwapt.mydomain.lan/wapt/.

Les paquets *profile* ne sont pas explicitement affectés à la machine (c'est-à-dire en tant que dépendances dans le paquet *host*) mais sont implicitement pris en compte par le moteur de dépendance de l'agent WAPT lors de la mise à niveau WAPT.

Note: Pour des raisons de performances, cette fonctionnalité n'est activée que si l'option `use_ad_groups` est activée dans le fichier de configuration `wapt-get.ini`.

---

les paquets *config*

Les paquets **config** sont utilisés pour modifier les paramètres de configuration des Agents WAPT. Ainsi, il est possible d'avoir un Agent WAPT générique et de personnaliser la configuration des Agents WAPT à l'aide de profils de machines. Par exemple, certaines machines peuvent avoir besoin de différentes règles WAPTWUA et d'autres machines doivent pouvoir être définies avec une maturité **DEV**, etc.

2.4 Principe de dépendance

Dans WAPT tout fonctionne selon le principe de dépendance.

Par défaut, l'agent WAPT recherchera son paquetage *host*. Le paquet *host* liste les paquets à installer sur l'ordinateur.

Ainsi, le paquet *host* sera correctement installé si toutes ses dépendances sont satisfaites.

Chaque sous-dépendance doit être satisfaite pour satisfaire une dépendance de niveau supérieur.

Lorsque toutes les dépendances sont satisfaites, l'hôte notifie son statut au serveur WAPT. Son status devient **OK** et vert dans la console WAPT, ce qui signifie que la machine a le profil de machine que l'"*Administrateur* ou Déployeur de paquets a défini pour elle.

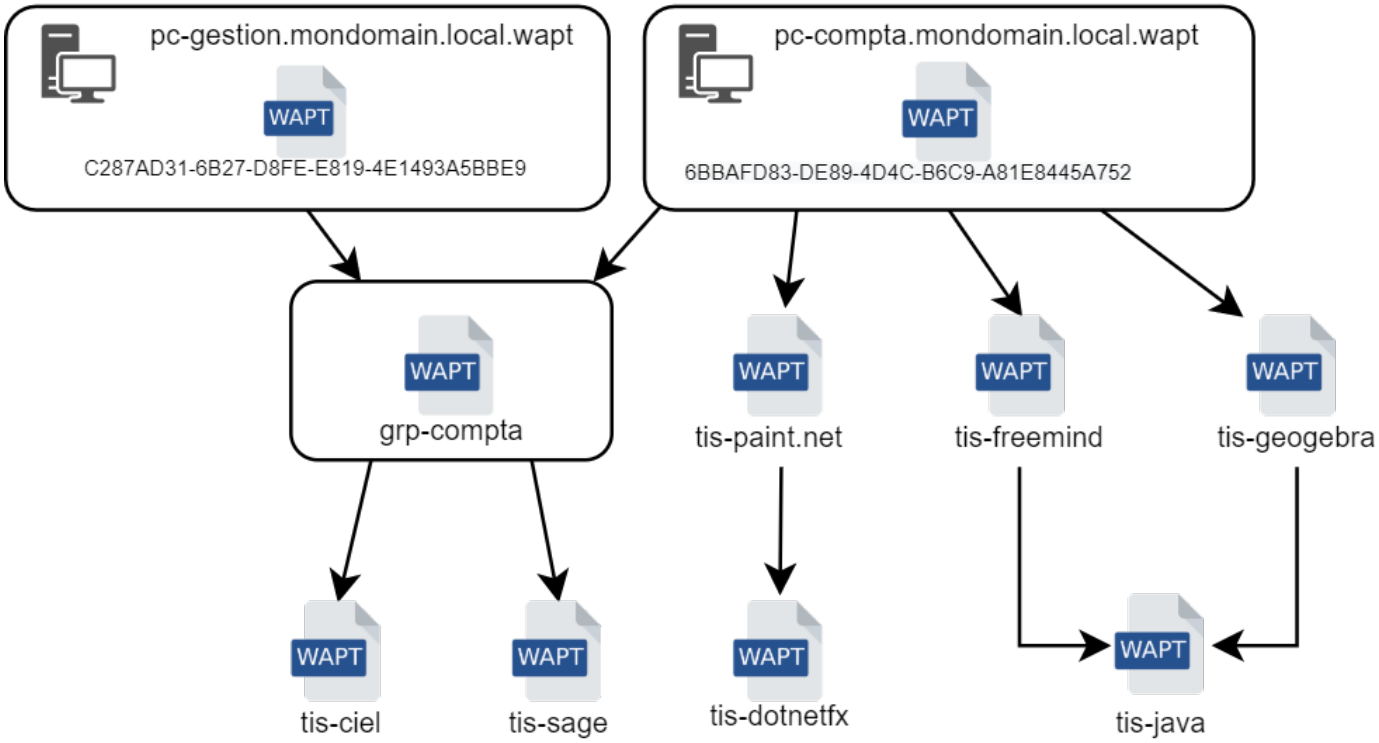


Figure5: Schéma conceptuel du mécanisme de dépendance

Indication: Lorsque l'on attribue un logiciel à un hôte en tant que dépendance, seul le nom canonique du logiciel sans son numéro de version est enregistré comme dépendance (ex : Je veux que Freemind soit installé sur cette machine dans sa dernière version et que **`Freemind`** soit configuré pour que le *User* ne m'appelle pas parce qu'il ne trouve pas l'icône sur son bureau !)

Pour chaque dépendance, l'agent WAPT se chargera d'installer automatiquement la dernière version disponible du paquet. Ainsi, si plusieurs versions de **`Freemind`** sont disponibles sur le dépôt, l'agent WAPT obtiendra toujours la dernière version, à moins que j'aie épinglé la version pour des raisons de compatibilité avec d'autres ensembles d'outils.

Ensuite, lorsque l'agent contacte le dépôt pour vérifier s'il y a de nouvelles mises à jour, il compare les versions des paquets du dépôt avec sa propre liste locale de paquets déjà installés sur la machine.

Si une mise à jour d'un paquet déjà installé est disponible, le client basculera le statut du paquet en **NEED UPGRADE**. Il installera ainsi les mises à jour au prochain **`upgrade`**.

2.5 Principe de Clé privée / certificat public

Comme les paquets Android **APK**, les paquets WAPT sont signés; un hash de la somme de contrôle de tous les fichiers contenus dans le paquet est calculé.

Cette méthode de signature permet de garantir la provenance et l'intégrité du paquet.

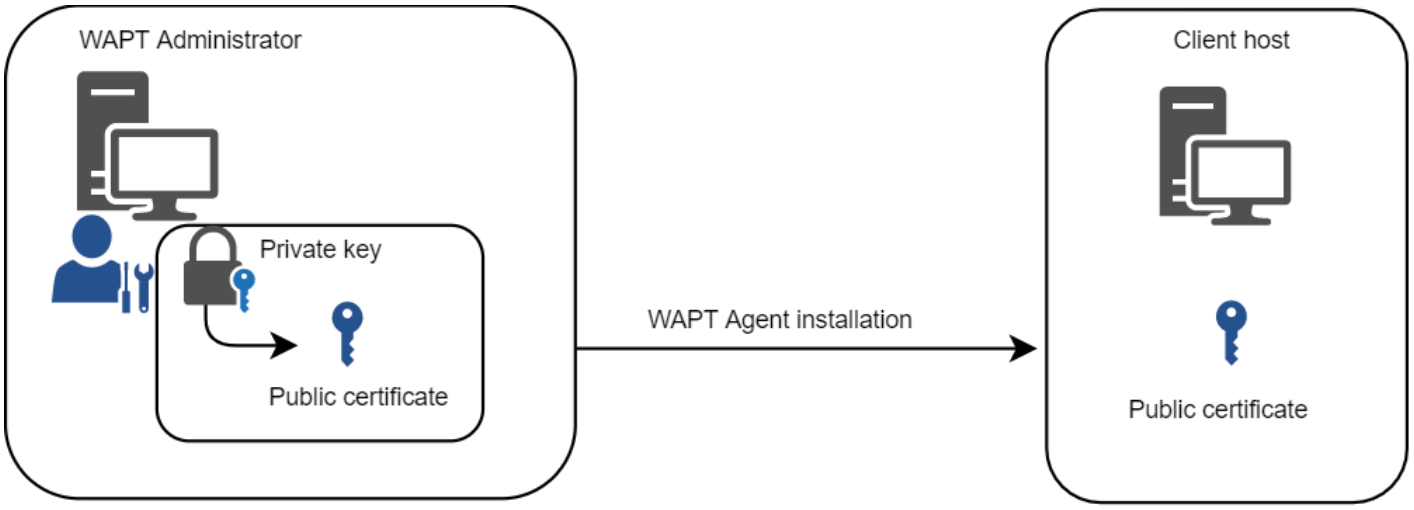


Figure6: Clé privée / certificat public

Pour fonctionner correctement, WAPT a besoin d'une paire clé privée /certificat public (auto-signée, émise par une autorité de certification interne *Certificate Authority* ou commerciale).

La **clé privée** sera utilisée pour **signer** les paquets WAPT tandis que le **certificat public** sera distribué avec chaque agent WAPT afin que les agents WAPT puissent valider les fichiers qui ont été signés avec la clé privée.

Les différents certificats publics seront stockés dans le sous-dossier `ssl` de l'agent WAPT. Ce dossier peut contenir plusieurs certificats publics.

2.5.1 Vérification des paquets

Lorsqu'un paquet WAPT est téléchargé, l'agent WAPT (*waptagent*) vérifie l'intégrité du paquet, puis vérifie que le paquet a été correctement **signé**.

Si la signature du paquet WAPT ne correspond à aucune des certificats publics situées dans `C:\Program Files (x86)\wapt\ssl` sur Windows ou */opt/wapt/ssl* sur Linux et MacOS, l'agent WAPT refusera d'installer le paquet.

Pour plus d'informations, veuillez consulter la documentation sur *comment l'intégrité du processus d'installation d'un paquet WAPT est assurée*.

2.5.2 La clé privée est importante

Attention: La clé privée ne doit **PAS** être stockée sur le serveur WAPT, ni sur aucun stockage public ou partagé auquel pourrait accéder du personnel non autorisé. En effet, la sécurité de WAPT repose sur le maintien de la clé privée **privée**.

La clé privée doit être stockée en lieu sûr, car **celui qui contrôle votre clé contrôle votre parc** !

Enfin, pour un maximum de sécurité, la clé privée pourra être sécurisée sur une smartcard ou un jeton cryptographique que les *Administrateurs* et *Déployeur de Paquet* transporteront physiquement sur eux, utilisant leur smartcard ou leur jeton cryptographique ponctuellement pour signer un paquet WAPT.

Note: La clé privée est protégée par un mot de passe par défaut.

Plus d'informations sur *générer le certificat de l'administrateur pour signer les paquets WAPT*.

2.5.3 Différenciation des rôles des utilisateurs dans WAPT

WAPT offre la possibilité de différencier les rôles en fonction de :

- Une PKI (Infrastructure à Clé Publique) ;
- ACL (Access Control Lists).

Infrastructure à Clé Publique (PKI)

Indication: L'utilisation d'une PKI existante est possible, la Console WAPT est livré avec un générateur simple de certificat.

WAPT fonctionne comme un mode CA (autorité de certification) en ce qui concerne la PKI.

De par sa conception, WAPT est capable de générer des certificats qui peuvent être utilisés comme clés parent pour générer d'autres clés enfant publiques et privées.

Par conséquent, l'administrateur principal de WAPT qui agit en tant qu'administrateur peut émettre des certificats pour chaque administrateur informatique afin que leurs actions puissent être identifiées lorsqu'ils utilisent WAPT.

Les certificats enfants émis par la CA peuvent eux-mêmes être configurés comme :

- La signature de code pour permettre aux administrateurs informatiques d'empaqueter, de signer et de déployer des paquets WAPT contenant des charges exécutables (c'est-à-dire `setup.py`).
- CA pour déléguer à d'autres administrateurs informatiques le droit d'émettre des certificats.
- Aucun droit pour limiter les administrateurs informatiques au seul déploiement de paquets contenant des charges non exécutables (c'est-à-dire configurer des hôtes).

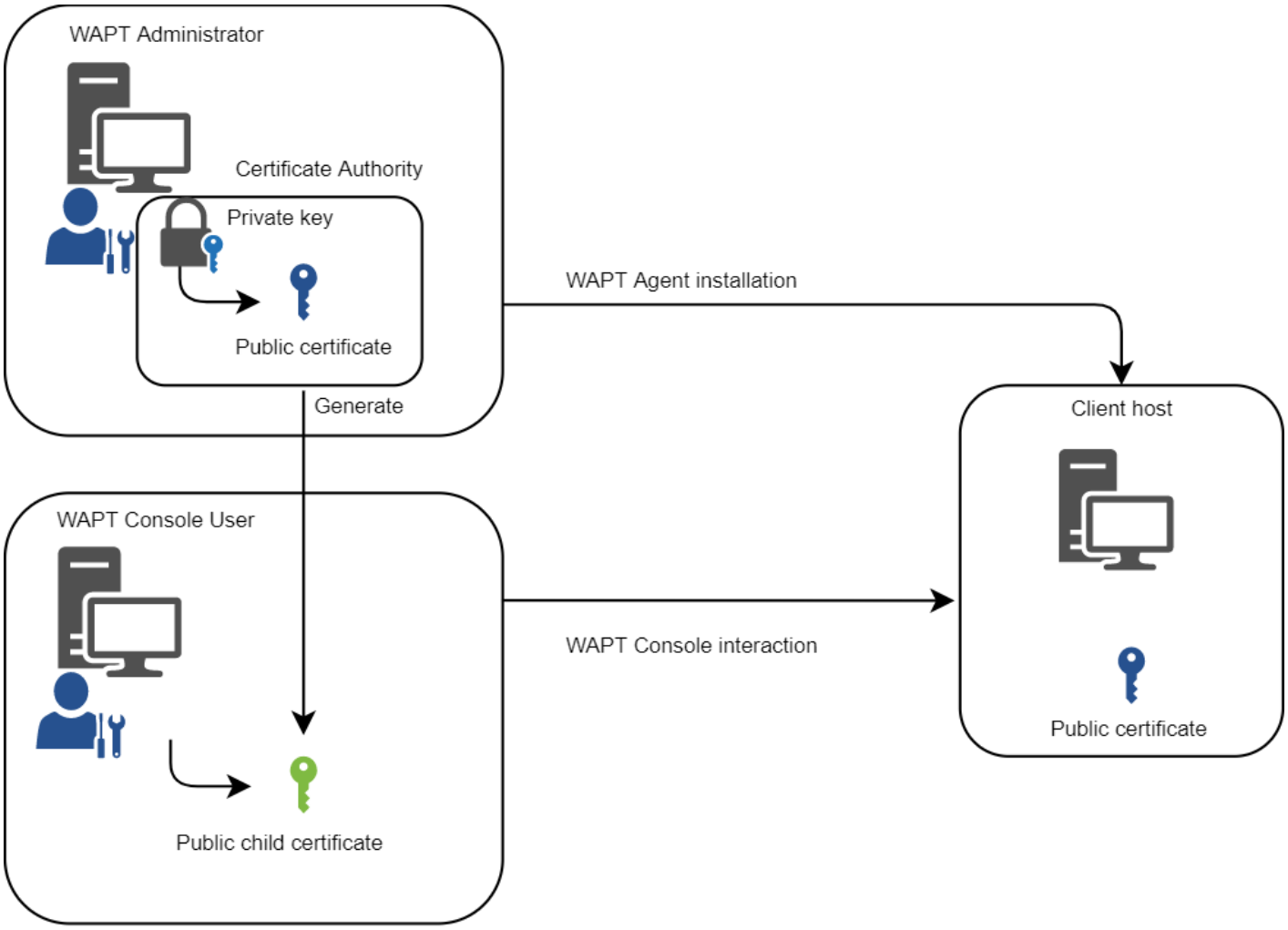


Figure7: Différenciation des rôles des utilisateurs WAPT

Pour en savoir plus sur la génération de l'autorité de certification (CA) avec WAPT, visitez cette documentation.

Liste de contrôle d'accès (ACL)

Avec WAPT, il est possible de définir les droits des utilisateurs en utilisant ACL.

Chaque technicien informatique est identifié par son propre certificat et les droits peuvent donc être appliqués finement sur une base individuelle.

Par exemple, un utilisateur de la console WAPT peut avoir le droit de « Voir » sur une machine mais ne pas être autorisé à cliquer sur « Modifier la machine ».

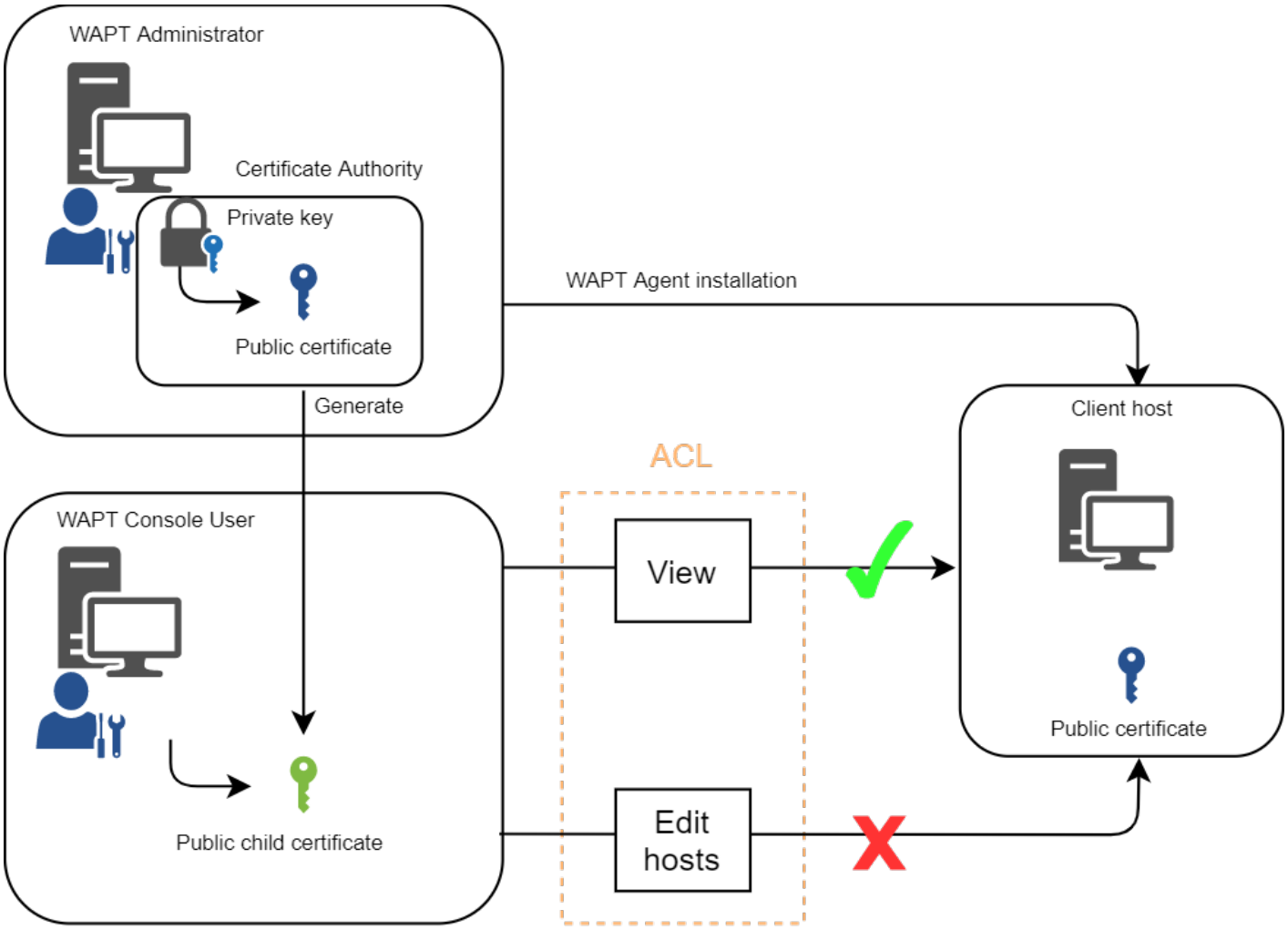


Figure8: Différenciation des rôles ACL

Pour en savoir plus sur l'ACLS dans WAPT, visitez *cette documentation*.

CHAPTER 3

Mode de fonctionnement WAPT

3.1 Inventaire logiciel

WAPT tient un inventaire matériel et logiciel de chaque machine.

Cet inventaire est stocké dans une petite base de données intégrée à chaque agent WAPT.

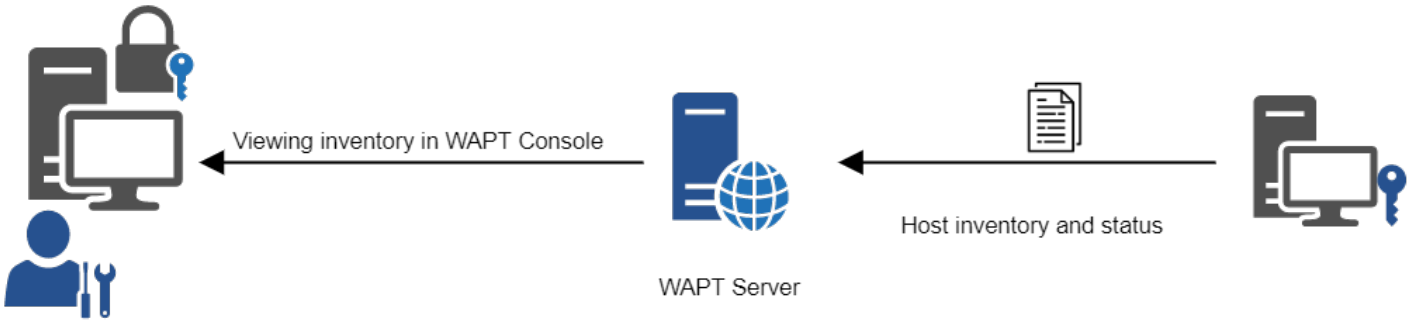


Figure1: Fonctionnement de la remontée d'inventaire

- Lors du premier enregistrement avec le serveur WAPT, l'agent WAPT envoie l'inventaire complet (BIOS, matériel, logiciel) au serveur.
- Lors de chaque mise à jour du client, l'agent WAPT remonte le delta de l'inventaire au serveur.

L'inventaire central vous permet de filtrer les hôtes par leurs composants, leurs logiciels ou tout autre argument de recherche.

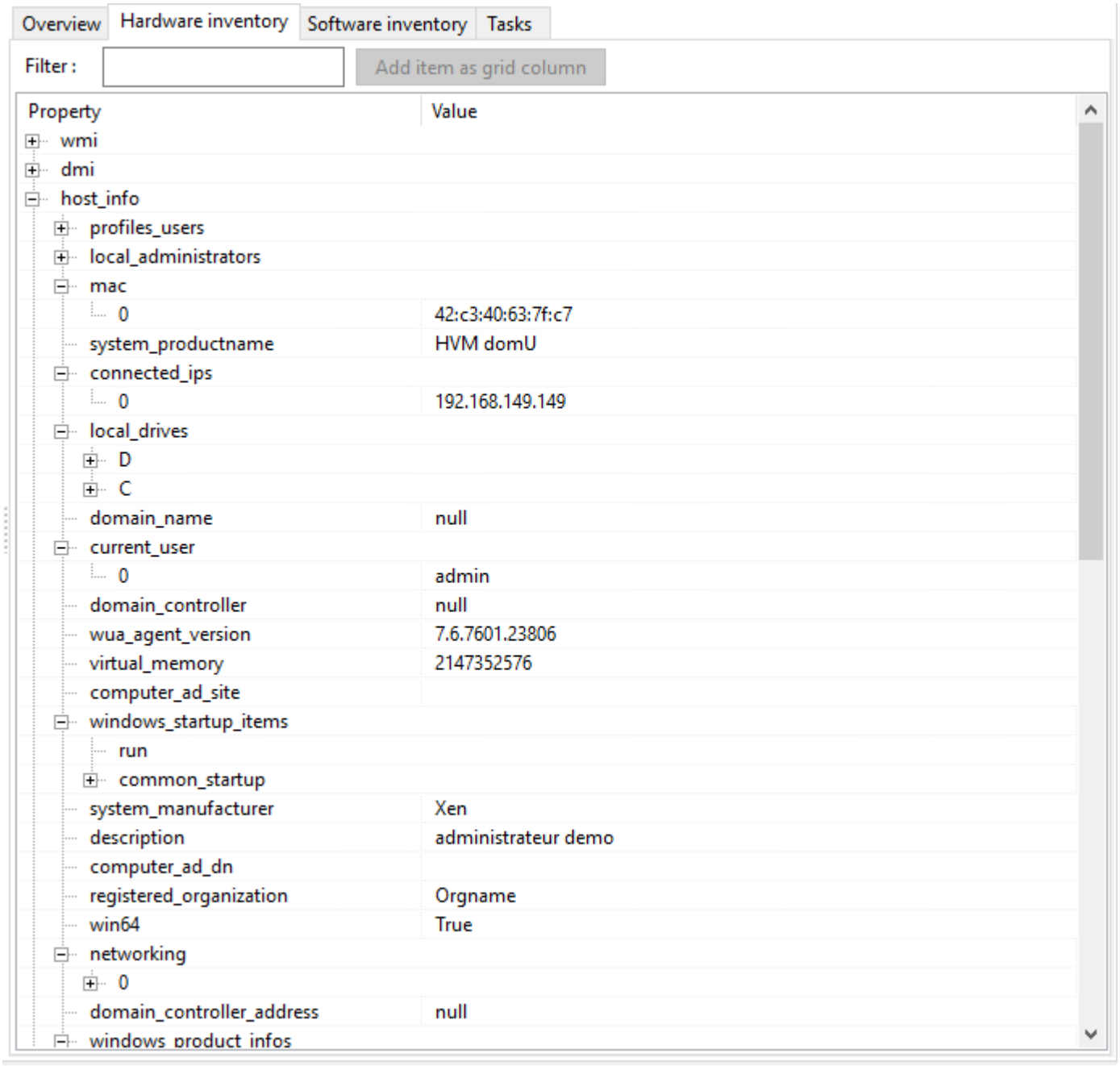


Figure2: L'inventaire dans la console WAPT

3.2 La remontée des informations d'inventaire

L'agent WAPT remonte également le statut des paquets WAPT.

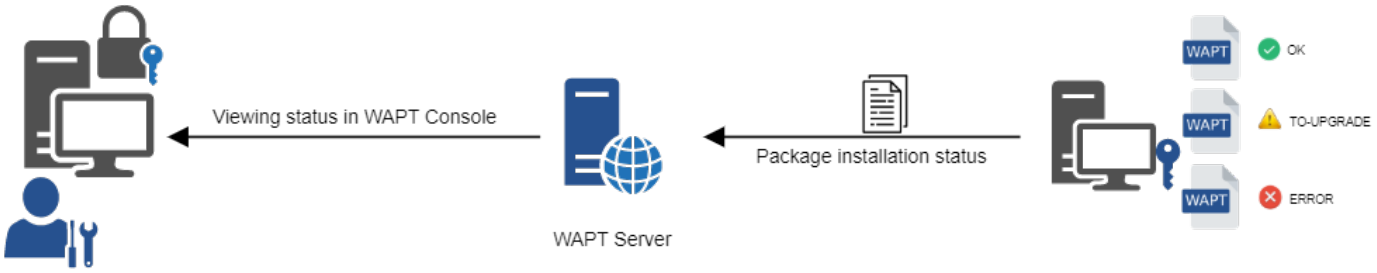
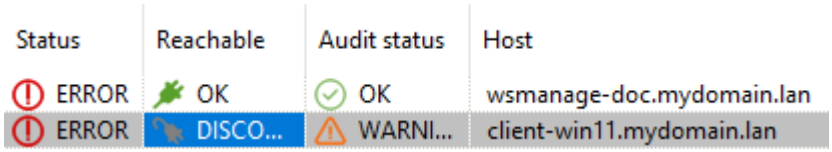


Figure3: La remontée du statut des paquets vers le serveur WAPT

En cas d'erreur lors de l'installation du paquet, l'information sera transmise au serveur WAPT. La machine apparaîtra alors en **ERROR** dans la console.



Les statuts possibles d'un hôte dans la console WAPT sont les suivants :

Table1: Paquets avec un statut d'erreur dans la console WAPT

Statut	Icône d'état
OK	
upgrade	upgrade
NEED-REMOVE	
ERROR	
NEED-INSTALL	Le statut de l'hôte nécessite une installation

Le *Administrator* peut voir le paquet retourné en erreur dans la console et corriger le paquet en conséquence.

Pour chaque **`upgrade`**, WAPT essaiera d'installer une nouvelle version du paquetage jusqu'à ce qu'aucun statut d'erreur ne soit renvoyé.

Note: Les agents WAPT signent leur inventaire avant de l'envoyer au serveur WAPT.

Pour plus d'informations, veuillez vous reporter à *Signature des remontées d'inventaire*.

3.3 Les interactions classiques de WAPT

3.3.1 update

Lorsqu'une commande **update** est lancée sur un agent, cela revient à ordonner à l'agent de vérifier le dépot WAPT pour les nouveaux paquets. **Par défaut, l'agent WAPT recherche les mises à jour toutes les deux heures**.

Si la date du fichier d'index `Packages` a changé depuis la dernière **update**, alors l'agent WAPT télécharge le nouveau fichier `Packages` (entre 20 et 100k), sinon, il ne fait rien.

L'agent WAPT compare ensuite le fichier `Packages` avec sa propre base de données locale.

Si l'agent WAPT détecte qu'un paquet doit être ajouté ou mis à jour, il fait passer le statut de l'hôte et celui du paquet à *NEED-UPGRADE*.

Il ne lancera pas l'installation du paquet immédiatement. L'agent WAPT attendra un ordre « **upgrade** » pour lancer la mise à niveau.

3.3.2 upgrade

Lorsque nous lançons une **upgrade**, nous demandons à l'agent WAPT d'installer les paquets ayant un statut *NEED-UPGRADE*.

Une **update** doit précéder une **upgrade**, sinon l'agent ne saura pas si des mises à jour sont disponibles.

Par défaut, l'agent WAPT déclenchera une **update**/ **download-upgrade** au démarrage ; après le démarrage, l'agent WAPT vérifiera ensuite toutes les 2 heures s'il a quelque chose à faire.

Les paquets à installer seront téléchargés et mis en cache dans le dossier `C:\Program Files (x86)\wapt\cache`.

waptexit lancera un **wapt-get upgrade** lorsque l'ordinateur s'éteindra. Un *Administrateur* peut forcer le lancement immédiat d'une **mise à niveau** à partir de la console WAPT. Alternativement, un utilisateur final peut choisir de lancer manuellement une *mise à niveau*. Enfin, une tâche planifiée peut être configurée sur les hôtes pour lancer une *mise à niveau*.

Si le serveur WAPT n'est pas joignable lors de la mise à niveau, l'agent WAPT sera toujours capable d'installer les paquets mis en cache.

Les 5 objectifs de l'agent WAPT sont donc :

- Pour installer un paquet **base**, un **groupe** ou un **unité** s'il est disponible.
- De supprimer les paquets obsolètes.
- Pour résoudre les dépendances et les conflits de paquets.
- Pour s'assurer que tous les paquets WAPT installés sont à jour par rapport à ceux stockés dans le dépot.
- Pour mettre régulièrement à jour le serveur WAPT avec son état matériel et l'état des logiciels installés.

3.4 Comportement de l'agent WAPT

Un concept clé qui peut être difficile à comprendre est le comportement d'un agent WAPT lors de l'installation d'un paquet et les considérations qui l'entourent.

L'installation du paquet d'agents WAPT peut être divisée en étapes simples:

- Lors du déclenchement d'un **update**, l'agent télécharge les paquets *NEED-UPGRADE* ou *NEED-INSTALL* et les stocke dans le dossier cache.

- Lors du déclenchement d'un **upgrade**, l'agent décompresse les paquets dans un dossier temporaire.
- Le contenu du `setup.py` est analysé et stocké dans la base de données de l'agent WAPT située dans `C:\Program Files (x86)\wapt\db\waptdb.sqlite`.
- Le `setup.py` est exécuté et le logiciel est installé à partir des fichiers décompressés.
- En cas de succès : les paquets téléchargés et les fichiers dézippés sont supprimés. Un statut **OK** est renvoyé au serveur WAPT.
- En cas d'échec : les paquets téléchargés sont conservés et les fichiers dézippés sont supprimés. Un statut **ERROR** est renvoyé au serveur WAPT.

Ce comportement est important pour comprendre le cycle de vie d'un paquet installé.

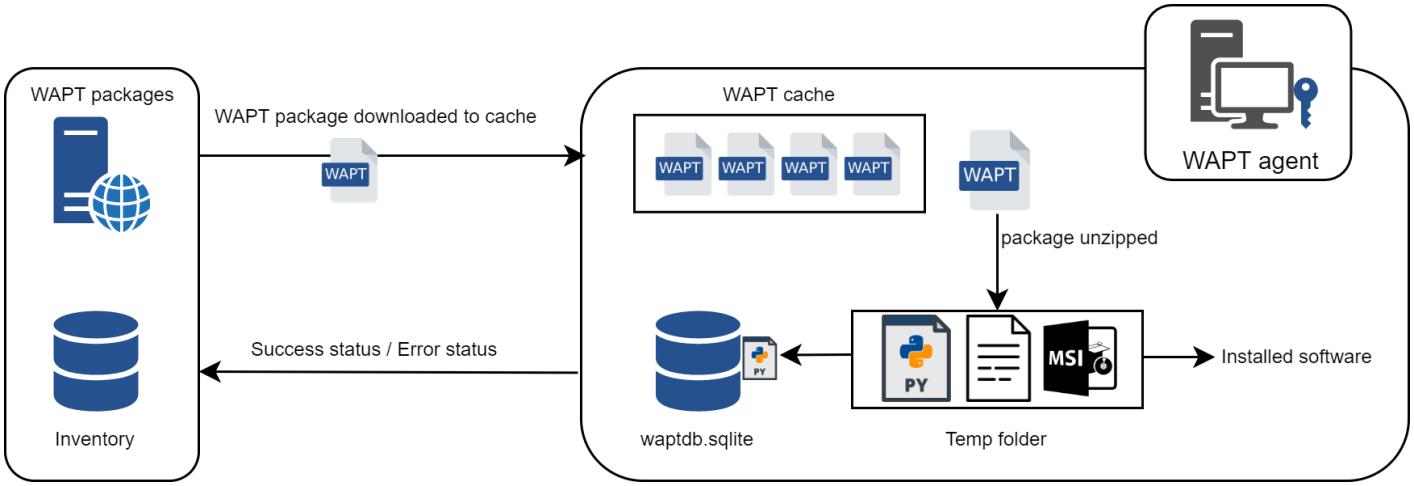


Figure4: Diagramme de flux montrant le processus d'installation d'un packaging WAPT

Par exemple, lors du retrait d'un paquet, les étapes suivantes sont suivies :

- Le contenu du `setup.py` est extrait de la base de données de l'agent WAPT située dans `C:\Program Files (x86)\wapt\db\waptdb.sqlite`.
- L'agent WAPT recherche le `UninstallString` dans la base de données locale.
- Si elle est définie dans le `setup.py` copié dans la base de données locale lors de l'installation initiale du paquetage WAPT, la fonction **`uninstall()`** est exécutée.

Des étapes similaires sont reproduites lors de l'exécution de **`session_setup`** et **`audit`**.

3.5 Diagramme complet du fonctionnement de WAPT

Nous retrouvons ici le comportement commun de WAPT, depuis la duplication d'un paquet à partir d'un dépôt externe accessible sur Internet, jusqu'à son déploiement sur les machines du réseau.

Lire le diagramme dans le sens des aiguilles d'une montre :

- Importer des paquets depuis un dépôt externe (ou créer un nouveau paquet à partir de zéro).
- Tester, ensuite valider, puis construire et enfin signer le paquet.
- Télécharger le paquet sur le dépôt principal.

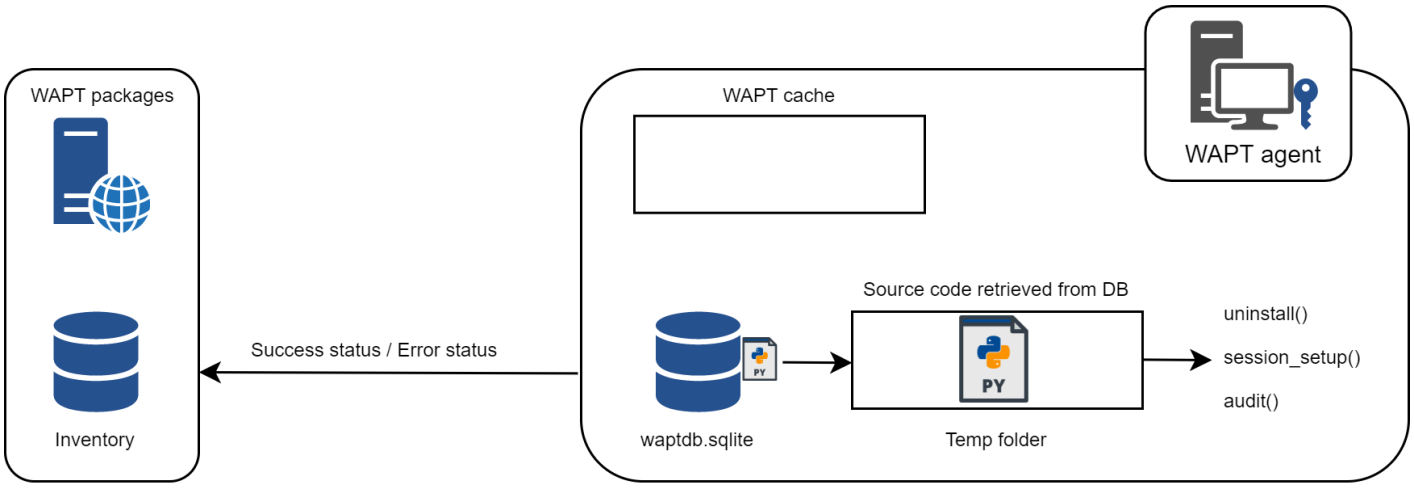


Figure5: Comportement de l'agent WAPT avec la désinstallation, le session_setup et l'audit

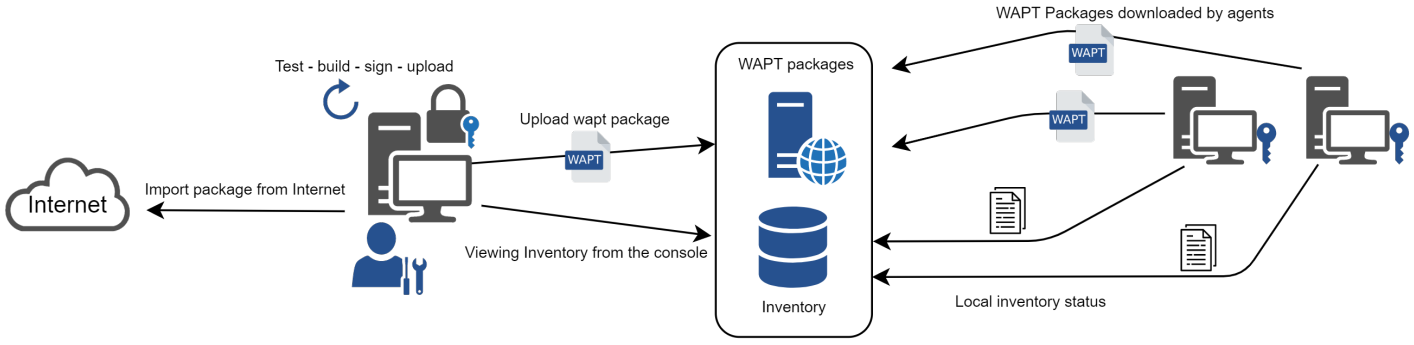


Figure6: Diagramme de flux montrant le mode de fonctionnement général avec WAPT

- Les paquets sont automatiquement téléchargés par les clients WAPT.
- Exécution des paquets selon la méthode sélectionnée:
  - L'*Administrateur* force l'"**upgrade**.
  - L'*Administrateur* propose l'" **upgrade** à l'"*Utilisateur*.
  - Une tâche planifiée lance l'exécution de la mise à jour.
  - La mise à jour est exécutée à l'extinction de la machine.
  - L'*Utilisateur* choisit le bon moment pour lui-même (à l'arrêt ou en utilisant le *self-service*).
- Remontée des informations d'inventaire.
- Consultation de la remontée d'inventaire via la console.

CHAPTER 4

Architecture du serveur WAPT

L'architecture du serveur WAPT repose sur plusieurs rôles distincts:

- Le *rôle de dépôt* pour la distribution des packages.
- Le rôle *inventaire* et *serveur central* pour l'inventaire du matériel et des logiciels.
- Le rôle *proxy* pour relayer les actions entre la console WAPT et les agents WAPT.

4.1 Fonctionnement du dépôt WAPT

Tout d'abord, le serveur WAPT sert de dépôt de fichiers web.

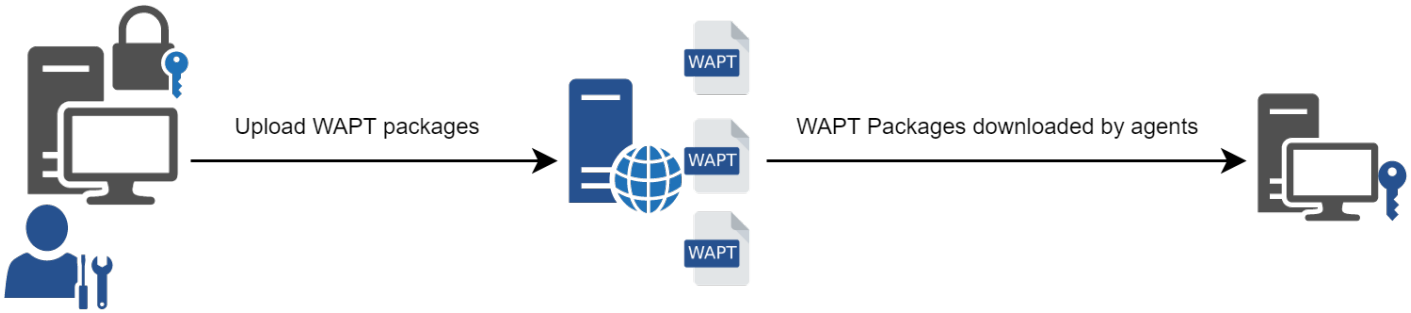


Figure1: Schéma conceptuel du mécanisme de dépendance

- Ce rôle de dépôt est accompli par un serveur web Nginx.
- Le dépôt permet la distribution des paquets WAPT, des installeurs *waptagent* et *waptsetup*.
- Les paquets WAPT sont accessibles avec un navigateur web à l'adresse https://srvwapt.mydomain.lan/wapt.
- Les paquets **host** sont contenus dans un répertoire inaccessible par défaut (https://srvwapt.mydomain.lan/wapt/wapt-host/).

4.2 Rôle d'inventaire

Deuxièmement, le serveur WAPT sert de serveur d'inventaire.

Le serveur d'inventaire est un service passif qui collecte les informations que les agents WAPT lui envoient :

- Inventaire matériel.
- Inventaire logiciel.
- Statut des paquets WAPT.
- Etat des tâches (*running*, *pending*, *error*).

Note: Le service WAPT n'est pas actif dans le sens où il ne fait que recevoir des informations des clients. Par conséquent, si le serveur d'inventaire tombe en panne, l'inventaire se rétablira de lui-même à partir des rapports d'état d'inventaire reçus des agents WAPT déployés.

Dans la version **Discovery** de WAPT, l'accès aux données d'inventaire n'est possible que par la console WAPT.

WAPT **Enterprise** est livré avec des capacités *reporting*. En parallèle, il est possible de pousser l'inventaire WAPT vers l'outil ITSM *GLPI*.

4.3 Rôle de Proxy

Troisièmement, le serveur WAPT sert de proxy de commande.

Il sert de relais entre la console de gestion WAPT et les agents WAPT déployés.

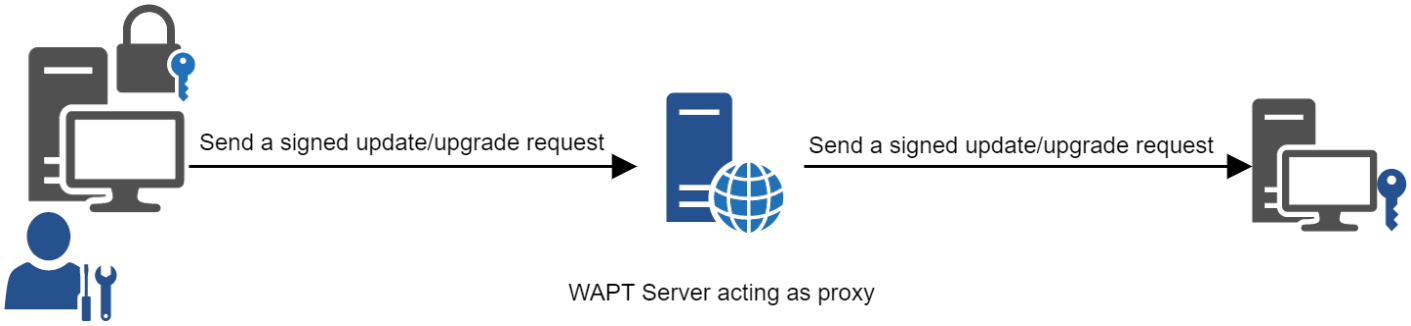


Figure2: Schéma conceptuel du mécanisme de dépendance

Note: Chaque action déclenchée sur un agent WAPT à partir du serveur WAPT est signée avec une clé privée.

Sans clé privée valide, il n'est pas possible de déclencher des actions à distance sur des appareils distants équipés de WAPT.

Pour plus d'informations sur les actions à distance, veuillez consulter la documenation sur *les actions de signature relayées aux agents WAPT*.

CHAPTER 5

Langage et environnement de développement WAPT

WAPT est construit en utilisant le langage Python.

Attention: Avec WAPT 2.0, les internes de WAPT sont passés à python3.

Les packages WAPT DOIVENT MAINTENANT suivre la nouvelle syntaxe python3.

Consultez *cette documentation* pour vous aider à identifier les problèmes potentiels lors du passage de vos paquets existants de Python2 à Python3.

Tout environnement de développement rapide d'applications destiné au développement de Python convient.

Tranquil IT a développé quelques plugins spécifiques WAPT utiles pour l'IDE **PyScripter** (https://sourceforge.net/projects/pyscripter).

Tranquil IT recommande d'utiliser **PyScripter** pour développer des paquets WAPT pour Windows et **vscode** pour développer des paquets WAPT pour macOS et Linux.

5.1 La puissance de Python

Toute la puissance de **Python** peut être avantageusement mise à profit.

De nombreuses bibliothèques existent déjà en Python pour :

- Faire des boucles conditionnelles (si … alors … autrement …).
- Copier, coller, déplacer des fichiers et des répertoires.
- Vérifier si les fichiers ou les répertoires existent.
- Vérifier si les clés de registre existent.
- Vérifier les droits d'accès, modifier les droits d'accès.

- La recherche d'informations sur des sources de données externes (LDAP, bases de données, fichiers, etc.).
- Et plus.

5.2 La puissance de WAPT

Les fonctions les plus couramment utilisées avec WAPT ont été simplifiées dans des bibliothèques appelées *Setuphelpers*.

Les fonctions **Setuphelpers** simplifient le processus de création et de test des paquets WAPT, validant ainsi les principaux objectifs de WAPT :

- **Ce qui était compliqué est rendu simple**.
- **Ce qui était simple est rendu trivial**.

CHAPTER 6

Historique des éditions et des versions de WAPT

Table1: Cycle de vie des logiciels

	24 Jan 2020	30 Mar 2021	30 Oct 2021	15 Mar 2022	30 Apr 2022	30 Jun 2022	10 Jan 2023	13 Jun 2023
Enterprise Discovery								Release 3.0 (To Be Defined)
Enterprise Discovery							Release 2.3	Maintien de la sécurité et des corrections de bogues
Enterprise Discovery				**:vert:`Version 2.2`**	Maintien de la sécurité et des corrections de bogues	Maintien de la sécurité et des corrections de bogues	Maintien de la sécurité uniquement	End Of Life
Enterprise			**:vert:`Version 2.1`**	Maintien de la sécurité et des corrections de bogues	Maintien de la sécurité et des corrections de bogues	Maintien de la sécurité uniquement	End Of Life	
Enterprise		**:vert:`Version 2.0`**	Maintien de la sécurité et des corrections de bogues	Maintien de la sécurité uniquement	Maintien de la sécurité uniquement	End Of Life		
Enterprise	**:vert:`Version 1.8`**	Maintenance de la sécurité et des corrections de bogues	Maintien de la sécurité uniquement	Maintien de la sécurité uniquement	Maintien de la sécurité uniquement	End Of Life		
Community	**:vert:`Version 1.8`**	Maintenance de la sécurité et des corrections de	Maintien de la sécurité uniquement	Maintien de la sécurité uniquement	End Of Support by Tranquil IT, Community support only			

6.1 Résumé des principes de fonctionnement de WAPT

- **WAPT est basé sur un agent qui n'autorise aucun port entrant ouvert** dans les pare-feu de la machine et qui initie un websocket bidirectionnel sécurisé avec le serveur pour permettre des rapports et des actions en temps réel.
- WAPT fonctionne avec des passerelles de données de confiance en utilisant une simple planification des tâches.
- WAPT fonctionne sur le principe de l'extraction progressive des mises à jour, puis de l'application des mises à niveau au moment opportun (fonctionne avec une bande passante faible ou intermittente, une latence élevée et des réseaux à forte instabilité).
- WAPT n'a pas besoin d'un Active Directory pour fonctionner (fonctionne aussi avec l'édition familiale de Windows) ; cependant, WAPT montrera la machine dans son emplacement Active Directory si l'hôte est joint à un AD.
- Méthodes pour déployer l'agent WAPT:
  1. En utilisant une GPO (Group Policy Object) ou un script Ansible.
  2. Après avoir téléchargé manuellement l'agent depuis le serveur WAPT ou en utilisant SSH (Secured Shell).
- Méthodes d'enregistrement des machines auprès du serveur WAPT:
  1. Automatiquement en utilisant le compte kerberos de la machine.
  2. Manuellement avec le login et le mot de passe WAPT *Superadmin*.
- Des mises à niveau peuvent être déclenchées:
  1. Lors de l'arrêt de la machine, c'est le mode standard.
  2. Par un Administrateur WAPT autorisé en cas d'urgence (ex : vulnérabilités critiques courant dans la nature).
  3. Par l'utilisateur au moment qu'elle choisit (ex : chariot de soins infirmiers 24/7 non utilisé pendant les pauses par un simple clic).
  4. Via une tâche planifiée s'exécutant à une heure prédéterminée (idéal pour les serveurs).
- La sécurité est assurée avec:
  1. La signature des paquets WAPT en utilisant la cryptographie asymétrique.
  2. L'authentification des hôtes par rapport au serveur WAPT en utilisant la cryptographie symétrique lors de l'enregistrement.
  3. La confidentialité du serveur WAPT en utilisant les certificats des clients déployés par WAPT.
  4. L'utilisation de ACL pour définir ce qu'un administrateur est autorisé à voir ou les actions qu'il est autorisé à effectuer en fonction de son certificat.

6.2 Liste des caractéristiques actuelles en date du 2023-04-17

Attention: Vous pouvez trouver sur Internet la mention d'une version GPLv3 **Community** de WAPT qui a été maintenue et supportée par Tranquil IT jusqu'à la version 1.8.2, soit jusqu'à environ juillet 2021.

La version **Community** du WAPT a été forkée de manière amicale. **Tranquil IT ne fournit plus aucun support, ni aucune maintenance, qu'elle soit gratuite ou payante sur WAPT =< 1.8.2**. Le support et la maintenance peuvent être obtenus auprès des opérateurs du *fork* à leurs tarifs et conditions.

[1] WAPT 1.8.2 Community est supporté par Tranquil IT jusqu'au 2022-04-30. Après cette date, le support sera assuré par la communauté uniquement.

Tranquil IT est le seul auteur et le titulaire intégral des droits d'auteur de WAPT 1.8.2 et exigera des responsables de *friednly forks* qu'ils s'abstiennent d'utiliser le nom *WAPT* car la marque WAPT est déposée et protégée par l'Institut National de la Propriété Intellectuelle (INPI) en France et dans le monde.

Table2: Comparaison des caractéristiques entre les versions WAPT en date du 2023-04-17

Caractéristique	Enterprise	Discovery
Déploiement, mise à jour et suppression des logiciels sur les hôtes	✅	✅
Maintenance et support (voir note de bas de page pour les conditions)	Équipe Tranquil IT	Forum Tranquil IT
Sous lincence	Propriétaire	Propriétaire
Limitation du nombre d'appareils	aucune limite	300
Version de Python utilisée dans le code et les paquets WAPT	3+ (actuel)	3+ (actuel)
Déployer et mettre à jour les **configurations dans le contexte du SYSTÈME**	✅	✅
Déployer et mettre à jour les **configurations dans le contexte de l'UTILISATEUR**	✅	✅
Obtenez un **inventaire complet** du matériel, des logiciels et des paquets WAPT appliqués	✅	✅
Bénéficier du **self-service différencié** (les utilisateurs autorisés peuvent installer les logiciels autorisés à partir du store de paquets WAPT autorisés)	✅	❌
Bénéficiez de **Mises à jour Windows simplifiées** qui fonctionnent mieux qu'un WSUS standard (seules les KB requises sont téléchargées depuis Microsoft)	✅	❌
Simplifiez et structurez votre charge de travail administrative en appliquant des paquets WAPT à vos UO (Unités d'Organisation)	✅	❌
Configurer et gérer facilement les dépôts decondaires WAPT **pour préserver la bande passante** pour les scénarios *Edge Computing*	✅	❌
Accédez à des **paquets WAPT prêts à être déployés** pour des logiciels communs gratuits	✅	✅

suite sur la page suivante

Table 2 – suite de la page précédente

Caractéristique	Enterprise	Discovery
Travailler avec des **recettes python facilement vérifiables** pour l'installation, la mise à jour et la suppression de logiciels et de configurations	✓	✓
Bénéficiez de **centaines d'assistants** pour simplifier le conditionnement des logiciels	✗[3]	✓
Chiffrez vos données sensibles pour le transport (clés de licence de logiciel, login, mot de passe, FQDN du serveur, informations API pour l'enregistrement du logiciel auprès du fournisseur, etc)	✓	✗
Automatisez l'audit de vos configurations pour une conformité **facile, automatisée et toujours à jour**	✓	✗
Profitez de la puissance de SQL intégrée à la console WAPT pour créer les **rapports dont vous avez besoin pour votre travail quotidien d'administrateur système ou dont votre organisation a besoin pour prendre des décisions budgétaires**	✓	✗
Authentifiez vos *Administrateurs* WAPT avec **Active Directory, LDAP, ou avec leurs certificats personnels**	✓	✗
Bénéficiez de rôles différenciés entre vos *Développeurs de Paquets* et vos *Déployeurs de Paquets* afin que vous puissiez **déléguer vos pouvoirs WAPT aux personnes les plus adéquates** (les développeurs de paquets connaissent les implications en matière de sécurité, les déployeurs connaissent les besoins des utilisateurs)	✓	✗
Bénéficier du mode multi-tenant et multi-client avec les ACLs (Access Control Lists) pour les MSPs (Managed Service Providers) ou les grandes organisations multi-départementales ou internationales utilisant un mécanisme interne basé sur la PKI pour le périmètre autorisé	✓	✗
Intégration avec Mesh Central pour un simple *partage d'écran* pour le support utilisateur	✓	✗

suite sur la page suivante

Table 2 – suite de la page précédente

Caractéristique	Enterprise	Discovery
Poursuite de la prise en charge de Windows XP dans WAPT pour les machines-outils d'usine, les équipements médicaux des hôpitaux, les instruments de recherche coûteux et difficiles à remplacer, etc	✓	✗
Mise à jour des paquets directement dans la console WAPT avec la fonction `update_package`	✓	✗
Intégrer l'inventaire WAPT avec l'outil populaire GLPI ITSM (IT Service Management)	✓	✗
Outil de déploiement d'images de systèmes d'exploitation intégré à WAPT	✓	✗
Vérifiez le paquet avec www.virustotal.com	✓	✓
Vérifié et approuvé par l'agence de cybersécurité internationalement reconnue ANSSI, **WAPT est le seul logiciel de déploiement au monde à posséder ce niveau de certification**	✓	✗
Redémarrage et arrêt à distance des ordinateurs clients	✓	✗
Envoyer un message au format html aux utilisateurs connectés	✓	✗
Déployer des paquets de configuration WAPT pour modifier facilement la configuration des agents WAPT distants	✓	✗
Télécharger des versions plus récentes du paquet WAPT public directement depuis le dépôt local	✓	✗

6.3 Fonctionnalités à venir

Vous trouverez ci-dessous une liste de fonctionnalités que nous avons identifiées comme étant vraiment utiles à WAPT et à la communauté des utilisateurs de WAPT et sur lesquelles nous avons déjà commencé à travailler. Aucun calendrier n'est promis, restez à l'écoute, nous vous promettons seulement que nous travaillons très dur pour atteindre ces objectifs.

[3] La version Enterprise intègre plus de fonctions SetupHelper que les versions **Community** et **Discovery**.

Caractéristique	Enterprise	Discovery
Historique des actions effectuées via WAPT pour un rapport complet du cycle de vie de la maintenance d'un logiciel hôte	✓	✗
Authentification des administrateurs WAPT à l'aide de jetons cryptographiques (ex : cartes à puce)	✓	✗
Accès à des paquets WAPT prêts à être déployés ou des squelettes de paquets pour des logiciels d'entreprise sous licence (logiciels d'entreprise courants pour l'industrie, le secteur médical, les bureaux, les collectivités publiques, la cybersécurité, etc.)	✓	✗
Accès aux extensions de paquets WAPT prêtes à être déployées pour simplifier le blindage du bureau en utilisant Applocker ou équivalent	✓	✗

6.4 Principaux avantages fonctionnels de la version Entreprise de WAPT

WAPT **Discovery** est conçu pour vous permettre d'essayer gratuitement WAPT sur un périmètre limité et avec des fonctionnalités haut de gamme limitées.

Avec WAPT **Enterprise**, vous bénéficiez automatiquement des fonctions de base incluses dans WAPT pour vous aider à déployer, mettre à niveau et supprimer des logiciels et des configurations sur vos appareils Windows, Linux et MacOS, à partir d'une console centrale, avec de nombreux autres avantages.

WAPT est un modèle *libre*. La version **Enterprise** partage la même base de code que la version **Discovery**. Une clé de licence **Enterprise** active permet d'activer les fonctionnalités supplémentaires suivantes :

- **Authentification Active Directory**

  ses développeurs de paquets WAPT, des déployeurs de paquets, des utilisateurs du self-service et pour l'enregistrement initial des agents WAPT auprès du serveur WAPT. En outre, l'affichage des appareils équipés de WAPT dans la console WAPT suit la même structure que la structure hiérarchique de l'Active Directory OU de l'organisation.

- **Séparation des rôles entre les développeurs de paquets et les déployeurs de paquets**.

  De cette façon, les équipes informatiques centrales peuvent construire les progiciels parce qu'elles connaissent les directives de sécurité de l'Organisation, et les équipes informatiques locales peuvent déployer les progiciels WAPT parce qu'elles connaissent les besoins de leur base d'utilisateurs.

  Une telle séparation est mise en œuvre à l'aide de jeux de clés différenciés (c'est-à-dire des certificats SSL **Code Signing** pour les développeurs de paquets et des certificats SSL **Simple** pour les *déployeurs* de paquets) et avec des rigths ACL.

- **ACL**.

  Les ACLs sont gérées par le *SuperAdmin* pour autoriser ou restreindre les *Administrators* WAPT à visualiser des informations ou à effectuer des actions uniquement sur un sous-ensemble de dispositifs enregistrés auprès du serveur WAPT.

  Les processus d'identification et d'authentification reposent soit sur l'utilisation d'Active Directory, de LDAP ou de certificats. Les autorisations accordées aux administrateurs sont gérées dans la base de données du serveur WAPT. Le périmètre des dispositifs sur lesquels les droits sont accordés est défini par le certificat de l'administrateur déployé.

  Cette fonction est particulièrement utile pour les grandes organisations multinationales, les administrations centrales avec de grands bureaux régionaux ou pour les MSP (Managed Service Providers) qui souhaitent centraliser la gestion de plusieurs clients tout en permettant à leurs clients finaux d'effectuer certaines tâches de gestion quotidiennes.

- **Libre service différencié**.

  WAPT Enterprise vous permet d'appliquer des listes de paquets autorisés à des groupes d'utilisateurs dans Active Directory.

Les utilisateurs autorisés sont libres d'installer des paquets qualifiés à partir de leur liste de paquets approuvés sans avoir à soumettre un ticket à leurs équipes informatiques.

Cette fonction est conçue pour offrir aux *Utilisateurs* le sentiment de liberté et d'autonomie qu'ils craignent de perdre dans les environnements gérés, tout en permettant aux RSSI d'appliquer des règles de sécurité strictes à l'aide d'une méthode telle que SRP (Software Restriction Policies), également connue sous le nom de *Applocker*.

- **WAPT WUA**.

  WAPT permet de gérer les mises à jour de Windows sur vos terminaux Windows.

  Le WAPT WUA est conçu pour fonctionner immédiatement, ménager votre stockage et préserver votre bande passante pour vos besoins de production.

- **Rapports avancés pour les équipes de l'entreprise**.

  Ces rapports complètent les rapports opérationnels déjà disponibles dans la console WAPT ; les rapports aident les opérateurs WAPT à démontrer leur efficacité avec WAPT pour assurer un plus grand niveau de sécurité et de conformité pour leurs réseaux, systèmes, logiciels et applications.

- **Configuration dynamique du dépôt**.

  À partir de WAPT 1.8, la réplication de référentiel peut être activée en utilisant un agent WAPT installé sur une machine existante, une appliance dédiée ou une machine virtuelle.

  Le rôle de réplication est déployé par le biais d'un paquet WAPT qui active le serveur web **`Nginx`** et configure la planification, les types de paquets, la synchronisation des paquets, et bien plus encore.

  Cette fonction permet aux agents WAPT de trouver dynamiquement le dépôt WAPT disponible le plus proche à partir d'une liste de règles stockées sur le serveur WAPT.

- **Intégration avec GLPI**

  GLPI est une solution populaire ITSM pour la gestion des tickets, des incidents et des actifs.

  WAPT peut maintenant envoyer de manière optionnelle un ensemble minimum d'informations utiles à un serveur GLPI.

6.5 Cas d'utilisation ciblés de WAPT Enterprise

La version Entreprise de WAPT est particulièrement recommandée pour les organisations:

- Qui gèrent de grandes bases installées de dispositifs (généralement plus de 300 unités).
- Qui sont répartis géographiquement avec de nombreuses filiales ou sites de production.
- Qui exigent une forte traçabilité des actions effectuées sur la base installée de dispositifs pour des raisons d'audit ou de sécurité.
- Qui accordent de l'importance à des solutions sécurisées et éprouvées dans leur recherches IT.

6.6 Description des services disponibles avec un contrat WAPT Enterprise

6.6.1 Accès aux futures améliorations de WAPT Enterprise

En souscrivant à un contrat WAPT **Enterprise** et en maintenant votre abonnement valide, vous bénéficiez des améliorations futures apportées au cœur de WAPT et vous bénéficiez automatiquement de toutes les améliorations futures de la version WAPT **Enterprise**.

L'expiration de votre abonnement fera automatiquement basculer votre instance WAPT vers sa version **Discovery** correspondante. Les fonctions avancées disponibles uniquement dans la version **Enterprise** ne seront plus accessibles et aucune action autre que la suppression d'hôtes à partir de la console ne sera autorisée tant que le nombre d'hôtes ne sera pas passé en dessous de 300.

6.6.2 Assistance téléphonique directe pour votre utilisation quotidienne de WAPT

Lorsque votre abonnement **dépasse un certain volume**, Tranquil IT, le créateur de WAPT, vous offre un accès privilégié à son équipe d'experts et de développeurs WAPT.

Nous vous donnons accès à une hot-line téléphonique dédiée avec une réponse directe pour satisfaire vos besoins d'assistance en **anglais** et **français**.

Nous nous engageons à vous fournir rapidement des réponses fiables et pertinentes sur le périmètre souscrit.

En souscrivant ou en renouvelant votre contrat WAPT **Enterprise**, vous recevrez une notification indiquant les modalités pratiques d'accès à notre support.

Attention: Le support ne concerne que l'utilisation dans votre Organisation du logiciel WAPT **Enterprise**, un support supplémentaire pour l'adaptation, la personnalisation, le débogage ou la création de paquets personnalisés WAPT peut être obtenu avec des tickets de support prépayés.

Jusqu'à trois personnes de votre *Organisation* peuvent communiquer avec notre support direct.

Note: Pour plus d'informations, contactez l'équipe commerciale de Tranquil IT.

6.6.3 Prix et accès préférentiel à la formation WAPT

Vous pouvez choisir de former votre équipe informatique sur n'importe quelle particularité de WAPT.

Note: Pour plus d'informations, contactez l'équipe commerciale de Tranquil IT.

CHAPTER 7

Quickstart - Installing WAPT Server

7.1 To read beforehand

- The quickstart guide install WAPT Server on a Windows Server. Installing WAPT on a Linux server is the recommended method, unless you are trialing WAPT and you are not familiar with Linux.
- The WAPT Server **MUST NOT** be installed on a computer with services already listening on port 443 (example WSUS with IIS). Port 443 is used by the WAPT Server and MUST be available to WAPT only.
- The WAPT Server **will not run** on a x86 version of Windows. It run only on a fresh Windows version currently supported by Microsoft. The server component of WAPT works just as well on a win10 client VM or a physical host as it does on a Windows server version.
- The installation of the WAPT Server **MUST** be done using a Local Administrator account on the host and **NOT a Domain Administrator** account.
- `Nginx` is the **ONLY** supported web server with WAPT. **Apache or IIS (with or without WSUS) are NOT supported in WAPT**.
- In case of problems when installing WAPT, visit the *Frequently Asked Questions*.

7.2 Installing WAPT server

- Download and execute as administrator `waptserversetup.exe`.
- Choose the language for the WAPT installer and click on *OK* to go on to the next step.
- Accept the licence terms and click on *Next* to go to next step.
- Choose additional configuration tasks (leave the default if not sure).
- Choose the password for the WAPT Server and click on the *Install* to launch the installation, wait for the installation to complete.

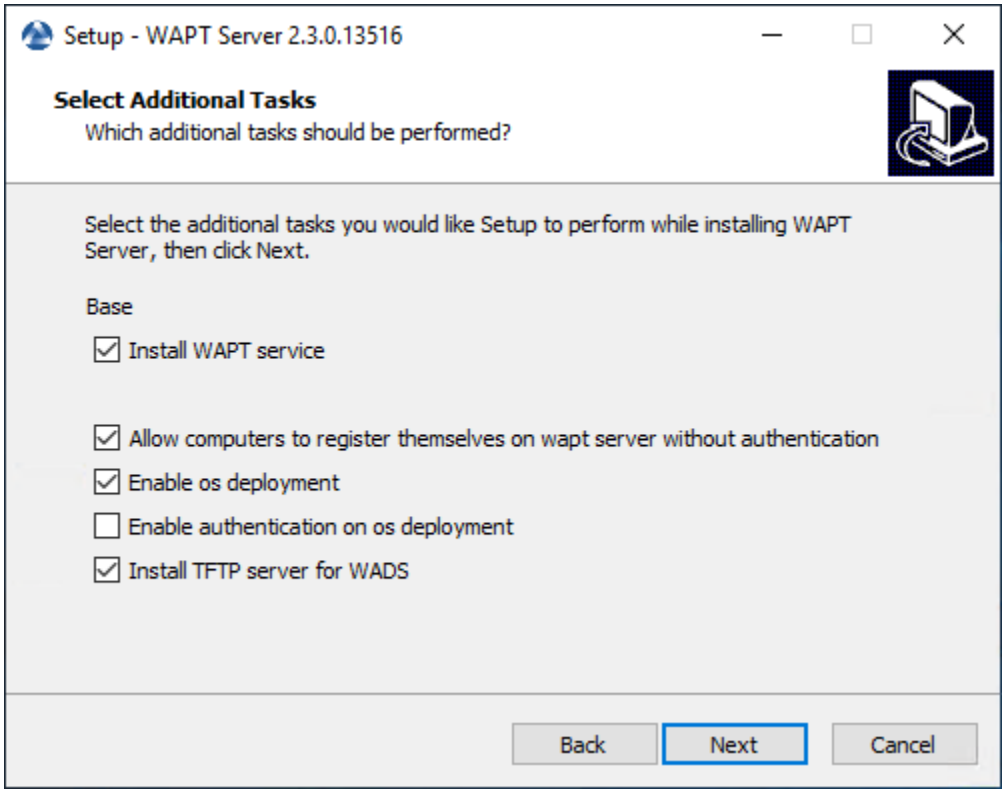
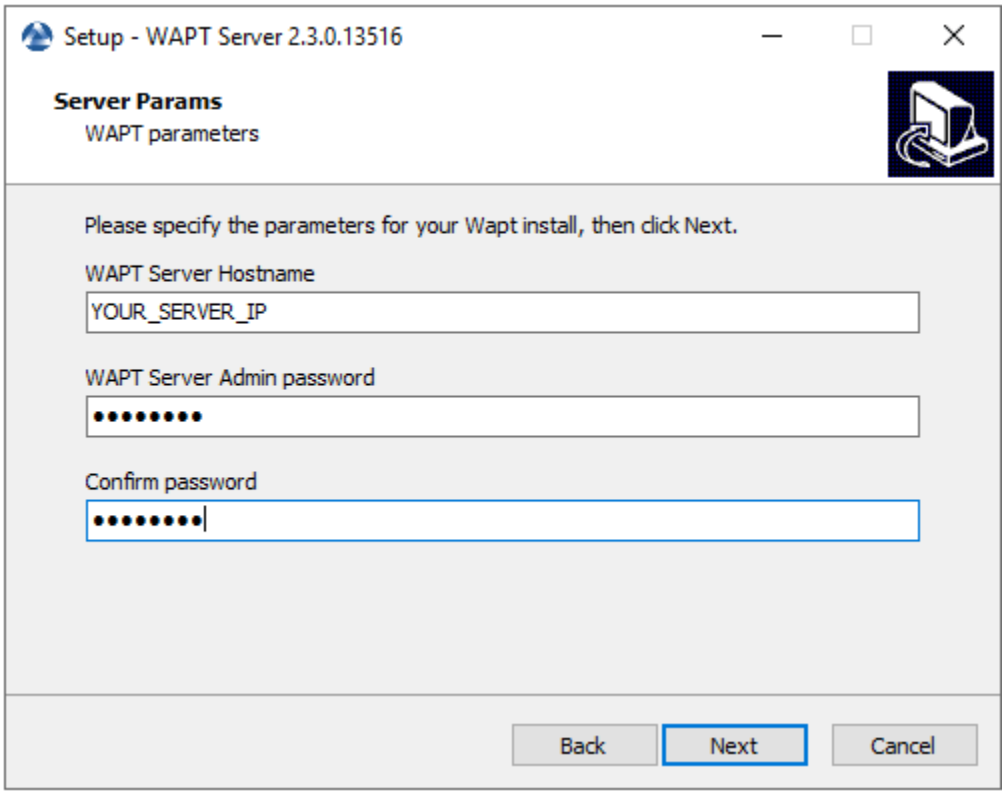


Figure1: Choosing the installer options for deploying the WAPT Server



- Click on *Finish* to close the window.

Attention:

- **For security, do not run the WAPT Console or your WAPT package development tool on the WAPT Server.**
- The WAPT Server on Windows **includes the WAPT Agent**. It is not necessary to install the WAPT Agent to manage the WAPT Server on Windows.

Your WAPT Server is now ready. You may now go to the documentation on *Quickstart - Installing the WAPT management Console*.

CHAPTER 8

Quickstart - Installing WAPT Console

The WAPT Server having been successfully installed, now we will install the WAPT Console.

- Managing WAPT is done mainly via the WAPT Console installed on the *Administrator*'s workstation.
- It is recommended that the Administrator's computer be joined to the *Organization*'s Active Directory.
- The host name of the Administrator's workstation **MUST NOT be longer than 15 characters.** This is a limit of *sAMAccountName* attribute in Active Directory.
- **The Administrator's computer will become critical for WAPT administration and WAPT package testing.**

> **Avertissement:** The WAPT Console **MUST NOT** be installed on your Windows based WAPT Server.
>
> The WAPT Console **MUST** be installed on the workstation from which you manage your network.

To download the `waptsetup.exe` file, point your web browser to your waptserver url https://YOUR_SERVER_IP, then click on the *WAPTSetup* link on the right-hand side of the WAPT Server web page. The WAPT Server home page only provides basic server status information and the download link for the WAPT Console.

8.1 Installing the WAPT Setup on the Administrator's computer

- Start the executable installer as *Local Administrator* on the *Administrator*'s workstation.
- Choose the language for the WAPT installer and click on *OK* to go on to the next step.
- Accept the licence terms and click on *Next* to go to next step.
- Click on *Next* to leave the default WAPT installation folder.
- Click on *Next* with just « Install WAPT service » checked.
- Set up the WAPT Server URL. Can be IP address or DNS Name. Leave the check options as it is.

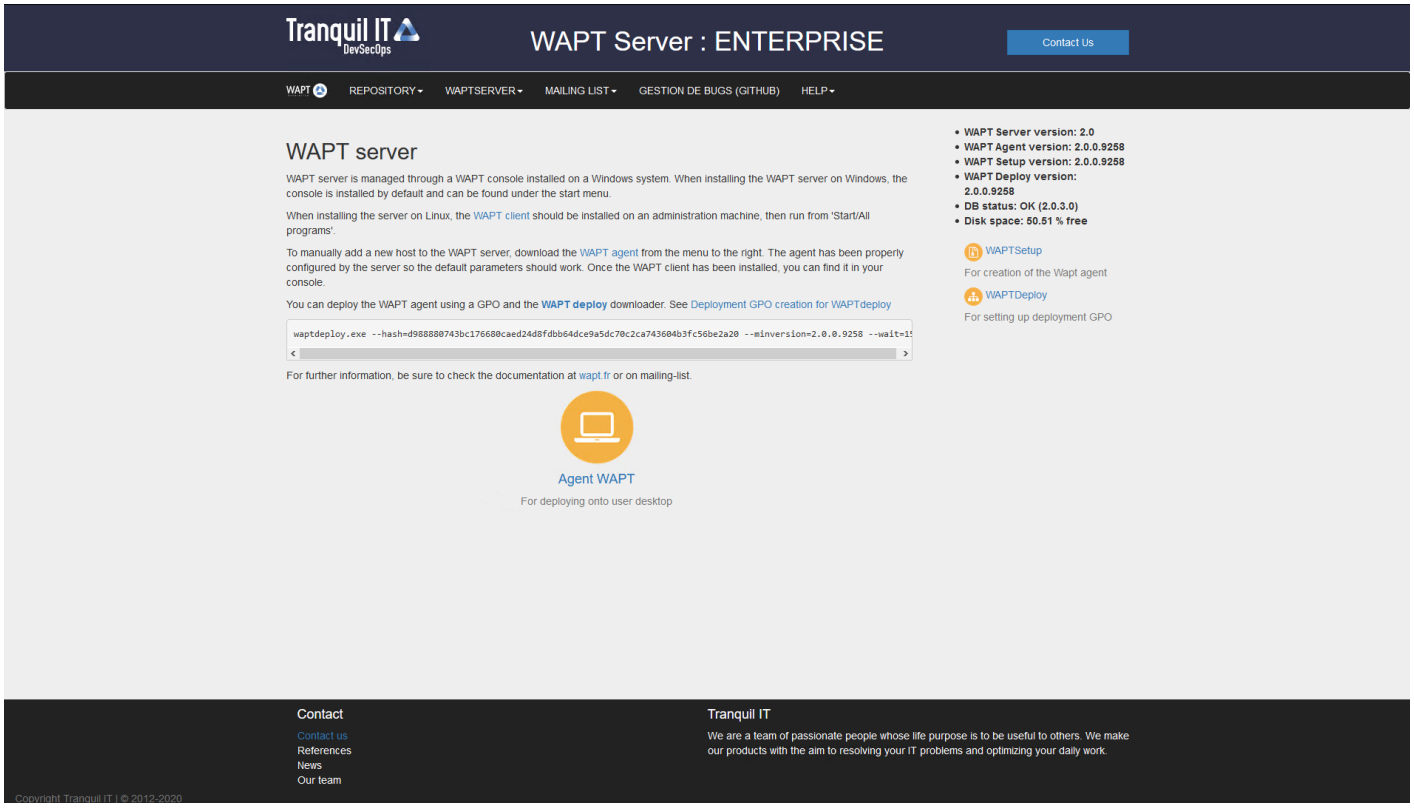


Figure1: The WAPT Server interface in a web browser

- Check *Static WAPT Informations* and set:
  - WAPT repository URL: http://YOUR_SERVER_IP/wapt.
  - WAPT Server URL: https://YOUR_SERVER_IP.

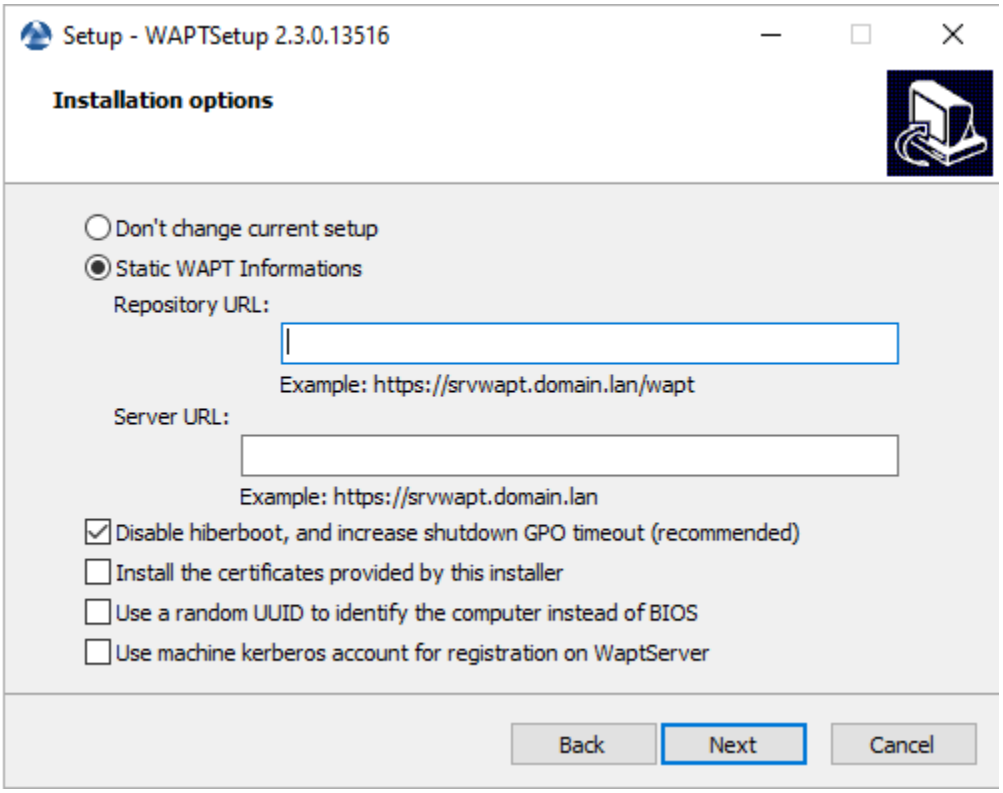


Figure2: Choosing the WAPT repository and the WAPT Server

- Choose the WAPT repository and the WAPT Server; click *Next*.
- Get a summary of the WAPT Console installation and click *Install* to launch the installation, wait for the installation to complete, then click on *Finish* (leave default options).

8.1.1 Starting the WAPT Console

- Launch the WAPT Console:
  - By looking for the binary.

    ```
    C:\Program Files (x86)\wapt\waptconsole.exe
    ```

  - Or using the *Start* Menu.
- Log into the WAPT Console with the *SuperAdmin* login and password.

On first start, you **MUST** start the WAPT Console with elevated privileges. *Right-click on the WAPT Console binary → Start as Local Administrator*.

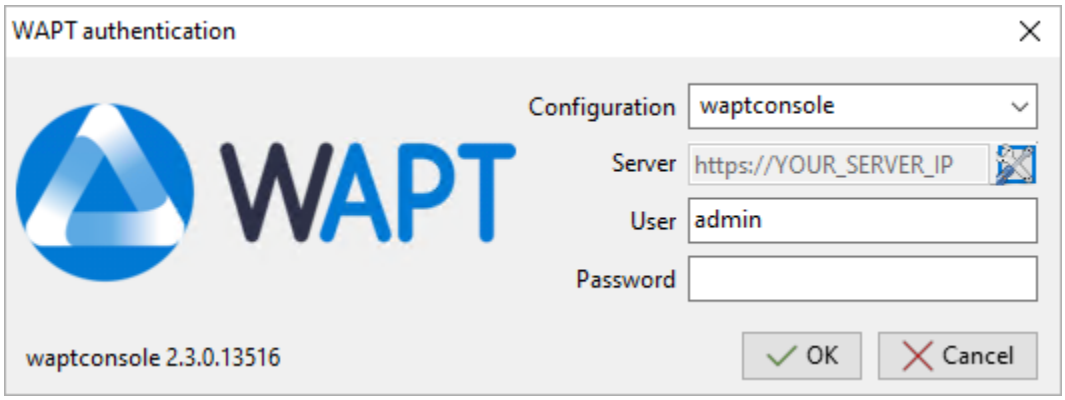


Figure3: The WAPT Console authentication window

8.2 Generating the Administrator's certificate for signing WAPT packages

- A message will appear indicating that no personal certificate has been defined.

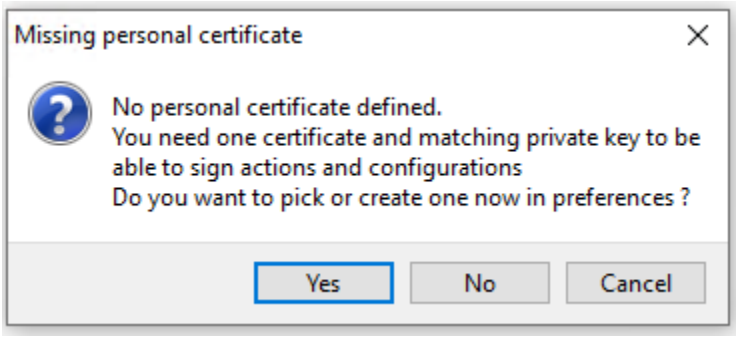


Figure4: WAPT personal certificate not found in the WAPT Console

- Select *Yes*
- Click on *New private key and certicate* and see *create your certificate*.
- In the example, the name of the public certificate signed with the private key is `wapt-private.crt`. This certificate is used to validate the signature of packages before installation. If the public certificate used on the WAPT Console is not derived from the private key used for generating the WAPT Agents, the WAPT Console will not see the WAPT Agents and you will not be able to perform any action on any WAPT Agent.
- In the example, the name of the private key is `wapt-private.pem`. It is located by default in the `C:\private` folder of the *Administrator* workstation and is password protected. it will be used along with the certificate to sign packages before uploading them onto the WAPT repository.

> **Danger:** The `wapt-private.pem` file is **fundamental for security**. It **MUST** be stored in a safe place and correctly protected. The `wapt-private.pem` file **MUST NOT** be stored on the WAPT Server.

- Fill the informations to create a self-signed certificate.
- Click on *OK* to go on to the next step.

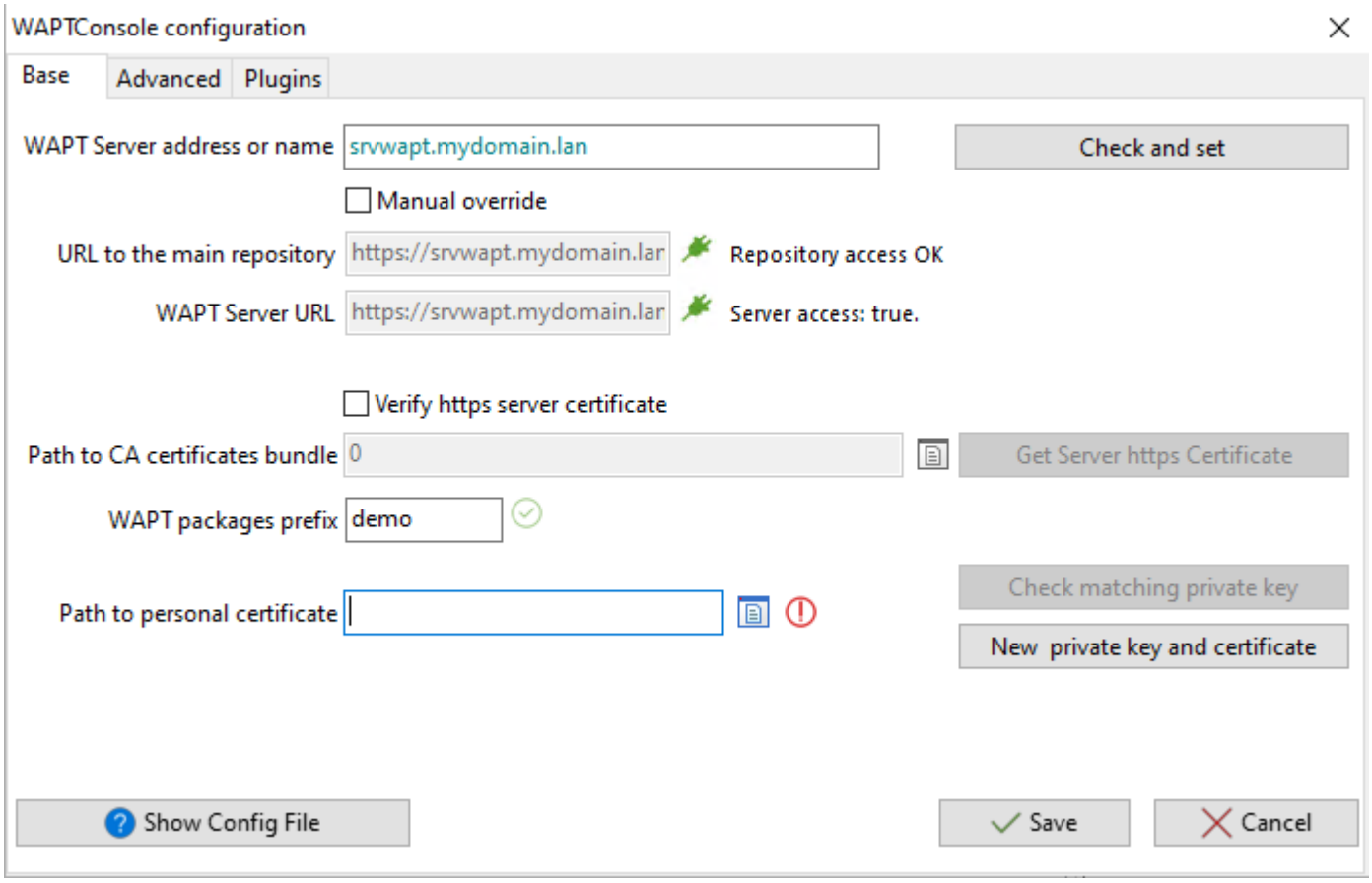
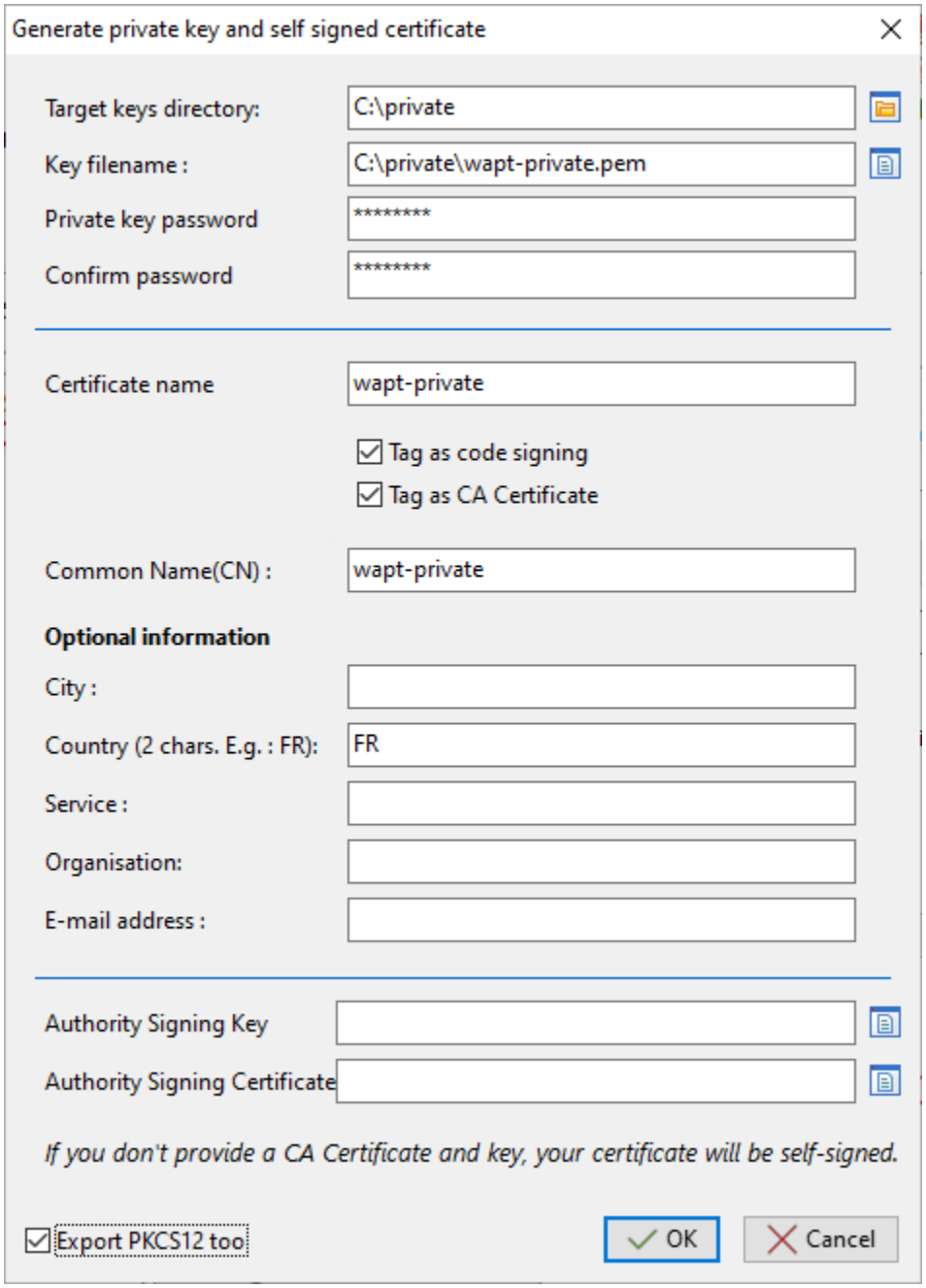


Figure5: Window for the basic configuration of the WAPT Console


Generate private key and self signed certificate
Target keys directory:
C:\private
Key filename :
C:\private\wapt-private.pem
Private key password
****
Confirm password
****
Certificate name
wapt-private
Tag as code signing
Tag as CA Certificate
Common Name(CN) :
wapt-private
Optional information
City :
Country (2 chars. E.g. : FR):
FR
Service :
Organisation:
E-mail address :
Authority Signing Key
Authority Signing Certificate
If you don't provide a CA Certificate and key, your certificate will be self-signed.
Export PKCS12 too
OK
Cancel

- Click on *Yes* to copy the newly generated certificate in the folder `C:\Program Files (x86)\wapt\ssl` on Windows or `/opt/wapt/ssl` on Linux or macOS. This certificate will be picked up during the compilation of the WAPT Agent and deployed on the client computers.

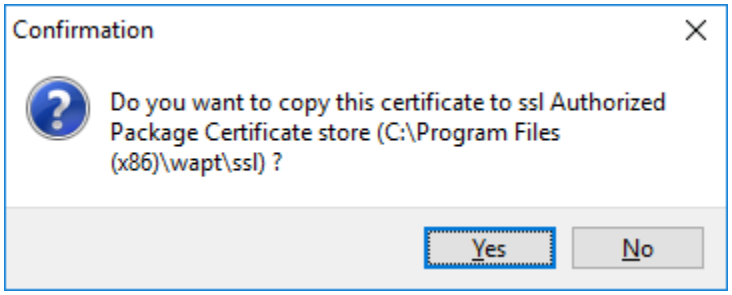


Figure6: Dialog box requesting confirmation of the copy of the certificate in the ssl folder in the WAPT Console

8.3 Packet prefix definition

- A message will appear indicating that no package prefix has been defined.

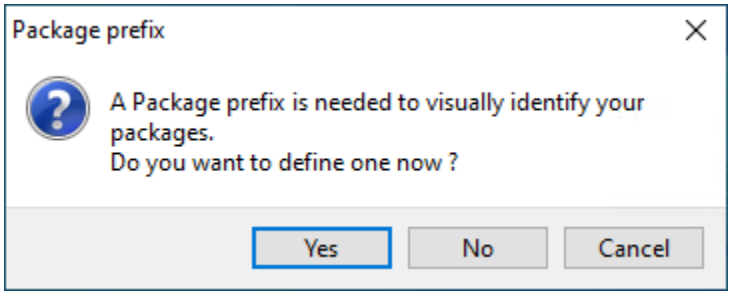


Figure7: Dialog box informing that no prefix has been set in the WAPT configuration

- Select *Yes*
- Set your packages prefix on *WAPT packages prefix*

8.4 Activating a WAPT licence

- A message will appear indicating that no licence has been found, click on yes to activate a licence. To activate the licence, use the `licence.lic` file provided by our sales department.

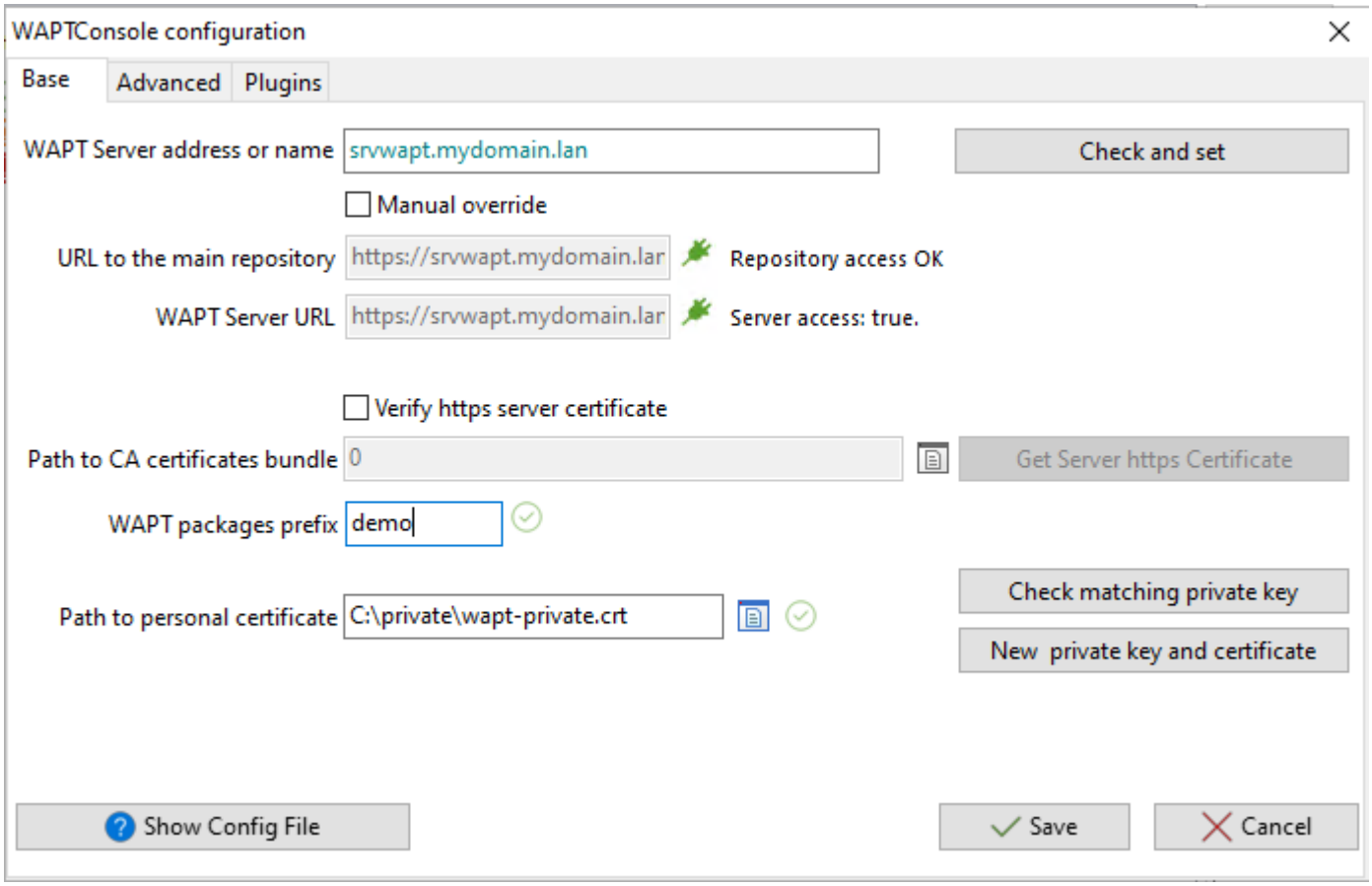


Figure8: Window for the basic configuration of the WAPT Console

Figure9: Window listing no subscribed WAPT licences in the WAPT Console

8.5 Creating the WAPT Agent

Note: A message may appear indicating that your WAPT Agent version is obsolete or not yet present.

If the *quickstart - administrator's certicate* existing, it's possible to *quickstart - generating new WAPT Agent* by clicking on *Yes*.

Also click on *No* and generate the *administrator's certicate*.

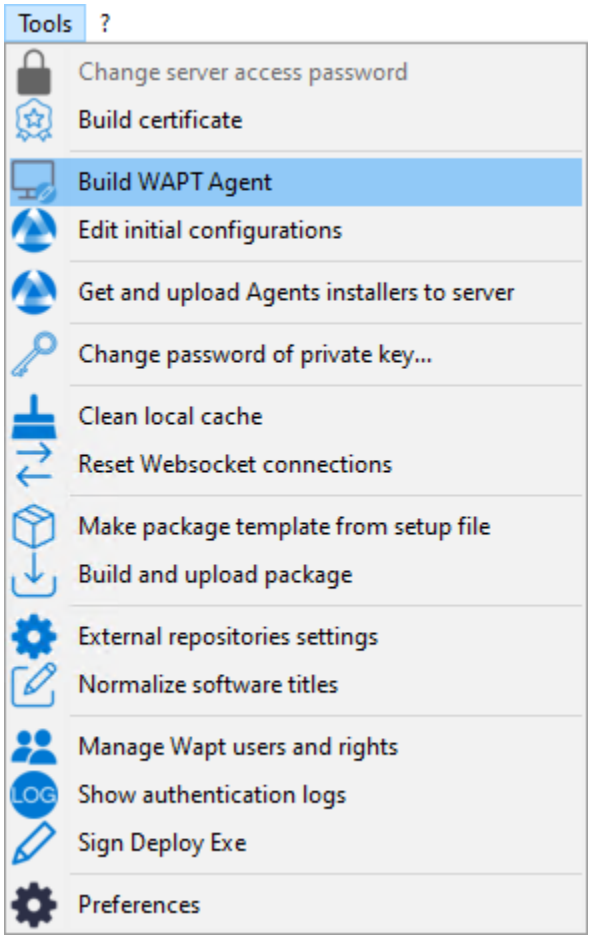


Figure10: Generating the WAPT Agent from the WAPT Console

- Fill in the informations that are necessary for the installer.
- Provide the password for unlocking the private key.

Once the WAPT Agent installer has finished building, a confirmation dialog pops up indicating that the **`waptagent`** binary has been successfully uploaded to https://YOUR_SERVER_IP/wapt/.

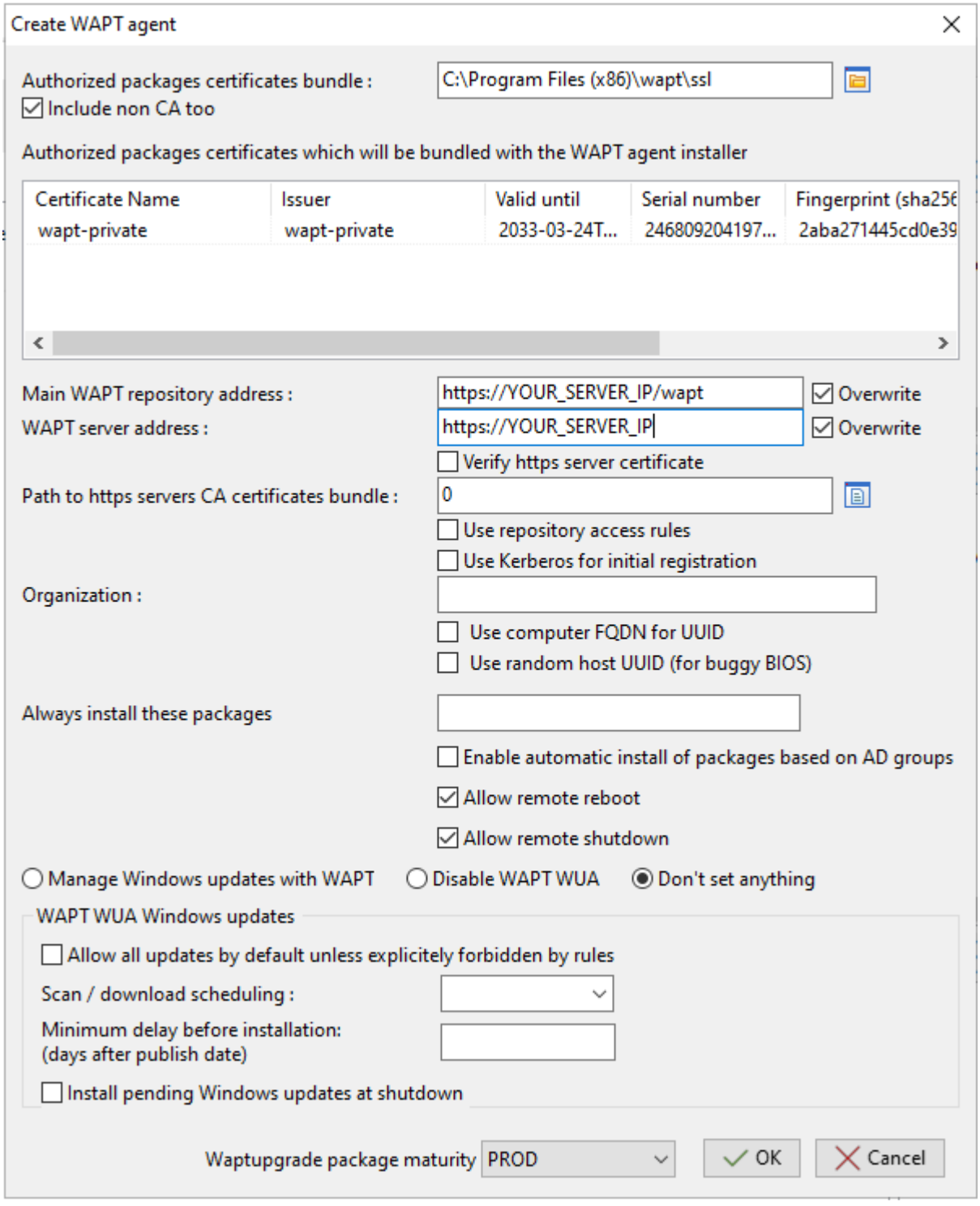


Figure11: Filling in the informations on your Organization

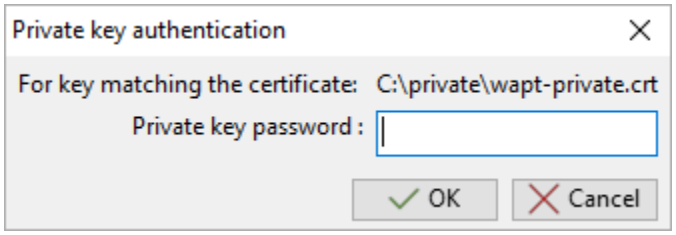


Figure12: Providing the password for unlocking the private key

CHAPTER 9

Quickstart - Installing WAPT Agent

9.1 Installation of WAPT Agent

Manually installing the WAPT Agent requires *Local Administrator* rights on the computer.

- Download the WAPT Agent from your WAPT Server then launch the installer. The **`waptagent.exe`** installer is available at WAPT serveur web home page. The direct download link is for example: https://YOUR_SERVER_IP/wapt/waptagent.exe.
- Choose the language for the WAPT installer and click on *OK* to go on to the next step.
- Accept the licence terms and click on *Next* to go to next step.
- Just click on next until the install button

CHAPTER 10

Prérequis d'installation

10.1 Prérequis d'installation

10.1.1 Conventions de dénomination

Vous devez prendre en considération quelques points de sécurité afin de tirer tous les avantages possibles du WAPT:

- Si vous êtes familier avec Linux, nous vous conseillons d'installer le serveur WAPT directement sur CentOS en suivant les recommandations de sécurité de l'" *ANSSI* ou les recommandations de l'agence de cyberdéfense de votre état.
- Bien que le serveur WAPT ne soit pas conçu pour être un actif sensible, nous recommandons qu'il soit installé sur une **machine dédiée** (physique ou virtuelle).

Attention: Dans toutes les étapes de la documentation, vous n'utiliserez aucun accent ni caractères spéciaux pour:

- le login des utilisateurs;
- le chemin de la clé privée et du certificat;
- le CN (Common Name);
- le chemin d'installation de WAPT;
- les noms de groupe;
- le nom des hôtes ou le nom du serveur;
- le chemin vers le répertoire `C:\waptdev`.

10.1.2 Préconisations matérielles

Le serveur WAPT peut être installé soit sur un serveur virtuel, soit sur un serveur physique.

Table1: Recommandations de RAM et de CPU optimales pour le serveur WAPT

Taille de parc	CPU	RAM	Optimisation serveur à appliquer
De 0 a 300 postes	2 CPU	2024 Mio	Non
De 300 a 1000 postes	4 CPU	4096 Mio	Oui
De 1000 a 3000 postes	8 CPU	8192 Mio	Oui
A partir de 3000 postes et plus	16 CPU	16384 Mio	Oui

- Un minimum de 10 Go d'espace libre est nécessaire pour le système, la base de données et les fichiers journaux.
- Un minimum de 10 Go d'espace libre est nécessaire pour le système, la base de données et les fichiers de journalisation. **Pour de meilleures performances, Tranquil IT recommande que la base de données soit stockée sur des supports rapides, tels que des disques SSD ou des SSD sur PCIe**.
- L'exigence globale en matière de disque dépendra du nombre et de la taille de vos paquets WAPT (logiciels) que vous stockerez sur votre dépôt principal, 30 Go étant un bon début. Il n'est pas strictement nécessaire de stocker les paquets WAPT sur des disques rapides.
- Enfin, nous avons connaissance d'utilisateurs disposant de serveurs équipés de multiples interfaces réseau 10Gbps déployant à pleine vitesse des paquets de mise à jour massifs de Katia, National Instruments et Solidworks sur leur LAN (Local Area Network).

10.1.3 Préconisations logicielles

Système d'exploitation

Le serveur WAPT est disponible sur Linux et Windows:

- Pour Linux, **Debian 11**, **Red Hat 7 / 8 et dérivés**, **Ubuntu server LTS 20.04** la version 64 bit est supporté. Il n'est pas obligatoire d'utiliser une distribution Linux serveur, mais utilisez une distribution **non graphique**.

Note: SELINUX est supporté mais pas obligatoire.

- Pour Windows, le serveur WAPT peut être installé sur **Windows Server** version 64 bits supportée par Microsoft (Win2012r2, Win2k16 ou Win2k19). Selon votre besoin, il peut également être installé sur une version récente de Win10 Pro/Ent (20H2 ou plus).

Attention:

- Le serveur WAPT ne fonctionnera que sur un système basé sur **64bit**.
- Installez le serveur **sans** interface graphique.
- Systemd doit être activé

Ouverture de ports

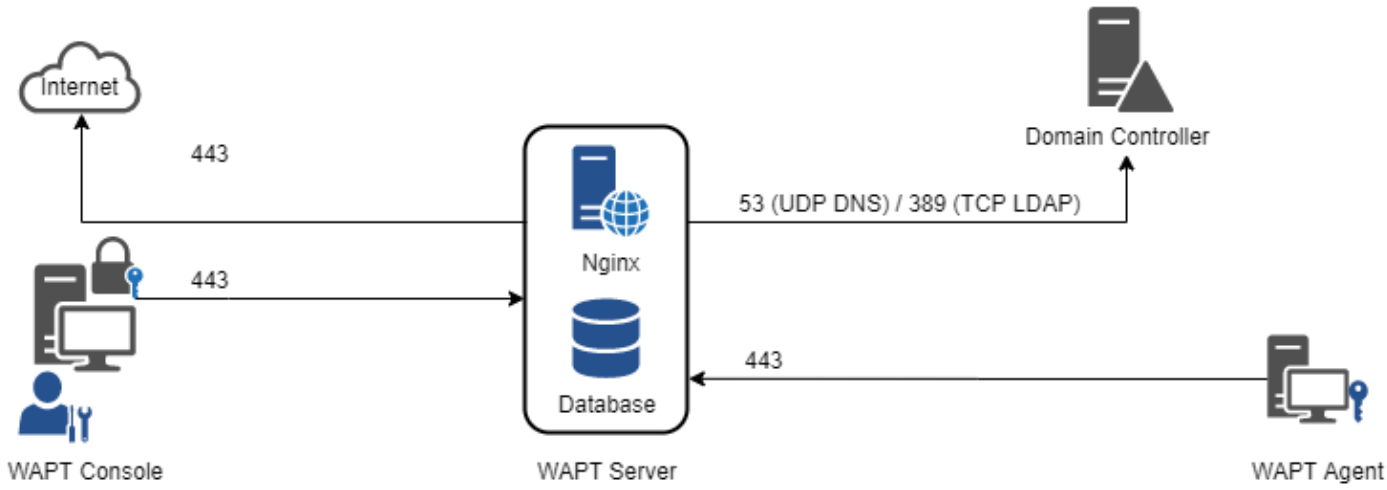


Figure1: Diagramme des flux de données de WAPT

Seuls les ports **80** et **443** doivent être ouverts pour les connexions entrantes car le framework WAPT fonctionne avec des websockets initiés par les Agents WAPT.

Entrant

Table2: Ports entrants à ouvrir pour que le WAPT fonctionne

Protocole	Numéro de port	Source	Destination	Description
TCP	**80**	Tous les agents WAPT	Serveur WAPT	Connexion Websocket (non-sécurisé) pour télécharger les paquets et les KB.
TCP	**443**	Tous les agents WAPT	Serveur WAPT	Connexion Websocket pour télécharger les paquets et les KB.
UDP	**69** Note : tftp utilise des ports éphémères/dynamiques pour le transport des données. Si vous avez un pare-feu entre le serveur et les ordinateurs, assurez-vous d'avoir activé le support pour tftp conntrack.	Si vous utilisez *Déploiement WADS* le port TFTP (69) doit être ouvert.	Serveur WAPT	Pour télécharger la première étape des fichiers de démarrage de l'OS avant que le protocole HTTP ne soit disponible.

Sortant

Table3: Ports sortants à ouvrir pour que le WAPT fonctionne

Protocole	Numéro de port	Source	Destination	Description
TCP	**80**	Serveur WAPT	Internet	Connexion Websocket (non-sécurisé) pour télécharger les paquets, `wsusscn2.cab` et les KB.
TCP	**80**	Serveur WAPT	Dépôt Linux (pour les serveurs Linux)	Téléchargement des packages WAPT en utilisant le protocole HTTP (non sécurisé).
TCP	**443**	Serveur WAPT	Dépôt Linux (pour les serveurs Linux)	Téléchargement des packages WAPT à l'aide de HTTPS (sécurisé).
TCP	**53**	Serveur WAPT	Contrôleur de domaine ou SERVEUR DNS (Domain Name Service)	Résolution de noms de domaine.
TCP	**389**	Serveur WAPT	Contrôleur de domaine ou serveur LDAP (Lightweight Directory Access Protocol)	Authentification LDAP pour authentifier les utilisateurs avec la Console WAPT ou le Self-service WAPT.
TCP	**636**	Serveur WAPT	Contrôleur de domaine ou serveur LDAP	Authentification LDAP.
UDP	**123**	Serveur WAPT	Contrôleur de domaine ou serveur NTP (Network Time Protocol)	NTP pour garder le temps synchronisé et kerberos fonctionnant correctement.

10.2 Conseils avant l'installation

10.2.1 Configurer les DNS de l'Organisation pour WAPT

Note: La configuration DNS n'est pas obligatoire, elle est fortement recommandée.

Afin de faciliter la gestion de votre installation WAPT, il est fortement recommandé de configurer le serveur *DNS* pour inclure le champ A ou le champ CNAME comme ci-dessous :

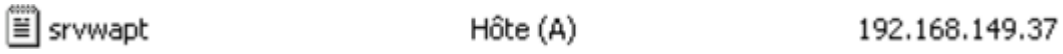
- *srvwapt.mydomain.lan*.
- *wapt.mydomain.lan*.

Remplacer *mydomain.lan* par le suffixe *DNS* utilisé sur votre réseau.

Ces champs seront utilisés par les agents WAPT pour trouver le serveur WAPT ou un dépôt secondaire WAPT de proximité sur le réseau.

10.2.2 Configurer les champs DNS avec les « Outils d'administration de serveur distant » Microsoft (RSAT).

- Le champ *A* pointe vers l'adresse IP du serveur WAPT.



Vous pouvez maintenant installer votre serveur WAPT sur l'OS de votre choix:

- *Installer le serveur WAPT sur GNU / Linux Debian*.
- *Installer le serveur WAPT sur CentOS / RedHat*.
- *Installer le serveur WAPT sur Windows*.

CHAPTER 11

Installer le serveur WAPT sur Debian

11.1 Configuration du serveur GNU/Linux Debian

Afin d'installer un Debian Linux 11 Bullseye ou Ubuntu Focal LTS frais (physique ou virtuel).

Avertissement:

- Installer la version 64bits.
- Installez le serveur sans interface graphique.
- Systemd doit être activé

Attention: La procédure de mise à jour est différente de l'installation. Pour une mise à jour, rendez-vous sur *la documentation pour mettre à jour le serveur WAPT*.

11.1.1 Configurer les paramètres réseau

Les différents paramètres présentés ci-dessous ne sont pas spécifiques à WAPT, vous pouvez les adapter en fonction de votre environnement.

Modifiez les fichiers suivants afin d'obtenir une stratégie de nommage (*FQDN*) et d'adressage réseau appropriée.

Dans l'exemple suivant :

- le nom *FQDN* est *srvwapt.mydomain.lan* ;
- le nom court du serveur WAPT est *srvwapt* ;
- le suffixe *DNS* est *mydomain.lan* ;

- l'adresse IP est *10.0.0.10/24* ;

11.1.2 Configurer le nom du serveur WAPT

Indication: Le nom du serveur WAPT ne doit pas dépasser **15 caractères** (limite liée au sAMAccountName dans Active Directory).

Le nom du serveur doit être un nom FQDN (Fully Qualified Domain Name), c'est à dire à la fois le nom de machine et le suffixe DNS.

- Modifier le fichier `/etc/hostname` et y renseigner le nom *FQDN* du serveur.

```
# /etc/hostname of the WAPT Server
srvwapt.mydomain.lan
```

- Configurez le fichier `/etc/hosts`, assurez-vous de mettre à la fois le *FQDN* et le nom court du serveur.

```
# /etc/hosts of the WAPT Server
127.0.0.1   localhost localhost.localdomain localhost4 localhost4.localdomain4
::1         localhost localhost.localdomain localhost6 localhost6.localdomain6
10.0.0.10   srvwapt.mydomain.lan srvwapt
```

Indication:

- Sur la ligne définissant l'adresse IP du serveur DNS, veillez à avoir l'IP du serveur (pas 127.0.0.1), puis le *FQDN*, puis le nom court.
- Ne changez pas la ligne avec *localhost*.

11.1.3 Configurer l'adresse IP du serveur WAPT

- Configurez l'adresse IP du serveur WAPT dans `/etc/network/interfaces`.

```
# /etc/network/interfaces of the WAPT Server
auto eth0
iface eth0 inet static
  address 10.0.0.10
  netmask 255.255.255.0
  gateway 10.0.0.254
```

- Appliquez la configuration réseau en redémarrant la machine avec un `reboot`.

```
reboot
```

- Si cela n'a pas déjà été fait, créez l'entrée *DNS* pour le serveur WAPT dans l'Active Directory de l'" *Organisation*.
- Après le redémarrage, configurez la langue du système en anglais afin d'avoir des journaux non localisés pour faciliter la recherche des erreurs courantes.

```
apt install locales-all -y
localectl set-locale LANG=en_US.UTF-8
localectl status
```

- Vérifier si la machine est correctement synchronisée avec le serveur NTP. Si elle n'est pas synchronisée, se référer à la documentation du système d'exploitation pour configurer **timedatectl**.

```
timedatectl status
```

- Mettre à jour et à niveau le système d'exploitation et s'assurer que le paquet d'Autorités de Certification par défaut de Debian est installé.

```
apt update && apt upgrade
apt install ca-certificates -y
```

- Redémarrer le serveur.

```
reboot
```

Le serveur est maintenant prêt.

L'installation de la partie serveur de WAPT se décompose en plusieurs étapes:

- Configurer les dépôts.
- Installer les paquets Linux complémentaires.
- Installation et provisionnement de la base de données PostgreSQL.
- Post-configuration du serveur WAPT.

Note: Les paquets du serveur WAPT et le dépôt sont signés par Tranquil IT et il est nécessaire d'obtenir la clé publique *gpg* ci-dessous afin d'éviter les messages d'avertissement pendant l'installation.

11.2 Installer les paquets du Serveur WAPT

- Mettre à jour la source APT, récupérer la clé *.gpg* de Tranquil IT, puis ajouter le dépôt de Tranquil IT.

```
apt install apt-transport-https lsb-release gnupg wget -y
wget -O - https://wapt.tranquil.it/$(lsb_release -is)/tiswapt-pub.gpg | apt-key add -
echo "deb https://wapt.tranquil.it/$(lsb_release -is)/wapt-2.3/ $(lsb_release -c -s) main" > /
↪etc/apt/sources.list.d/wapt.list
```

- Installer les paquets du Serveur WAPT.

```
export DEBIAN_FRONTEND=noninteractive
apt update
apt install tis-waptserver tis-waptsetup -y
unset DEBIAN_FRONTEND
```

11.3 Post-configuration

CHAPTER 12

Script de post-configuration du serveur WAPT

Attention: Pour que le post-configuration fonctionne correctement, vous devez d'abord avoir un *hostname* pour le serveur WAPT. Pour vérifier, utilisez la commande **`echo $(hostname)`** qui doit retourner l'adresse DNS qui sera utilisé par les agents WAPT sur les ordinateurs clients.

Le script de post-configuration réécrit la configuration de nginx.

Ce script de post-configuration doit être exécuté en tant que **root**.

- Lancez le script.

```
/opt/wapt/waptserver/scripts/postconf.sh
```

- Cliquez sur *Oui* pour lancer le script de post-configuration.

```
do you want to launch post configuration tool?

           < yes >          < no >
```

- Choisissez un mot de passe (si ce n'est pas déjà défini) pour le compte *SuperAdmin* du serveur WAPT (longueur minimale de 10 caractères).

```
Please enter the wapt server password (min. 10 characters)

*****************

               < OK >          < Cancel >
```

- Confirmer le mot de passe.

```
Please enter the server password again:

******************

                < OK >          < Cancel >
```

- Choisissez le mode d'authentification pour l'enregistrement initial des agents WAPT:
  - Le choix #1 permet d'enregistrer les ordinateurs sans authentification. Le serveur WAPT enregistre tous les ordinateurs qui demandent à être enregistré.
  - Le choix #2 active l'enregistrement initial basé sur kerberos (vous pouvez l'activer plus tard).
  - Le choix #3 n'active pas le mécanisme d'authentification kerberos pour l'enregistrement initial des machines équipées de WAPT. Le serveur WAPT va demander un identifiant et mot de passe pour chaque machine qui s'enregistre.

```
WaptAgent Authentication type?

--------------------------------------------------------------------------
(x) 1 Allow unauthenticated registration
( ) 2 Enable kerberos authentication required for machines registration.
        Registration will ask for password if kerberos not available
( ) 3 Disable kerberos but registration require strong authentication
--------------------------------------------------------------------------
                                        < OK >          < Cancel >
```

- Si vous souhaitez utiliser WAPT pour le déploiement de systèmes d'exploitation, sélectionnez *Yes*.

```
Do you want to activate os deployment?

        < Yes >        < No >
```

- Si vous avez répondu *Yes* pour activer la fonctionnalité de déploiement de systèmes d'exploitation avec WAPT, le script de post-configuration vous demandera si vous souhaitez utiliser une authentification sécurisée afin de déployer l'image du système d'exploitation. Il vous demandera alors un utilisateur / mot de passe lorsque vous essayerez de déployer le système d'exploitation.

```
Would you like to activate secure authentication on wads?

        < Yes >        < No >
```

- Toujours en ce qui concerne la fonction WADS, si vous avez répondu *Yes* aux 2 dernières questions, vous aurez une dernière question :

```
Would you like to mention an IP subnet exempt from wads authentication

        < Yes >        < No >
```

Si vous répondez *Yes* à cette question, vous devrez donner un sous-réseau IP qui peut être une liste (exemple : 192.168.0.0/24, 192.168.1.0/24).

- Sélectionnez *OK* pour démarrer le serveur WAPT.

```
Press OK to start waptserver

            < OK >
```

- Sélectionnez *Oui* pour configurer Nginx.

```
Do you want to configure nginx?

       < Yes >        < No >
```

- Remplissez le *FQDN* du serveur WAPT.

```
FQDN for the WAPT Server (eg. wapt.example.com)

---------------------------------------------
wapt.mydomain.lan
---------------------------------------------

           < OK >          < Cancel >
```

- Sélectionner *OK* et un certificat auto-signé sera généré, cette étape peut prendre un certain temps.

```
Generating DH parameters, 2048 bit long safe prime, generator 2 This is going to take a long time
.....................................+.............................+...
```

Nginx est maintenant configuré, sélectionnez *OK* pour redémarrer **Nginx**:

```
The Nginx config is done.
We need to restart Nginx?

              < OK >
```

Le post-confoguration est maintenant terminé.

```
Postconfiguration completed.
Please connect to https://wapt.mydomain.lan/ to access the WAPT Server.

                                   < OK >
```

Liste des options du script de post-configuration:

Options	Description
`--force-https`	Configurer **Nginx** de sorte que *le port 80 soit redirigé de façon permanente vers le 443*

Le serveur est maintenant prêt. Vous pouvez consulter la documentation sur *l'installation de la console WAPT*.

CHAPTER 13

Installer le Serveur WAPT sur une distribution basée sur RedHat

13.1 Configurer le serveur CentOS / RedHat

Afin d'installer une nouvelle machine CentOS7 (virtuelle ou physique), veuillez vous référer à la documentation officielle de CentOS. Cette documentation est également valable pour Redhat7.

Avertissement:

- Installez le serveur sans interface graphique.

13.1.1 Configurer les paramètres réseau

Les différents paramètres présentés ci-dessous ne sont pas spécifiques à WAPT, vous pouvez les adapter en fonction de votre environnement.

Modifiez les fichiers suivants afin d'obtenir une stratégie de nommage (*FQDN*) et d'adressage réseau appropriée.

Dans l'exemple suivant :

- le nom *FQDN* est *srvwapt.mydomain.lan* ;
- le nom court du serveur WAPT est *srvwapt* ;
- le suffixe *DNS* est *mydomain.lan* ;
- l'adresse IP est *10.0.0.10/24* ;

13.1.2 Configurer le nom du serveur WAPT

Indication: Le nom court du serveur WAPT ne doit pas dépasser 15 caractères (limite liée au sAMAccountName dans Active Directory).

Le nom du serveur doit être un nom FQDN, c'est à dire à la fois le nom de machine et le suffixe DNS.

- Modifier le fichier `/etc/hostname` et y renseigner le nom *FQDN* du serveur.

```
# /etc/hostname of the WAPT Server
srvwapt.mydomain.lan
```

- Configurez le fichier `/etc/hosts`, assurez-vous de mettre à la fois le *FQDN* et le nom court du serveur.

```
# /etc/hosts of the waptserver
127.0.0.1   localhost localhost.localdomain localhost4 localhost4.localdomain4
::1         localhost localhost.localdomain localhost6 localhost6.localdomain6
10.0.0.10   srvwapt.mydomain.lan srvwapt
```

Indication:

- Sur la ligne définissant l'adresse IP du serveur DNS, veillez à avoir l'IP du serveur (pas 127.0.0.1), puis le *FQDN*, puis le nom court.
- Ne modifiez pas la ligne avec `localhost`.

13.1.3 Configurer l'adresse IP du serveur WAPT

- Modifiez le fichier `/etc/sysconfig/network-scripts/ifcfg-eth0` et définissez une adresse IP statique. Le nom du fichier peut être différent, comme `ifcfg-ens0` par exemple.

```
# /etc/sysconfig/network-scripts/ifcfg-eth0 of the WAPT Server
TYPE="Ethernet"
BOOTPROTO="static"
NAME="eth0"
ONBOOT="yes"
IPADDR=10.0.0.10
NETMASK=255.255.255.0
GATEWAY=10.0.0.254
DNS1=10.0.0.1
DNS2=10.0.0.2
```

- Appliquez la configuration réseau en redémarrant la machine avec un `reboot`.

```
reboot
```

- Si ce n'est pas déjà fait, *créer les entrées DNS pour le serveur WAPT* dans le *Organisation* Active Directory ou sur votre serveur DNS.

- Après le redémarrage, configurez la langue du système en anglais afin d'avoir des journaux non localisés pour faciliter la recherche des erreurs courantes.

```
localectl set-locale LANG=en_US.utf8
localectl status
```

- Vérifiez que l'horloge de la machine est à l'heure (avec NTP installé), et que SELinux et le pare-feu sont activés.

```
date
sestatus
systemctl status firewalld
```

- Vérifier si la machine est correctement synchronisée avec le serveur NTP. Si elle n'est pas synchronisée, se référer à la documentation du système d'exploitation pour configurer **timedatectl**.

```
timedatectl status
```

- Mettre à jour CentOS et configurez le dépôt EPEL (Extra Packages for Enterprise Linux).

```
yum update
yum install epel-release wget sudo -y
```

Le serveur WAPT est maintenant prêt.

Attention: La procédure de mise à jour est différente de l'installation. Pour une mise à jour, rendez-vous sur *la documentation pour mettre à jour le serveur WAPT*.

13.2 Installer des paquets complémentaires

Redhat 9 and derivatives

- Ajout du dépôt Tranquil'iT.

```
cat > /etc/yum.repos.d/wapt.repo <<EOF
[wapt]
name=WAPT Server Repo
baseurl=https://wapt.tranquil.it/redhat9/wapt-2.3/
enabled=1
gpgcheck=1
EOF
```

- Récupérer la clé `.gpg` et installer les paquets nécessaires.

```
wget -q -O /tmp/tranquil_it.gpg "https://wapt.tranquil.it/redhat9/RPM-GPG-KEY-TISWAPT-9"; rpm --
↪import /tmp/tranquil_it.gpg
yum install epel-release -y
yum install tis-waptserver tis-waptsetup cabextract nginx-mod-http-auth-spnego -y
```

- Initialiser la base de données PostgreSQL et activer les services.

```
sudo /usr/bin/postgresql-setup initdb
sudo systemctl enable postgresql waptserver nginx
sudo systemctl start postgresql nginx
```

Redhat 8 et dérivés

- Ajout du dépôt Tranquil'iT.

```
cat > /etc/yum.repos.d/wapt.repo <<EOF
[wapt]
name=WAPT Server Repo
baseurl=https://wapt.tranquil.it/redhat8/wapt-2.3/
enabled=1
gpgcheck=1
EOF
```

- Récupérer la clé `.gpg` et installer les paquets nécessaires.

```
wget -q -O /tmp/tranquil_it.gpg "https://wapt.tranquil.it/centos8/RPM-GPG-KEY-TISWAPT-8"; rpm --
↪import /tmp/tranquil_it.gpg
yum install epel-release -y
dnf module enable nginx:1.20 -y
yum install tis-waptserver tis-waptsetup cabextract nginx-mod-http-auth-spnego -y
```

- Initialiser la base de données PostgreSQL et activer les services.

```
sudo /usr/bin/postgresql-setup initdb
sudo systemctl enable postgresql waptserver nginx
sudo systemctl start postgresql nginx
```

Redhat 7 / CentOS 7 et dérivés

- Ajout du dépôt Tranquil'iT.

```
cat > /etc/yum.repos.d/wapt.repo <<EOF
[wapt]
name=WAPT Server Repo
baseurl=https://wapt.tranquil.it/centos7/wapt-2.3/
enabled=1
gpgcheck=1
EOF
```

- Récupérer la clé `.gpg` et installer les paquets nécessaires.

```
wget -q -O /tmp/tranquil_it.gpg "https://wapt.tranquil.it/centos7/RPM-GPG-KEY-TISWAPT-7"; rpm --
↪import /tmp/tranquil_it.gpg
yum install epel-release -y
yum install tis-waptserver tis-waptsetup cabextract nginx-mod-http-auth-spnego -y
```

- Initialiser la base de données PostgreSQL et activer les services.

```
sudo /usr/pgsql-14/bin/postgresql-14-setup initdb
sudo systemctl enable postgresql-14 waptserver nginx
sudo systemctl start postgresql-14 nginx
```

13.3 Post-configuration

Attention: Pour que le post-configuration fonctionne correctement, vous devez d'abord avoir un *hostname* pour le serveur WAPT. Pour vérifier, utilisez la commande **echo $(hostname)** qui doit retourner l'adresse DNS qui sera utilisé par les agents WAPT sur les ordinateurs clients.

Le script de post-configuration réécrit la configuration de nginx.

Ce script de post-configuration doit être exécuté en tant que **root**.

- Lancez le script.

```
/opt/wapt/waptserver/scripts/postconf.sh
```

- Cliquez sur *Oui* pour lancer le script de post-configuration.

```
do you want to launch post configuration tool?

           < yes >          < no >
```

- Choisissez un mot de passe (si ce n'est pas déjà défini) pour le compte *SuperAdmin* du serveur WAPT (longueur minimale de 10 caractères).

```
Please enter the wapt server password (min. 10 characters)

*****************

               < OK >          < Cancel >
```

- Confirmer le mot de passe.

```
Please enter the server password again:

*****************

               < OK >          < Cancel >
```

- Choisissez le mode d'authentification pour l'enregistrement initial des agents WAPT:
  - Le choix #1 permet d'enregistrer les ordinateurs sans authentification. Le serveur WAPT enregistre tous les ordinateurs qui demandent à être enregistré.
  - Le choix #2 active l'enregistrement initial basé sur kerberos (vous pouvez l'activer plus tard).
  - Le choix #3 n'active pas le mécanisme d'authentification kerberos pour l'enregistrement initial des machines équipées de WAPT. Le serveur WAPT va demander un identifiant et mot de passe pour chaque machine qui s'enregistre.

```
WaptAgent Authentication type?

------------------------------------------------------------------------
(x) 1 Allow unauthenticated registration
( ) 2 Enable kerberos authentication required for machines registration.
        Registration will ask for password if kerberos not available
( ) 3 Disable kerberos but registration require strong authentication
------------------------------------------------------------------------
                                        < OK >          < Cancel >
```

- Si vous souhaitez utiliser WAPT pour le déploiement de systèmes d'exploitation, sélectionnez *Yes*.

```
Do you want to activate os deployment?

        < Yes >        < No >
```

- Si vous avez répondu *Yes* pour activer le déploiement de systèmes d'exploitation avec WAPT, le script de post-configuration vous demandera si vous souhaitez utiliser une authentification sécurisée afin de déployer l'image du système d'exploitation. Il demandera alors un utilisateur / mot de passe lorsque vous essayerez de déployer le système d'exploitation.

```
Would you like to activate secure authentication on wads ?

        < Yes >        < No >
```

- Toujours en ce qui concerne la fonction WADS, si vous avez répondu *Yes* aux 2 dernières questions, vous aurez une dernière question :

```
Would you like to mention subnet ip exempt from wads authentication

        < Yes >        < No >
```

Si vous répondez *Yes* à cette question, vous devrez donner un sous-réseau IP qui peut être une liste (exemple : 192.168.0.0/24, 192.168.1.0/24).

- Sélectionnez *OK* pour démarrer le serveur WAPT.

```
Press OK to start waptserver

             < OK >
```

- Sélectionnez *Oui* pour configurer Nginx.

```
Do you want to configure nginx?

        < Yes >        < No >
```

- Remplissez le *FQDN* du serveur WAPT.

```
FQDN for the WAPT Server (eg. wapt.example.com)

--------------------------------------------
wapt.mydomain.lan
```

(suite sur la page suivante)

(suite de la page précédente)

```
-----------------------------------------

           < OK >          < Cancel >
```

- Sélectionnez *OK* et un certificat auto-signé sera généré, cette étape peut prendre un certain temps.

```
Generating DH parameters, 2048 bit long safe prime, generator 2 This is going to take a long time
.....................................+.............................+...
```

Nginx est maintenant configuré, sélectionnez *OK* pour redémarrer **Nginx**:

```
The Nginx config is done.
We need to restart Nginx?

              < OK >
```

Le post-confoguration est maintenant terminé.

```
Postconfiguration completed.
Please connect to https://wapt.mydomain.lan/ to access the WAPT Server.

                                    < OK >
```

Liste des options du script de post-configuration:

Options	Description
`--force-https`	Configurer **Nginx** de sorte que *le port 80 soit redirigé de façon permanente vers le 443*

Votre serveur WAPT est maintenant prêt. Vous pouvez consulter la documentation sur *l'installation de la console WAPT*.

CHAPTER 14

Installer le serveur WAPT sur Windows

Attention:

- Le serveur WAPT ne peut pas être installé sur une machine qui écoute déjà sur le port 80 et 443 (exemple WSUS avec IIS).
- Les ports 80, 443 et 8080 sont utilisés par le serveur WAPT et doivent être disponibles.
- Si les ports 80 et 443 sont déjà occupés par un autre service web, vous devriez consulter la documentation officielle de Microsoft pour modifier les ports par défaut sous Windows.
- Le serveur WAPT **ne fonctionnera pas** sur une version x86 de Windows.
- L'installation du serveur WAPT doit être effectuée en utilisant un compte **Administrateur local** sur l'hôte et **PAS un compte Administrateur de domaine**.

Danger: `Nginx` est le **seul** serveur web supporté avec WAPT. **Apache ou IIS (avec ou sans WSUS) ne sont PAS supportés par WAPT**.

En cas de difficulté lors de l'installation de WAPT, visitez *la Foire Aux Questions*.

Note:

- L'installation de WAPT sur un serveur Linux est la méthode recommandée, sauf si vous testez WAPT et que vous n'êtes pas familier avec Linux.
- Le serveur WAPT peut être installé sur un **système 64bit seulement**, pour les nouvelles installations utiliser une **:raw-html:`<font color="red"><b>Version de Windows actuellement supportée par Microsoft</b></font>`**.

Indication: Le composant serveur de WAPT fonctionne aussi bien sur une VM client win10 ou une machine physique que sur une

version serveur de Windows.

- Télécharger et exécuter `waptserversetup.exe`.

Attention: L'installation du serveur WAPT doit être effectuée en utilisant un compte **Administrateur local** sur l'hôte et **PAS un compte Administrateur de domaine**.

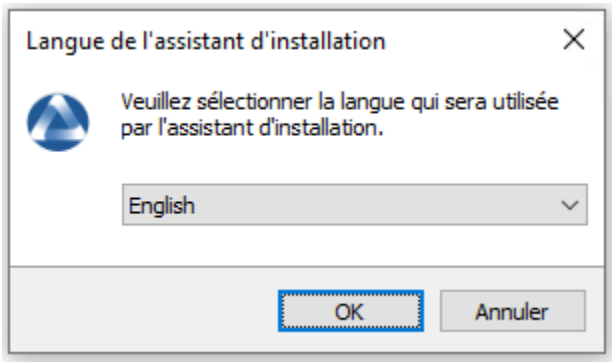
Avertissement: Le script de post-configuration réécrit la configuration de nginx. Si vous utilisez une configuration spéciale, sauvegardez votre fichier `nginx.conf` avec la commande :

```
copy C:\wapt\waptserver\nginx\conf\nginx.conf C:\wapt\waptserver\nginx\conf\nginx.conf.old
```

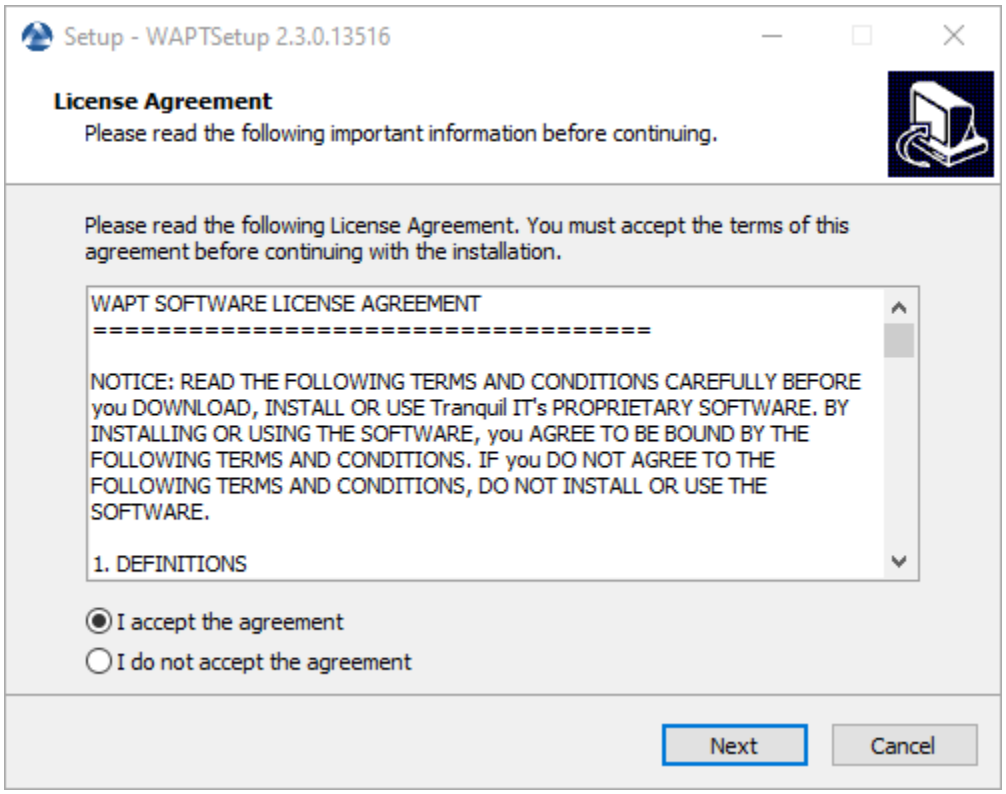
Il sera nécessaire d'écraser la configuration après le post-configuraiton avec la commande:

```
copy C:\wapt\waptserver\nginx\conf\nginx.conf.old C:\wapt\waptserver\nginx\conf\nginx.conf
```

- Choisir la langue de l'installeur WAPT.



- Acceptez la licence publique GNU et cliquez sur *Suivant* pour passer à l'étape suivante.



- Acceptez la licence publique GNU et cliquez sur *Suivant* pour passer à l'étape suivante.
- Choisissez une tâche supplémentaire (laissez la valeur par défaut si vous n'êtes pas sûr).
- Choisissez le mot de passe pour le serveur WAPT.

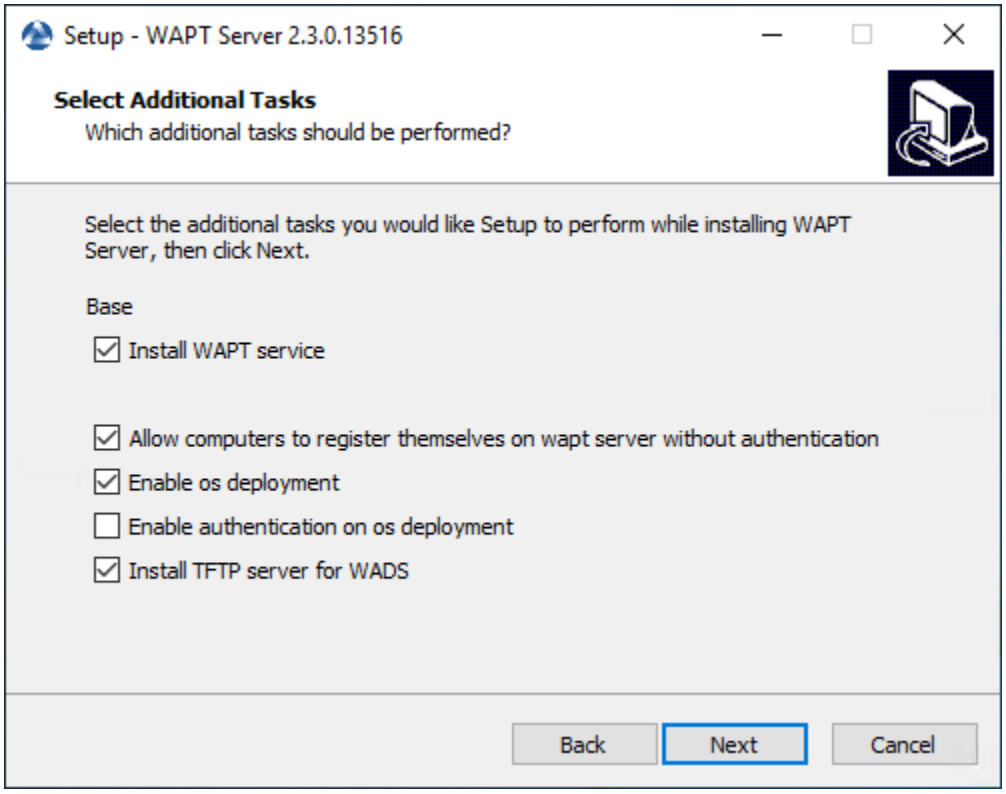
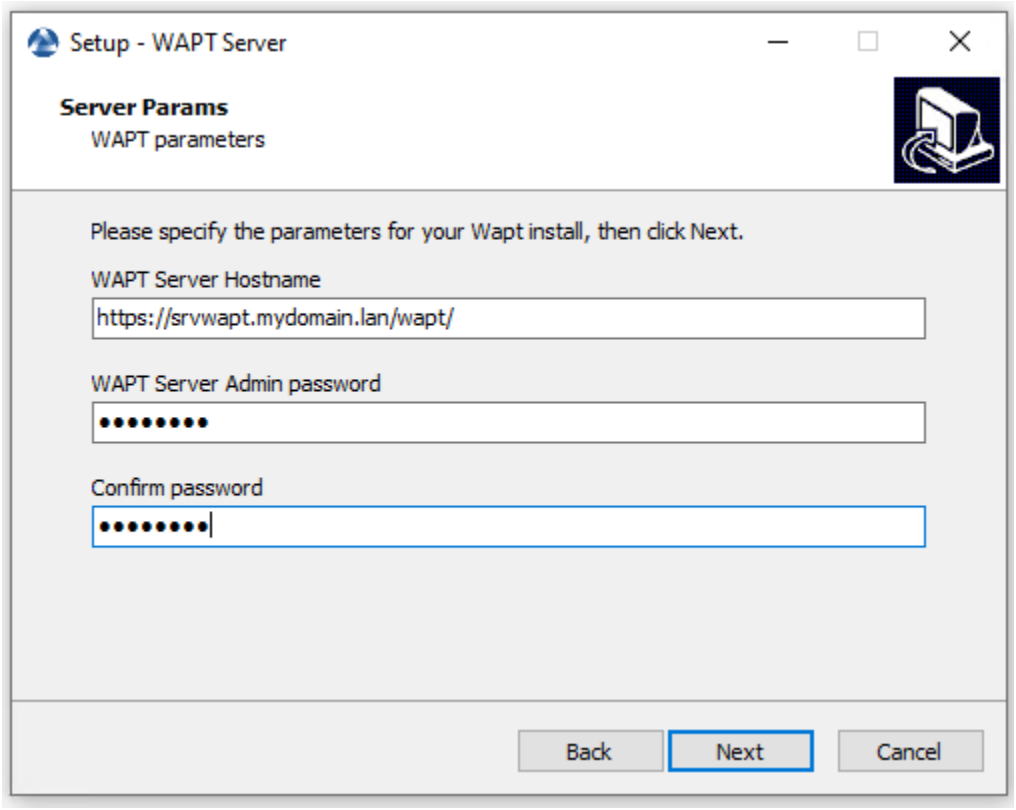
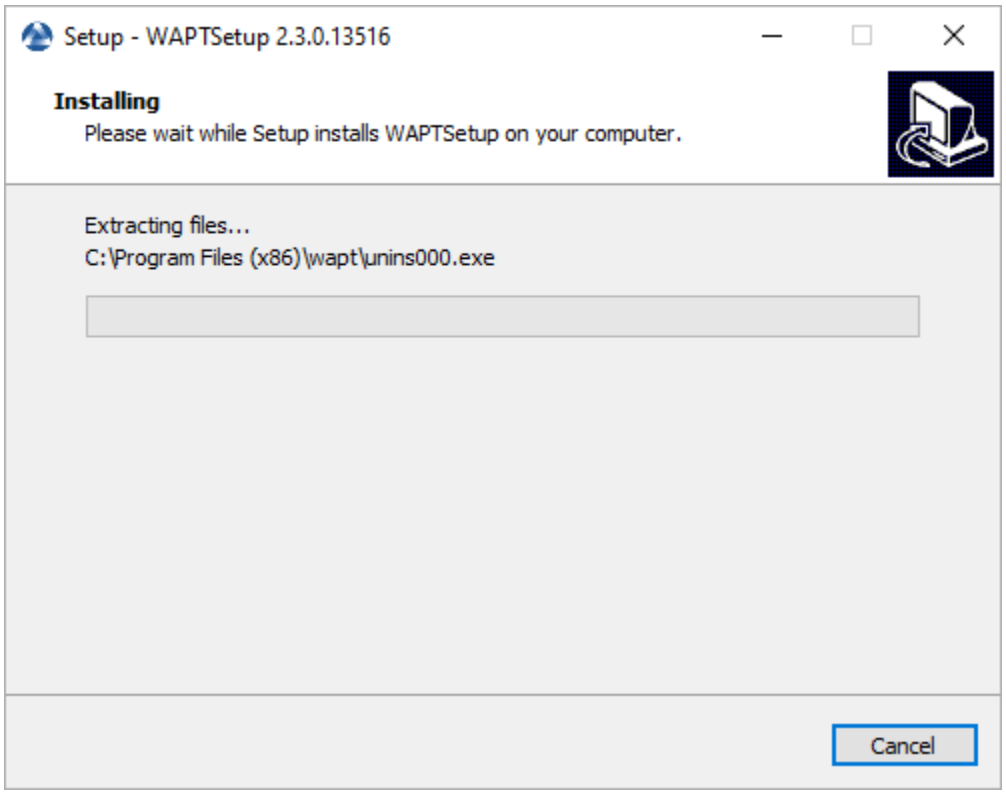


Figure1: Choix des options du programme d'installation pour le déploiement du serveur WAPT



- Cliquez sur le bouton *Install* pour lancer l'installation, attendez que l'installation soit terminée.



- Cliquez sur *Terminé* pour fermer la fenêtre.

Attention:

- **Pour des raisons de sécurité, n'exécutez pas la console WAPT ou votre outil de développement de paquet WAPT sur le serveur WAPT.**
- Le Serveur WAPT sur Windows **embarque un Agent WAPT**. Il n'est pas nécessaire d'installer l'Agent WAPT pour administrer le Server WAPT sur Windows.

Votre serveur est maintenant prêt. Vous pouvez maintenant consulter la documentation sur *Installation de la console de gestion WAPT*.

Le serveur WAPT ayant été installé avec succès, nous allons maintenant installer la console WAPT.

CHAPTER 15

Console de gestion WAPT

Si vous avez déjà *généré l'agent WAPT* et *déployé l'agent* sur le poste de travail de votre *Administrateur*, alors *lancez la console WAPT*.

- La gestion de WAPT se fait principalement via la console WAPT installée sur le poste de travail de l'"*Administrateur*.
- Il est recommandé de joindre l'ordinateur de l'administrateur à l'Active Directory de l'" *Organisation*.
- Le nom d'hôte du poste de travail de l'administrateur **ne doit pas comporter plus de 15 caractères**, ce qui est une limite de l'attribut *sAMAccountName* dans Active Directory.
- **L'ordinateur de l'administrateur deviendra essentiel pour l'administration de WAPT et le test des paquets WAPT.**
- Si les enregistrements DNS sont correctement configurés, vous devriez être en mesure d'accéder à l'interface web WAPT en visitant https://srvwapt.mydomain.lan.
- En date du 2023-04-17, la Console WAPT n'est supportée que sous Windows. Les versions Linux et macOS sont en cours de développement.

Avertissement: La console WAPT **NE DOIT PAS** être installée sur votre serveur WAPT basé sur Windows.

La console WAPT doit être installée sur le poste de travail à partir duquel vous gérez votre réseau.

Pour télécharger le fichier `waptsetup.exe`, pointer votre navigateur Web sur votre url waptserver https://srvwapt.mydomain.lan, puis cliquer sur le lien *WAPTSetup* sur le côté droit de la page Web du serveur WAPT. La page d'accueil du serveur WAPT ne fournit que des informations de base sur l'état du serveur et un lien de téléchargement de la console WAPT.

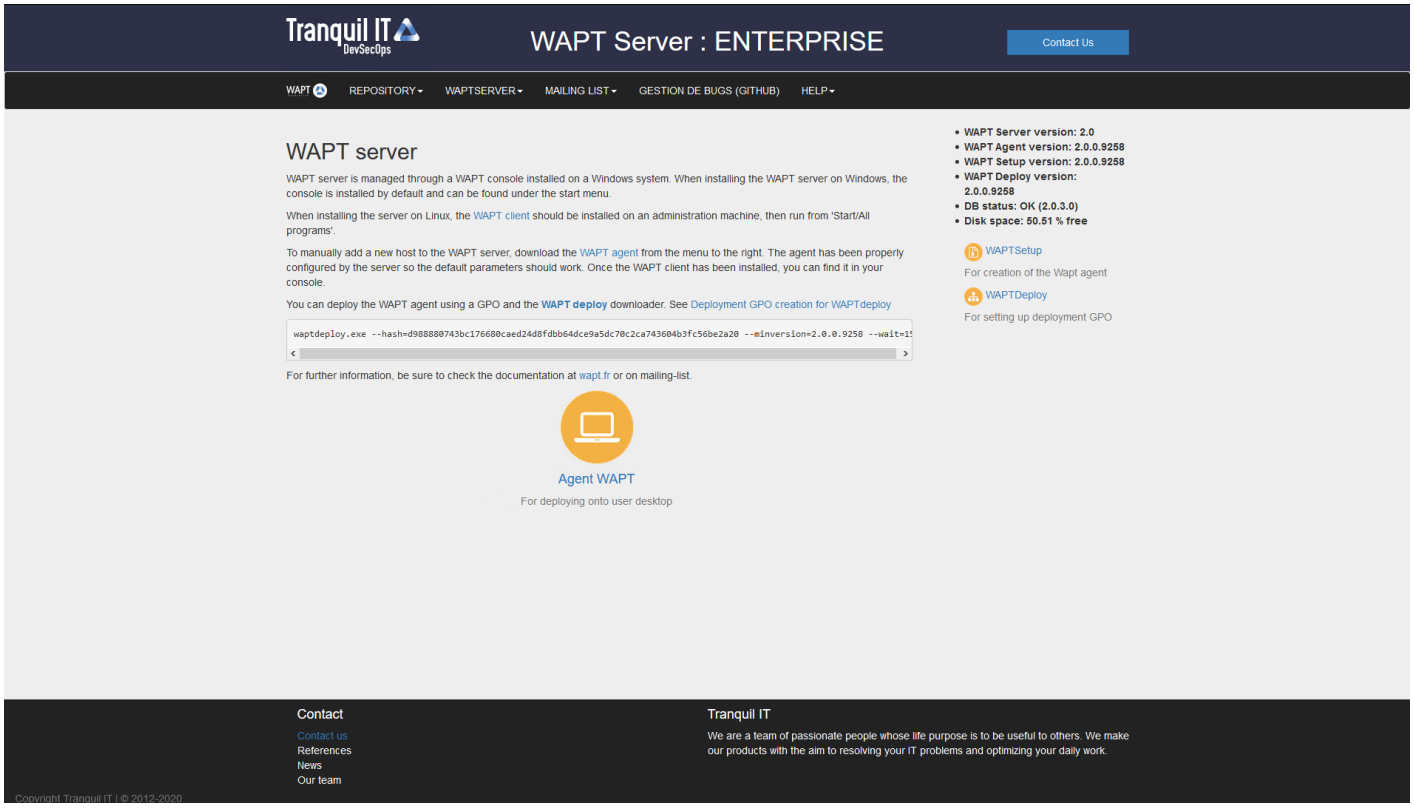
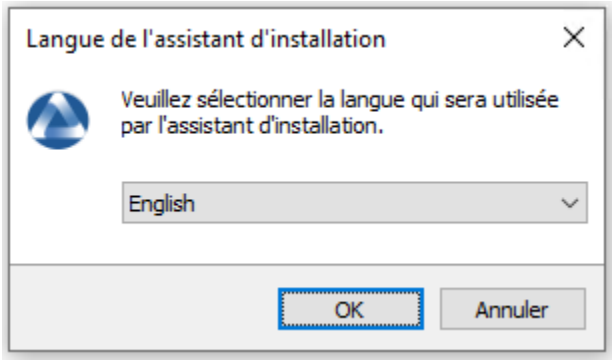


Figure1: L'interface du serveur WAPT dans un navigateur web

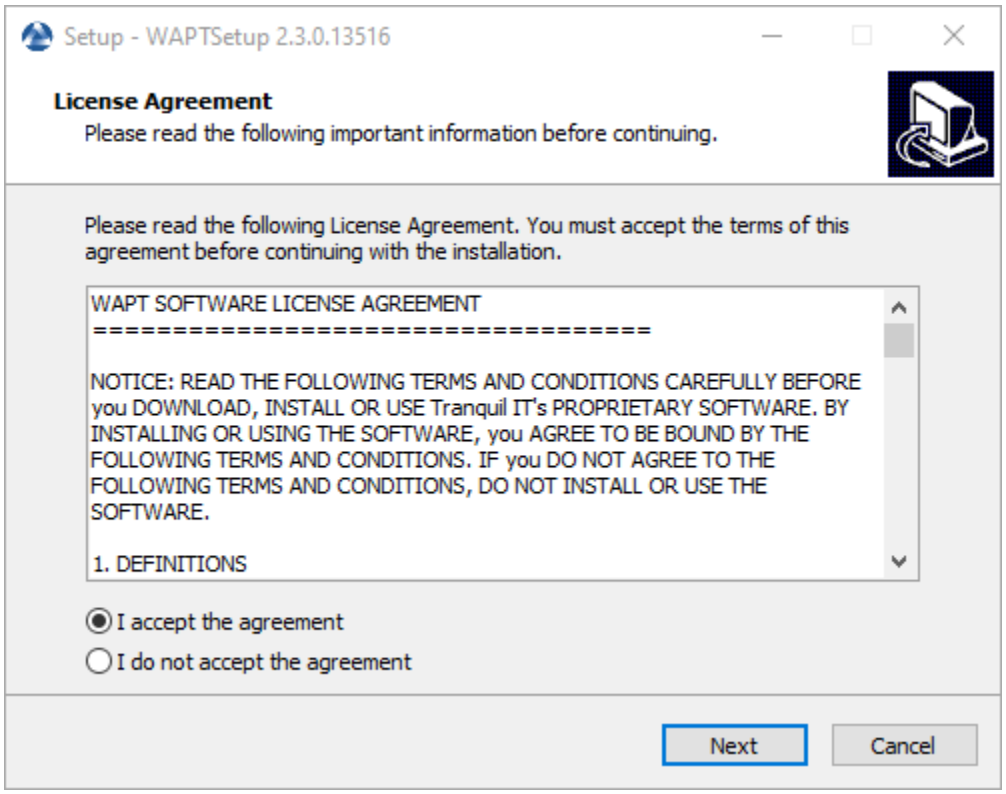
15.1 Installation sur l'ordinateur de l'administrateur

Attention: Si **waptagent** n'est pas compilé et installé sur votre ordinateur, vous devez installer waptsetup.

- Lancez le programme d'installation exécutable en tant que *Administrateur local* sur le poste de travail de l'*Administrateur*.
- Choisir la langue pour WAPT.



- Cliquez sur *OK* pour passer à l'étape suivante.



- Acceptez les conditions de la licence et cliquez sur *Next* pour passer à l'étape suivante.
- Choisissez des tâches de configuration supplémentaires (laissez la valeur par défaut si vous n'êtes pas sûr).

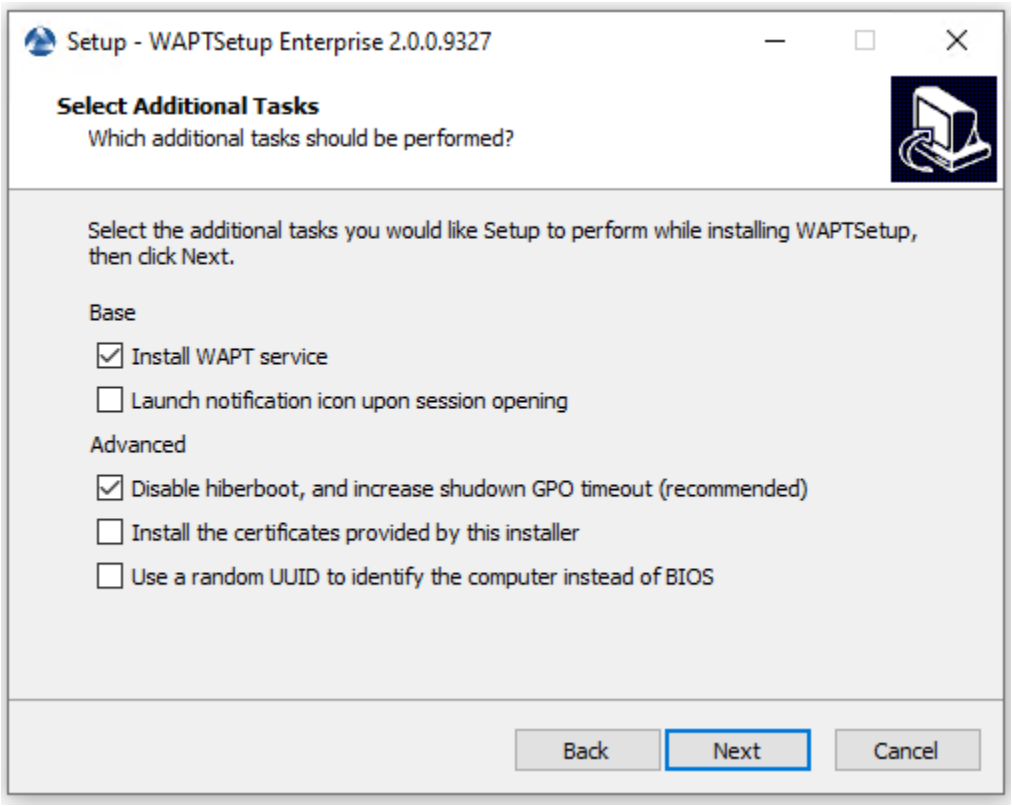


Figure2: Choisir les options de l'installateur

Table1: Choisissez les options de l'installateur

Paramètres	Description	Valeur par défaut
Organisation	Active le service WAPT sur cet ordinateur.	Coché
Lancer l'icône de notification lors de l'ouverture de session	Lancer waptagent dans la barre d'état système au démarrage.	Non coché
Désactiver l'hiberboot, et augmenter le temps pour les GPO (recommandé)	Désactiver le démarrage rapide de Windows pour la stabilité, élargir le délai d'attente pour WAPTExit.	Coché
Installer les certificats fournis par cet installeur	Installez le certificat Tranquil iT uniquement sur cet ordinateur.	Non coché
Utiliser un UUID aléatoire pour identifier l'ordinateur au lieu du BIOS	Pour plus d'informations, consultez la documentation sur *BIOS UUID bugs*	Non coché

- Configurez l'URL du serveur WAPT .

Indication: Ici, deux choix s'offrent à vous.

- S'il s'agit de la première installation et que l'agent WAPT.
  - Vérifiez les « Informations statiques WAPT » et définissez-les :
    - URL du dépot WAPT : http://srvwapt.mydomain.lan/wapt.
    - URL du serveur WAPT : https://srvwapt.mydomain.lan.
- Si la console WAPT ou l'agent WAPT est déjà installé:
  - Cochez *Ne pas modifier la configuration actuelle*, puis cliquez sur *Suivant*.
- Cliquez sur *Installer* pour lancer l'installation, attendez que l'installation se termine, puis cliquez sur *Terminé* (laissez les options par défaut).

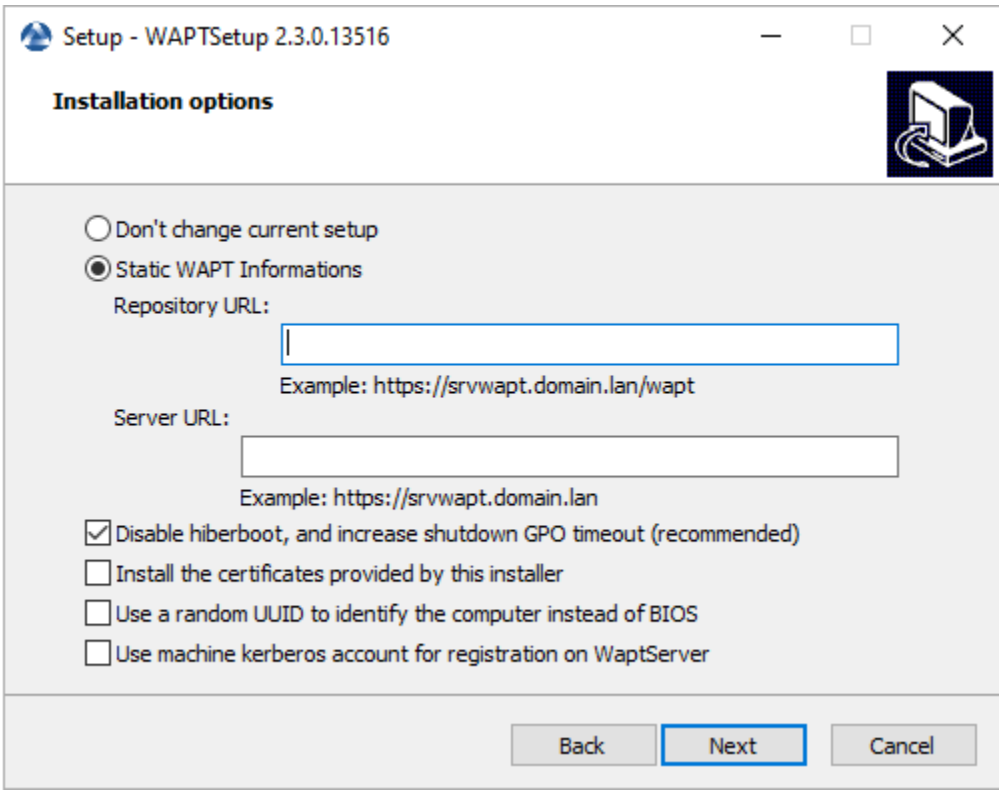


Figure3: Choix du dépot et du serveur WAPT

- Choisissez le référentiel WAPT et le serveur WAPT ; cliquez sur *Suivant*.

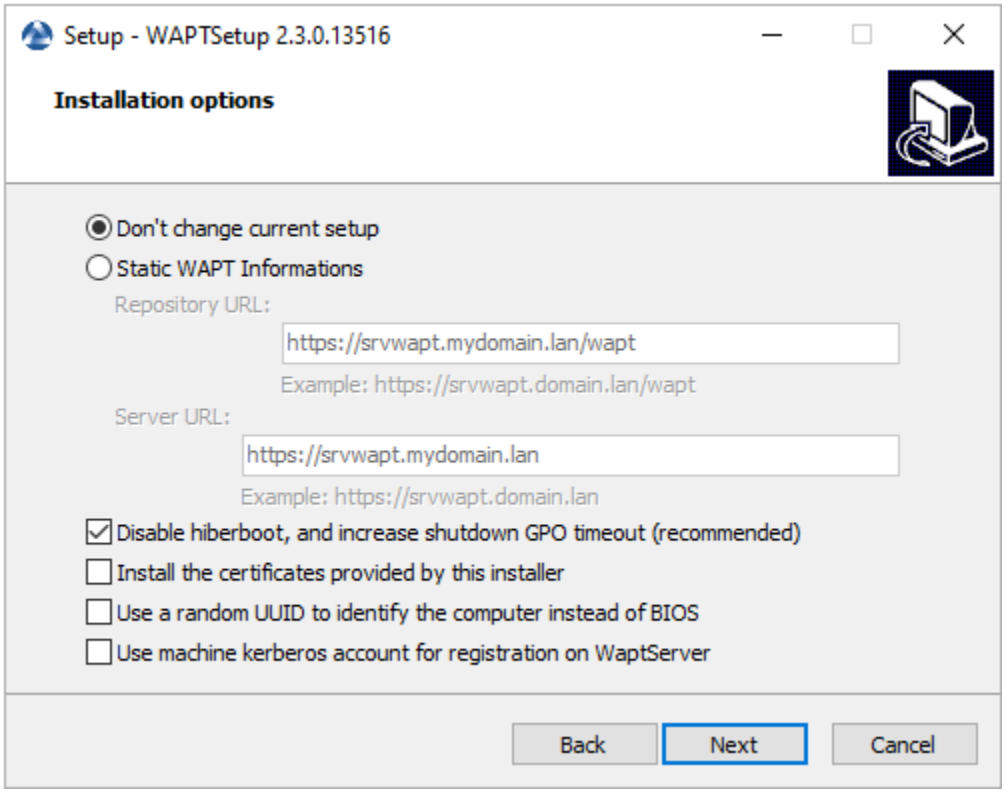


Figure4: Le dépôt et le serveur WAPT sont déjà configurés

- Obtenir un résumé de l'installation de la console WAPT.

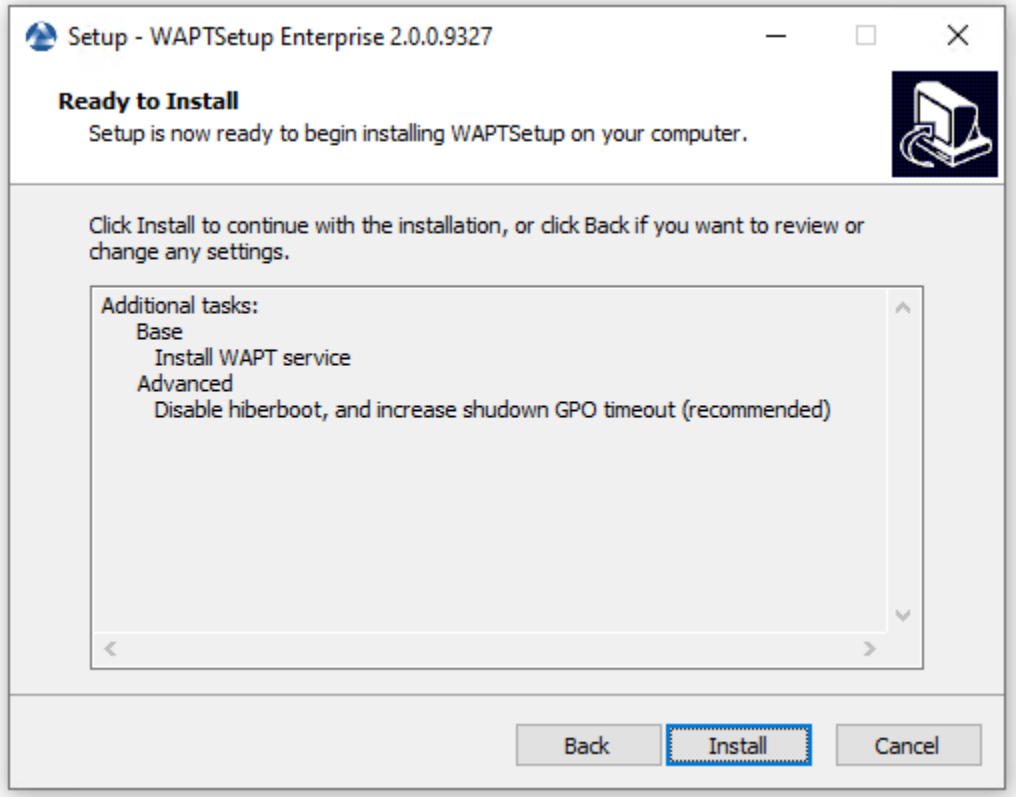
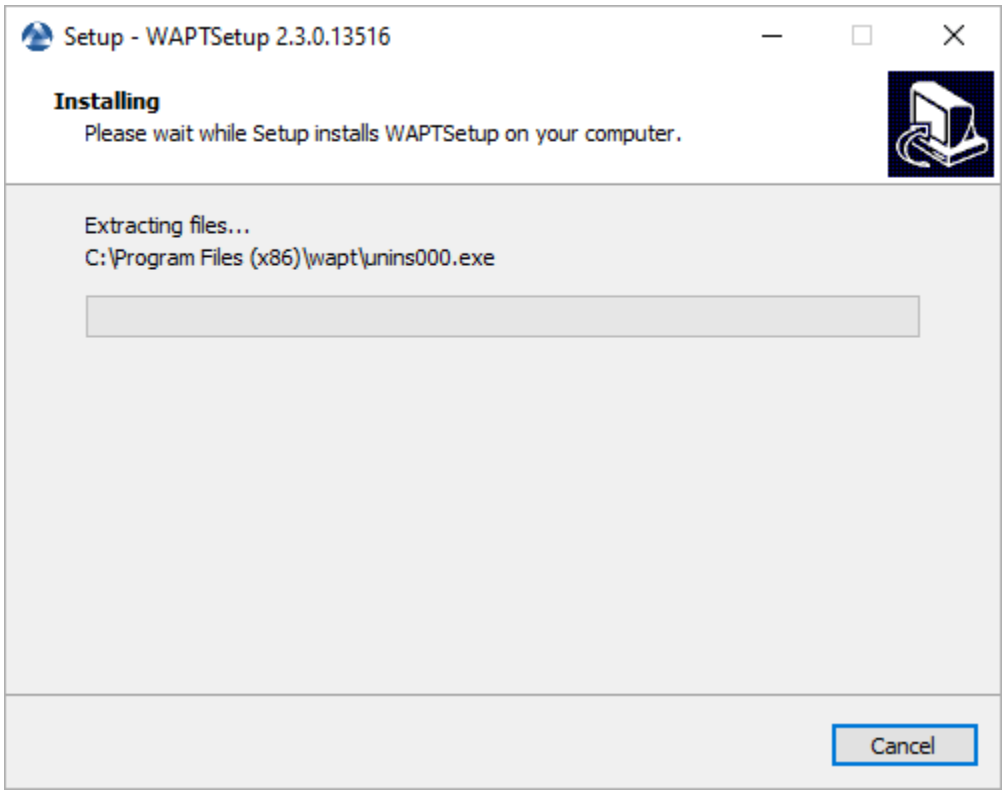
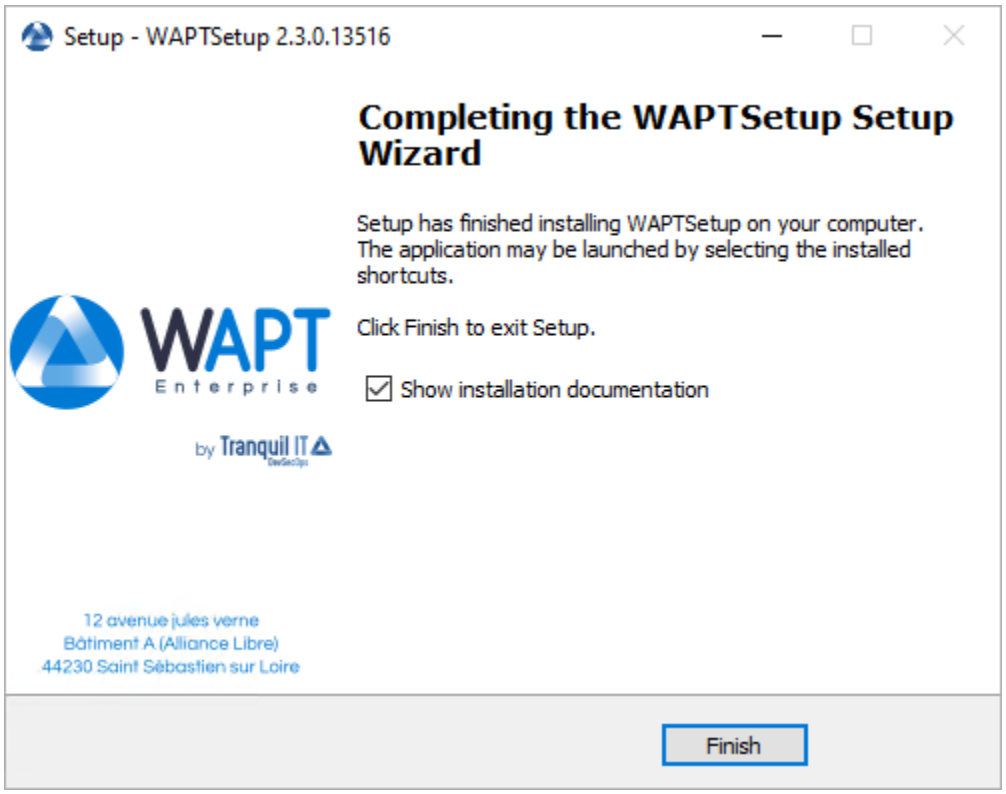


Figure5: Obtenir un résumé de l'installation de la console WAPT





- Décochez *Afficher la documentation d'installation*.

15.2 Démarrer la console WAPT

- Lancez la console WAPT :
  - En cherchant le binaire.

    `C:\Program Files (x86)\wapt\waptconsole.exe`
  - Ou en utilisant le menu *Démarrer*.

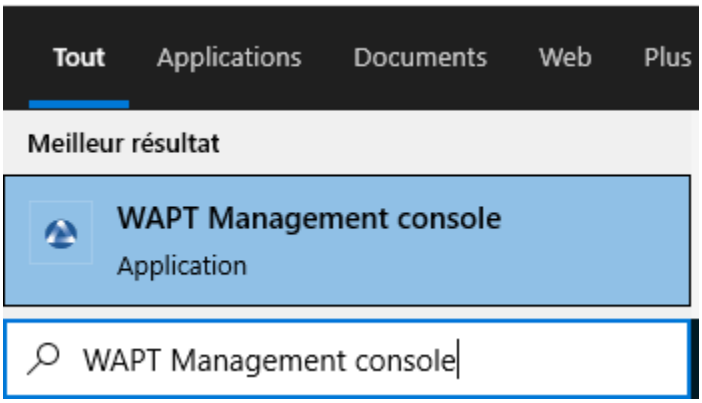


Figure6: Lancement de la console WAPT à partir du menu de démarrage de Windows

- Connectez-vous à la console WAPT avec le login et le mot de passe *SuperAdmin*.

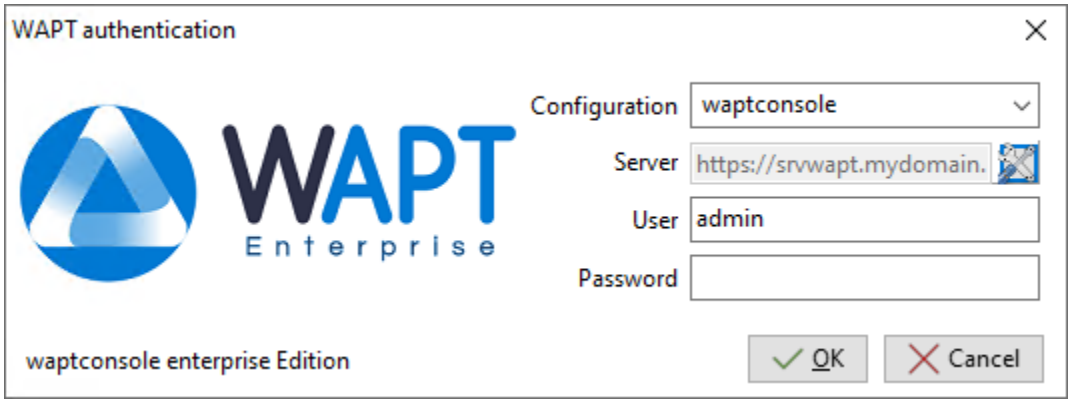


Figure7: La fenêtre d'authentification de la console WAPT

Si vous avez des problèmes pour vous connecter à la console WAPT, veuillez vous référer à la FAQ : *Message d'erreur à l'ouverture de la console*.

Il est recommandé de lancer la console WAPT avec un compte d'administrateur local pour permettre le débogage local des paquets WAPT.

Pour la version Enterprise, il est possible de s'authentifier avec l'"*Active Directory*.

15.2.1 Premier démarrage après l'installation du serveur

Indication: Au premier démarrage, vous devez lancer la console WAPT avec des privilèges élevés. *Cliquez avec le bouton droit de la souris sur le binaire de la console WAPT → Démarrer en tant qu'administrateur local*.

Affectation du certificat

Note: Un message peut apparaître indiquant qu'aucun certificat personnel n'a été défini.

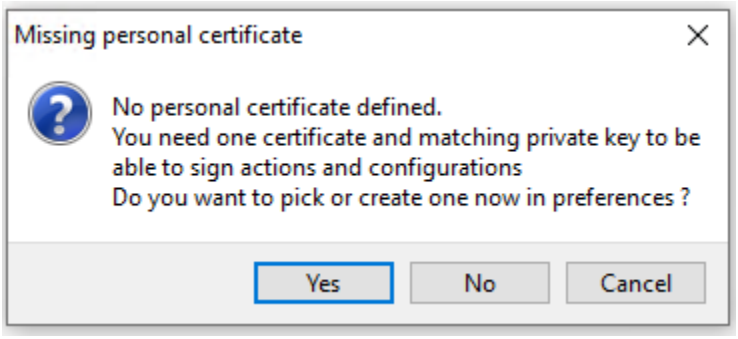


Figure8: Certificat personnel WAPT non présent

- Sélectionnez *Oui*
- Cliquez sur *Générer un certificat* puis allez *créer votre certificat.*

Définition du préfixe de paquet

Note: Un message peut apparaître indiquant qu'aucun certificat personnel n'a été défini.

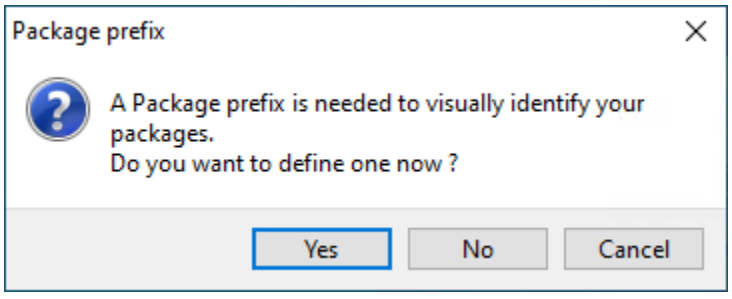


Figure10: Boîte de dialogue informant qu'aucun préfixe n'a été défini dans la configuration WAPT

- Sélectionnez *Oui*

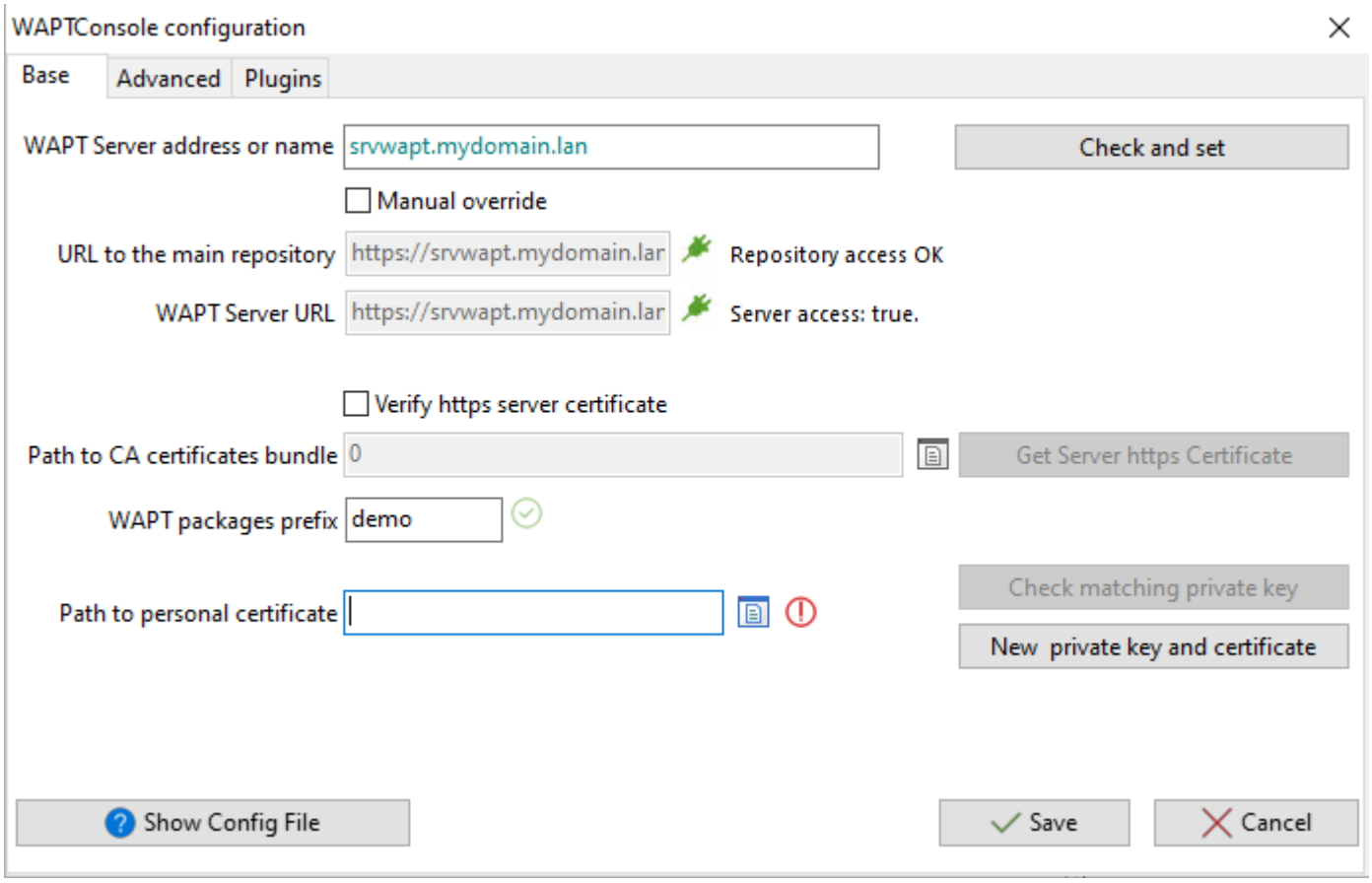


Figure9: Fenêtre pour la configuration basique de la console WAPT

- Définissez votre préfixe de paquet sur *préfixe des paquets WAPT*

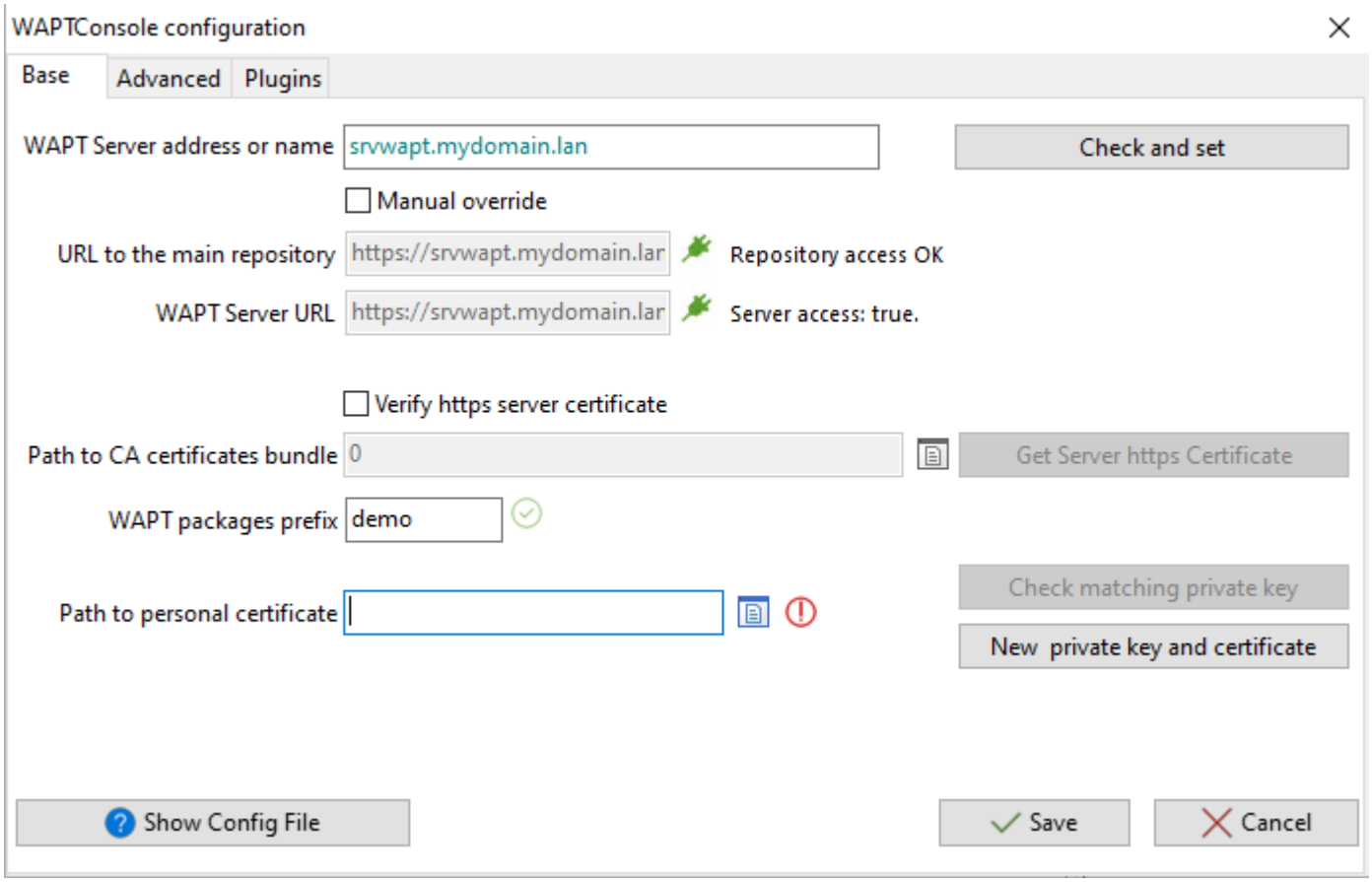


Figure11: Fenêtre pour la configuration basique de la console WAPT

Avertissement: Le préfixe est sensible à la casse, nous recommandons d'utiliser les minuscules.

erreurs du waptagent.exe

Note: Un message peut apparaître indiquant que la version de votre agent WAPT est obsolète ou n'existe pas encore.

Si le *certicat de l'Administrateur* existe, il est possible de *générer un nouvel agent* en cliquant sur *Oui*.

Aussi, cliquez sur *Non* et générez le *certificat Administrateur*.

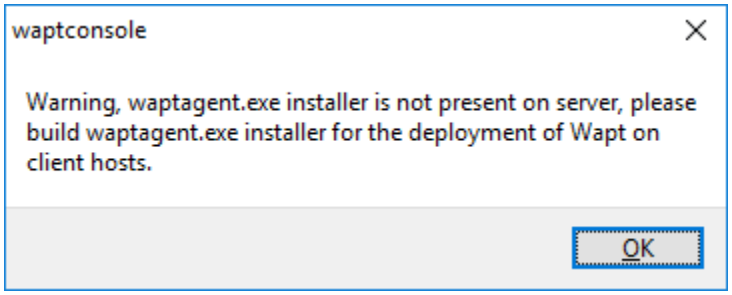


Figure12: Boîte de dialogue informant que l'agent WAPT n'est pas présent sur le serveur WAPT

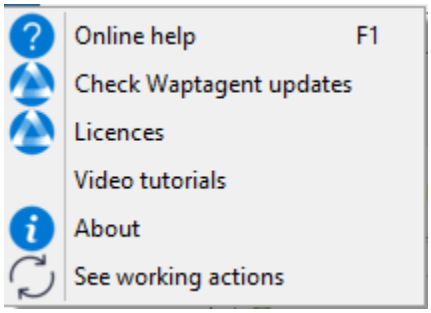
CHAPTER 16

Activer la licence

Sur WAPT, la différence entre les versions **Discovery** et **Enterprise** est gérée par la licence utilisée.

Pour activer la licence, utilisez le fichier `licence.lic` communiqué par notre département de vente.

- Dans la console WAPT, cliquez sur l'onglet *?* :



- Choisissez *Licences* :
- Sélectionner le fichier `licence.lic` et cliquer sur *Ouvrir* :

16.1 Supprimer la licence

- Dans la console WAPT, cliquez sur l'onglet *?* :

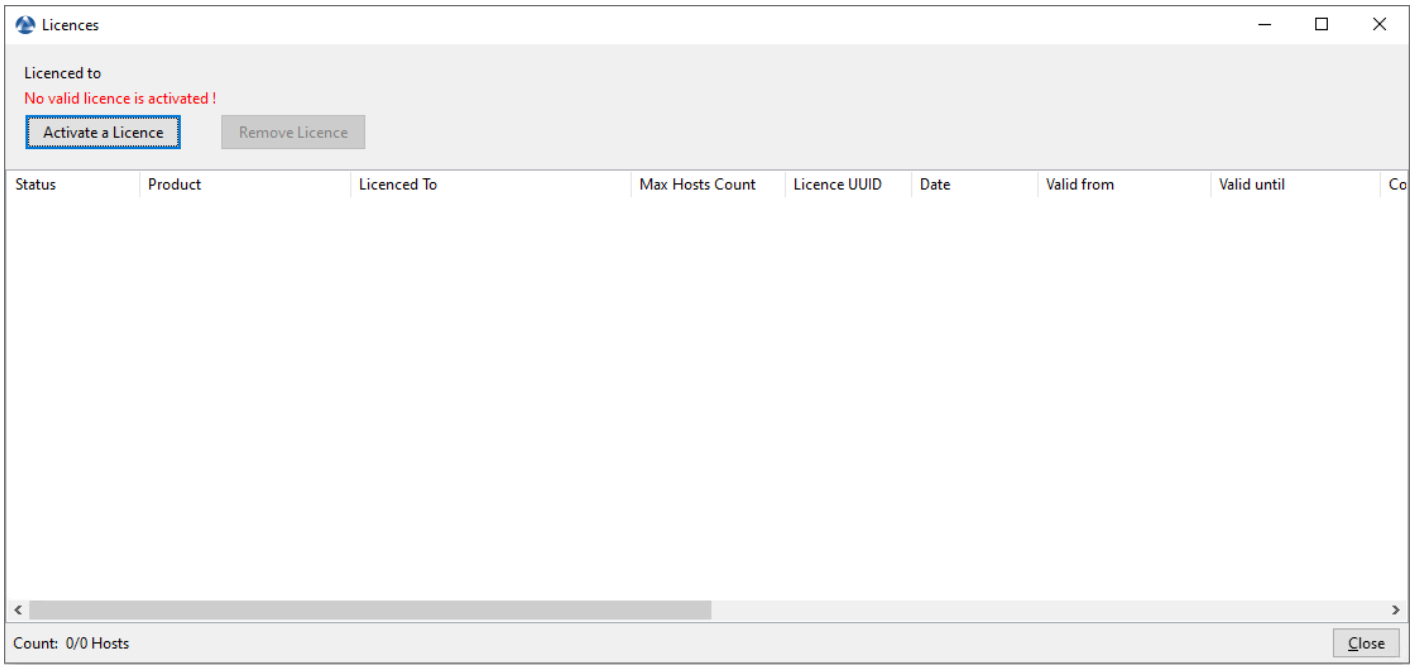


Figure1: Fenêtre indiquant qu'il n'y a pas de licences WAPT souscrites dans la console WAPT

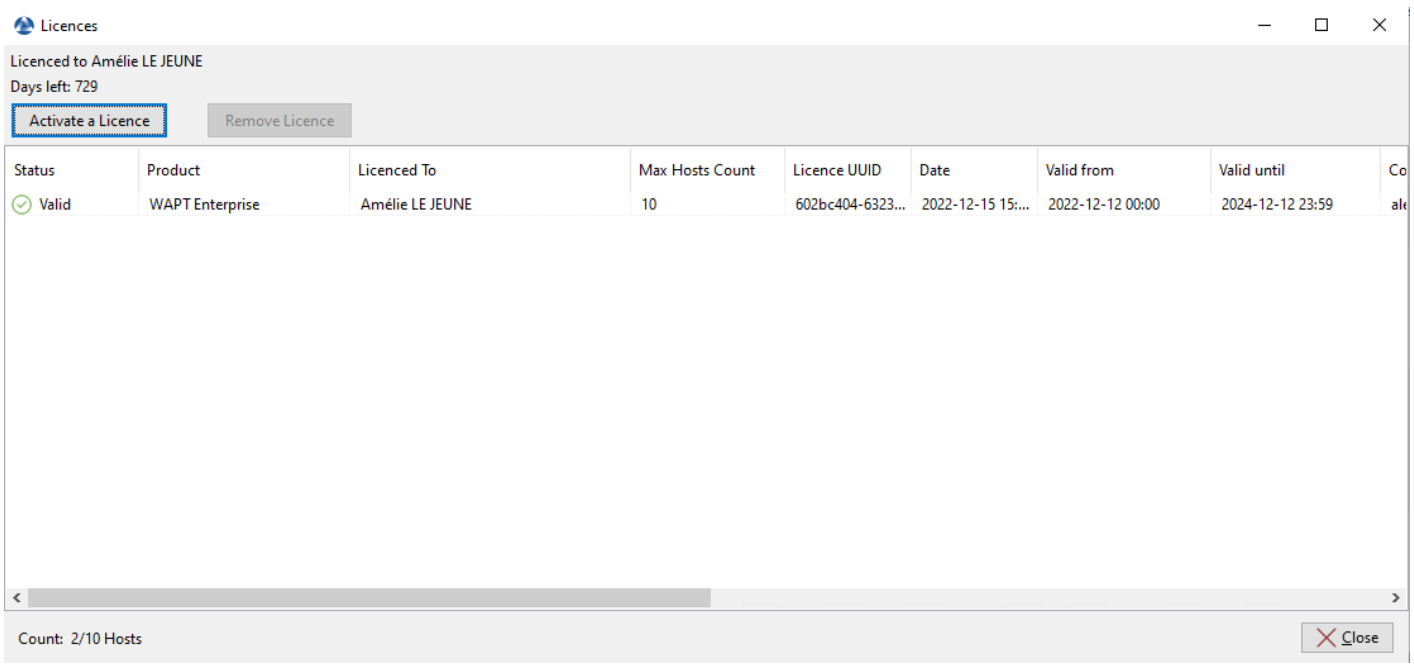
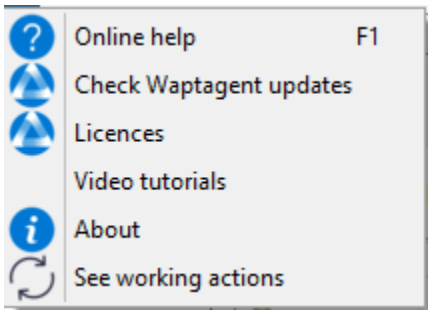


Figure2: Fenêtre montrant une licence activée dans la console WAPT



- Choisissez *Licences* :

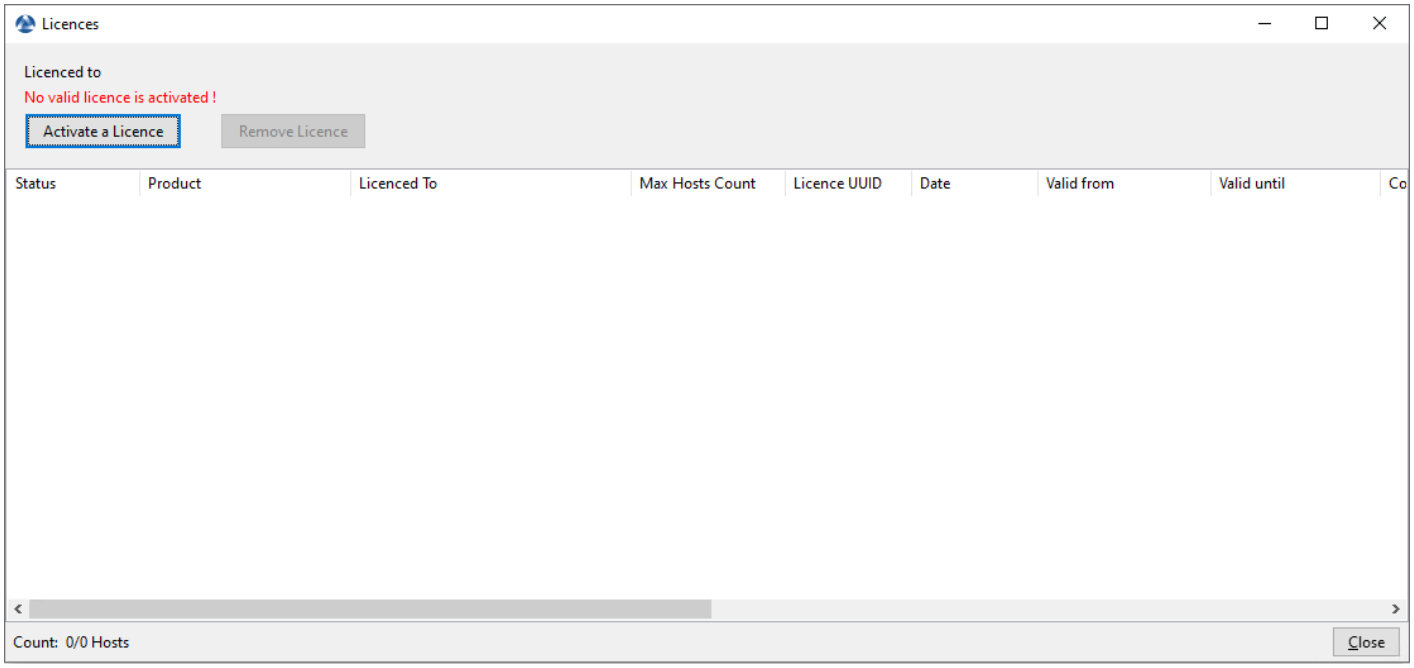


Figure3: Fenêtre indiquant qu'il n'y a pas de licences WAPT souscrites dans la console WAPT

- Sélectionnez la ligne et cliquer sur *Retirer la licence* :

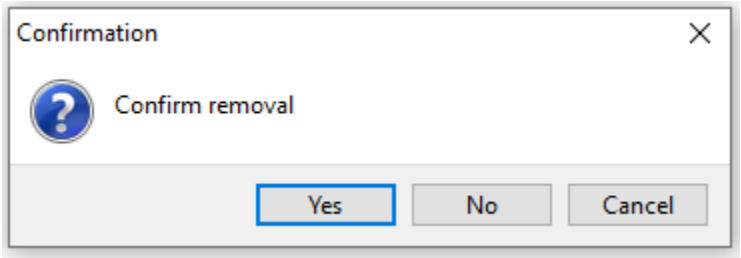


Figure4: Fenêtre de confirmation pour retirer une licence de la Console WAPT

- Après confirmation, la licence est retirée :

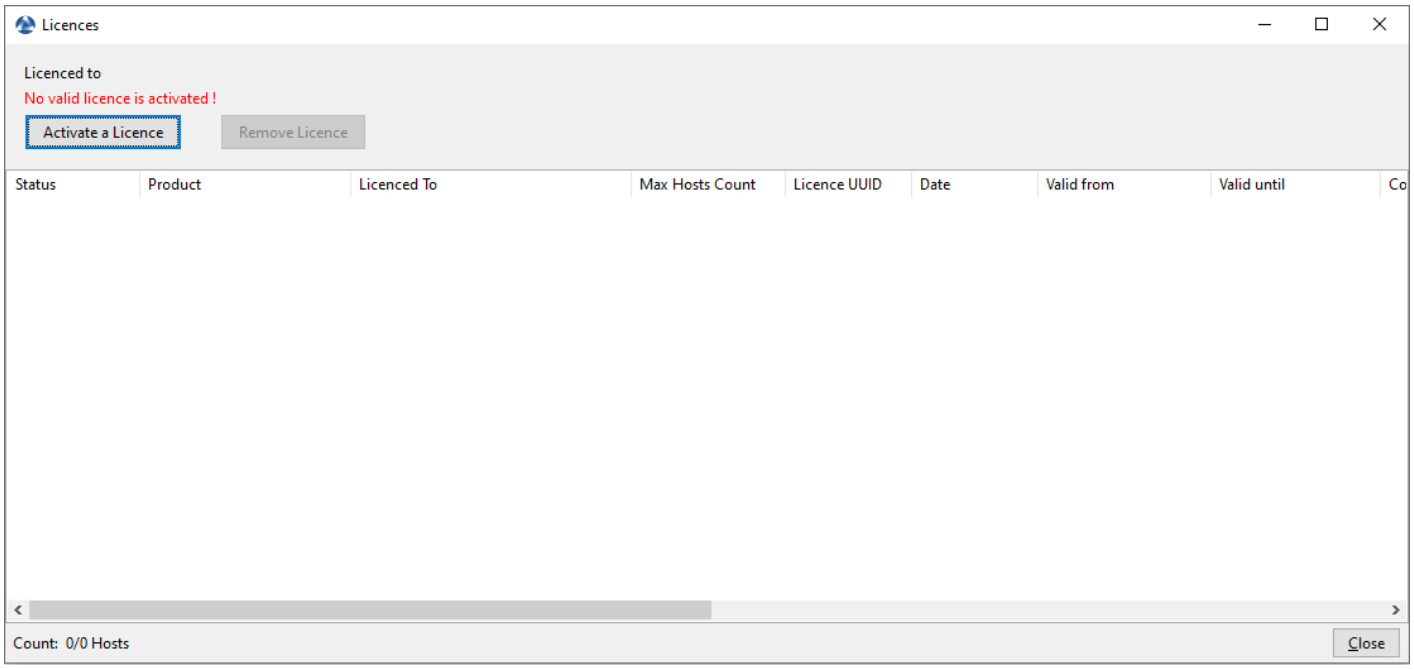


Figure5: Fenêtre indiquant qu'il n'y a pas de licences WAPT souscrites dans la console WAPT

16.2 Emplacement de la licence

Le `licence.json` est stocké sur le serveur sur l'emplacement suivant :

Debian / Ubuntu

```
/var/www/licences.json
```

RedHat and derivatives

```
/var/www/html/licences.json
```

Windows

```
C:\wapt\waptserver\repository\licences.json
```

16.3 Erreur de licence

16.3.1 Expiration de la licence

Si la licence est expirée, le statut affiche *Expiré*.

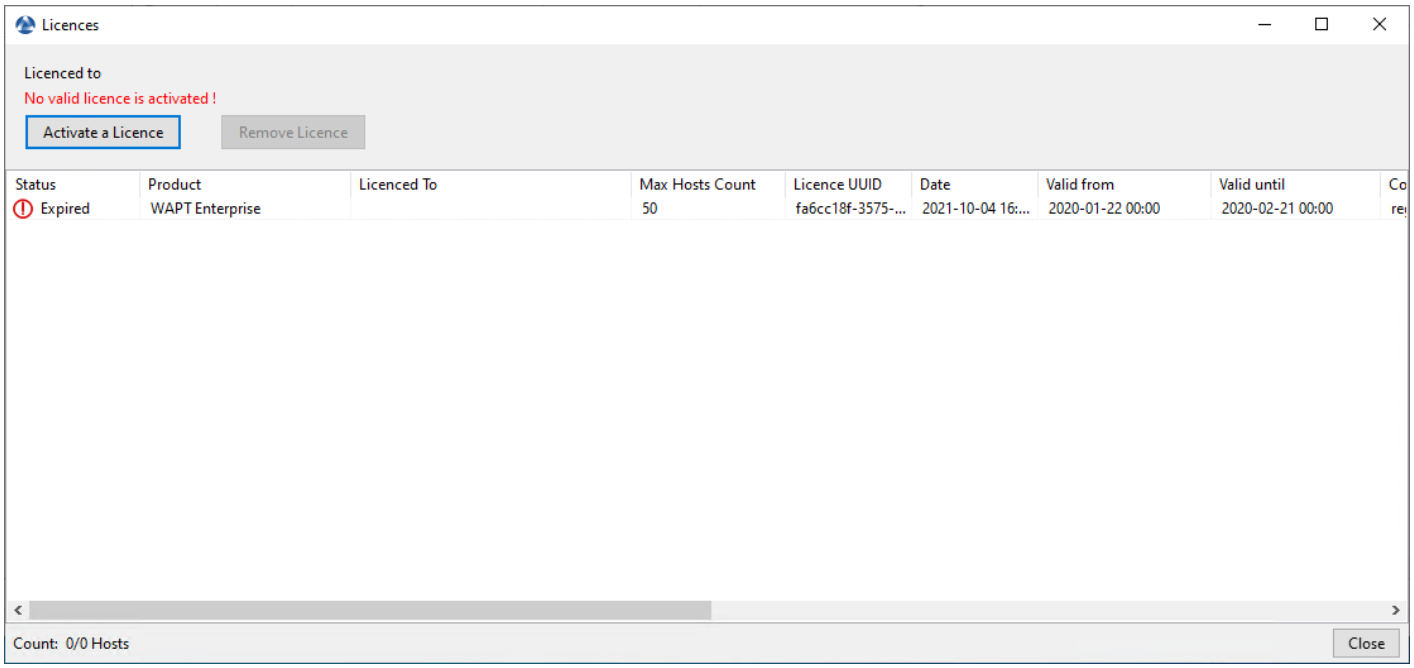


Figure6: Fenêtre montrant une licence expirée dans la console WAPT

16.3.2 Emplacement de l'ancienne licence

Sur la console d'installation, si la licence est localisée sur l'ancien emplacement, cette erreur apparaît :

16.3.3 Erreur d'activation de la licence

Cette erreur est dû à un problème avec le script de post-configuration et une configuration spéciale de NGINX.

3 points sont à vérifier :

1. Vérifier si `/etc/nginx/sites-enabled/wapt.conf` est un lien symbolique du fichier `/etc/nginx/sites-available/wapt.conf` en utilisant cette commande :

```
ls -l /etc/nginx/sites-enabled/wapt.conf
```

   - Si le lien symbolique existe, la sortie est :

```
lrwxrwxrwx 1 root root 36 Jun 9 09:35 /etc/nginx/sites-enabled/wapt.conf --> /etc/nginx/
↪sites-available/wapt.conf
```

   - Si le lien symbolique n'existe pas, supprimez `/etc/nginx/sites-enabled/wapt.conf` et créez un nouveau lien symbolique :

```
rm /etc/nginx/sites-enabled/wapt.conf

ln -s /etc/nginx/sites-available/wapt.conf /etc/nginx/sites-enabled/wapt.conf
```

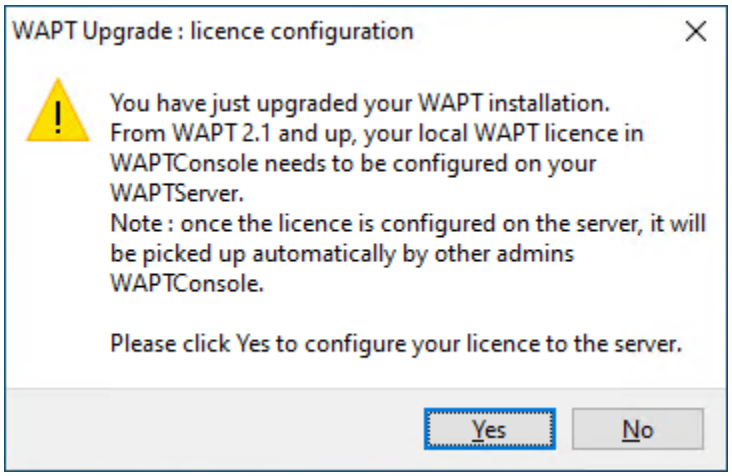


Figure7: Message d'erreur de la licence WAPT lors de la mise à niveau de WAPT vers 2.1

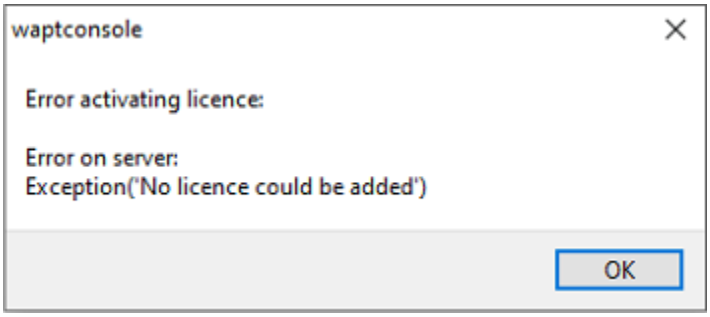


Figure8: Boîte de dialogue informant qu'une erreur s'est produite lors de l'activation d'une licence WAPT

2. Vérifier si le fichier `licences.json` est présent dans la section *location* du fichier `/etc/nginx/sites-enabled/wapt.conf` :

```
location ~ ^/(wapt/waptsetup-tis.exe|wapt/waptagent.exe|wapt/waptdeploy.exe|sync.
↪json|rules.json|licences.json)$ {
        add_header Cache-Control "store, no-cache, must-revalidate, post-check=0, pre-
↪check=0";
        add_header Pragma "no-cache";
        root "/var/www";
    }
```

   - Si le fichier `licences.json` existe, alors redémarrez **`Nginx`** :

```
systemctl restart nginx
```

   - Egalement, ajouter `licences.json` dans la section *location* dans `/etc/nginx/sites-enabled/wapt.conf` et redémarrer NGINX.

```
systemctl restart nginx
```

3. `/var/www/licences.json` vide:

```
> /var/www/licences.json
```

   - Retenter *l'activation de la licence*.

CHAPTER 17

Génération du certificat principal

- Dans cet exemple, le nom de la clé privée est `wapt-private.pem`.
- Dans cet exemple, le nom du certificat public signé avec la clé privée est `wapt-private.crt`.

17.1 Clef privée *wapt-private.pem*

Danger: Le fichier `wapt-private.pem` est **fondamental pour la sécurité**. Il **DOIT** être stocké dans un endroit sûr et correctement protégé. Le fichier `wapt-private.pem` **NE DOIT PAS** être stocké sur le Serveur WAPT.

Le fichier `wapt-private.pem` est la clé privée, il est situé par défaut dans le dossier `C:\private` du poste *Administrateur* et est protégé par un mot de passe.

Cette clé privée sera utilisée avec le certificat pour signer les paquets avant de les télécharger sur le dépot WAPT.

17.2 Certificat public : *wapt-private.crt*

Le fichier `wapt-private.crt` est le certificat public qui est utilisé avec la clé privée. Il est créé par défaut dans le dossier `C:\private` de l'administrateur, copié et déployé dans `C:\Program Files (x86)\wapt\ssl` sur les postes de travail Windows ou dans `/opt/wapt/ssl` sur les périphériques Linux et MacOS gérés par l'administrateur via un package WAPT, un GPO ou un rôle Ansible.

Ce certificat est utilisé pour valider la signature des paquets avant leur installation.

Attention:

- Si le certificat public utilisé sur la console WAPT n'est pas dérivé de la clé privée utilisée pour générer les agents WAPT, aucune interaction ne sera possible.
- Les certificats enfants des clés privées sont fonctionnels pour les interactions.

17.3 Génération d'un certificat à utiliser avec WAPT

Dans la console WAPT, aller dans *Outils* → *Générer un agent WAPT*.

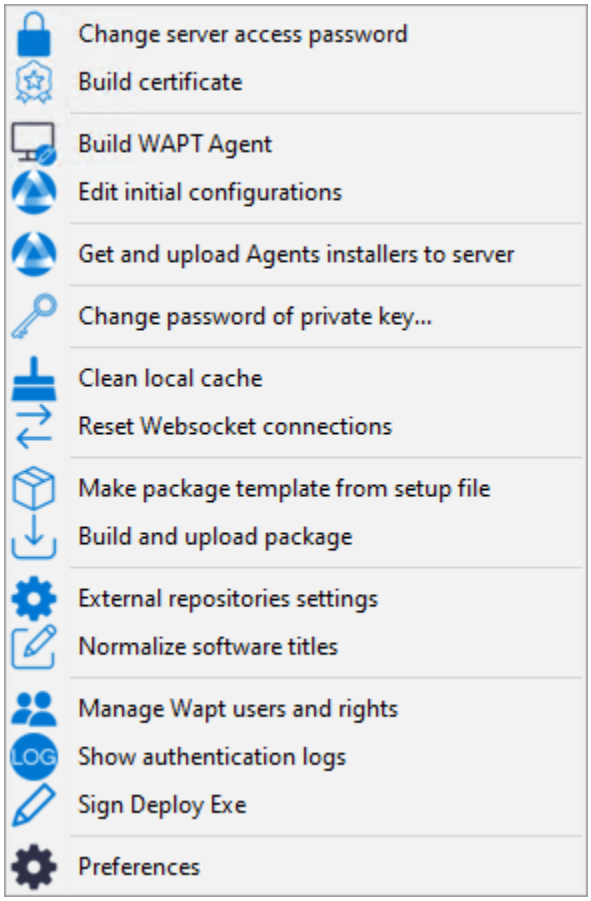


Figure1: Création d'un certificat auto-signé

Avec WAPT Enterprise, vous pouvez créer une clé principale avec une propriété d'Autorité de Certification qui peut à la fois signer les paquets WAPT et signer les nouveaux certificats à utiliser avec WAPT.

Afin de créer de nouveaux certificats signés pour les utilisateurs délégués, veuillez vous référer à *créer un nouveau certificat*.

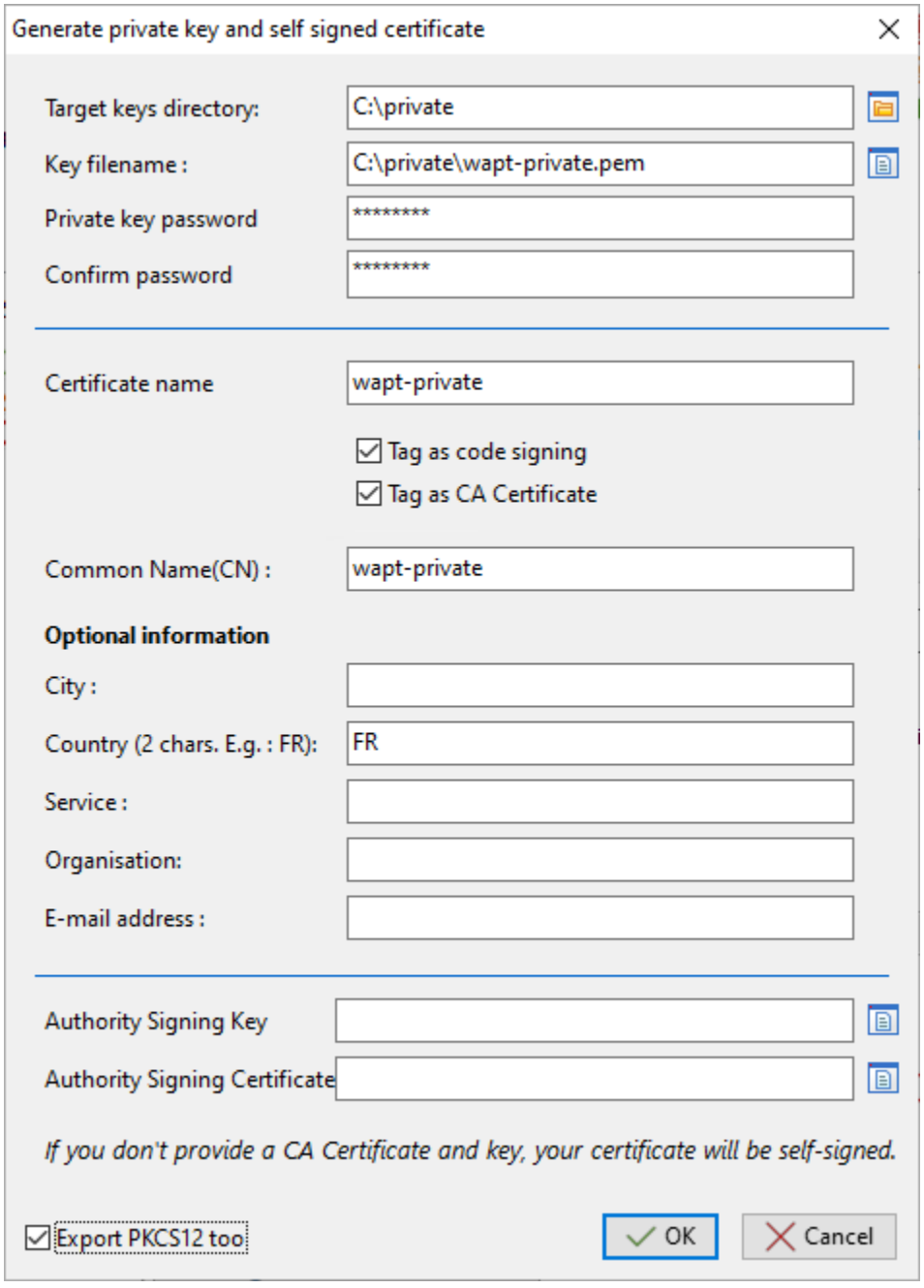


Figure2: Création d'un certificat auto-signé pour la version WAPT Enterprise

Table1: Affectation du certificat

Valeur	Description	Requis	Enterprise
Organisation	Dossier du certificat de confiance.	✓	
Nom du fichier clé	Définit le nom de la clé privée `.pem`.	✓	
Organisation	Fournissez le mot de passe pour déverrouiller la clé privée.	✓	
Confirmer le mot de passe	Fournissez le mot de passe pour déverrouiller la clé privée.	✓	
Liste de certificats autorisés	Dossier du certificat de confiance.	✓	
Organisation	Définit si le certificat/la paire de clés sera autorisé(e) à signer les packages logiciels.	✓	+
Vérifier le certificat https serveur	Définit si le certificat peut être utilisé pour signer d'autres certificats (autorité de certification principale ou intermédiaire).	✓	+
Nom commun (NC)	Dossier du certificat de confiance.	✓	
Organisation	Chemin d'accès aux certificats utilisés pour la vérification HTTPS.	✗	
Pays (2 chars. Ex : FR)	Définit le nom du pays du titulaire du certificat (FR, EN, ES, DE . . .) à enregistrer dans le certificat.	✗	
Organisation	Définit le nom du service ou du département organisationnel du titulaire du certificat à enregistrer dans le certificat.	✗	
Organisation	Nom de l'organisation permettant d'identifier l'origine des paquets WAPT.	✗	
Adresse du serveur WAPT	Définit l'adresse e-mail du titulaire du certificat à enregistrer dans le certificat.	✗	
Clé de signature de l'autorité	Définit la clé (`.pem`) de la CA.	✗	+
Liste de certificats autorisés	Définit le certificat (`.crt`) de la CA.	✗	+
Exporter PKCS12	Force la création du certicat `*.p12` dans le répertoire *Targets keys*	✗ (recommandé)	

Des détails supplémentaires sont stockés dans la clé privée. Ces informations permettront d'identifier l'origine du certificat et l'origine du paquet WAPT.

La complexité du mot de passe **Doit**se conformer aux exigences de sécurité de votre *Organisation* (consultez le site ANSSI pour des recommandations sur les mots de passe).

Danger:

- Le fichier `wapt-private.pem` ne doit pas être stocké sur le serveur WAPT.

- Cliquez sur *OK* pour passer à l'étape suivante.

  Si tout s'est bien passé, le message suivant apparaît :

- Sélectionner *Oui*.
- Cliquez sur *Yes* pour copier le certificat nouvellement généré dans le dossier `C:\Program Files (x86)\wapt\ssl` sous Windows ou `/opt/wapt/ssl` sous Linux ou macOS. Ce certificat sera récupéré lors de la compilation de l'agent WAPT et déployé sur les ordinateurs clients.

Vous pouvez passer à l'étape suivante et *construire le programme d'installation de l'agent WAPT*.

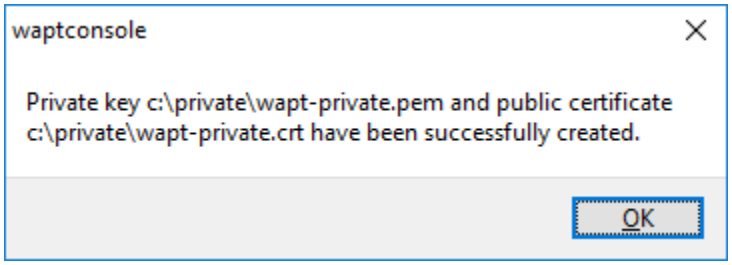


Figure3: Le certificat a été généré avec succès

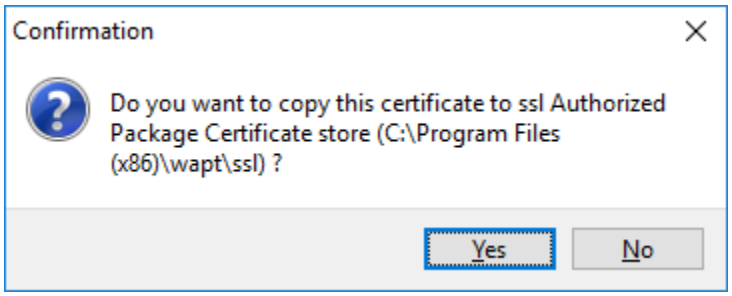


Figure4: Boîte de dialogue demandant la confirmation de la copie du certificat dans le dossier ssl de la console WAPT

CHAPTER 18

Générer l'agent

Le binaire **waptagent** est un installateur InnoSetup.

Une fois que la console WAPT a été installée sur l'ordinateur *Administrator*, nous avons tous les fichiers nécessaires pour construire le programme d'installation de l'agent WAPT :

Avant de construire l'agent WAPT, vérifier que vos certificats sont prêts. Si vous souhaitez déployer d'autres certificats publics sur les ordinateurs de votre *Organisation* équipés de WAPT, vous devrez les copier dans un dossier commun puis sélectionner le dossier lors de la génération de l'Agent WAPT.

Dans l'ancienne méthode de construction de waptagent, c'était assez dangereux car vous pouviez **COPIER la clé privée** de n'importe quel *Administrateur* dans `C:\Program Files (x86)\wapt`. Cela signifie que, par erreur, la clé privée serait déployée sur tous les ordinateurs du parc, ce qui constituerait un grave **risque de sécurité**.

Avant 2.3.0, ce dossier était utilisé lors de la construction de l'agent WAPT et les clés privées seraient ensuite déployées sur tous les ordinateurs.

Maintenant, la nouvelle méthode est beaucoup plus sûre :

La méthode utilise un waptsetup qui est signé par Tranquil IT, nous le copions et nous poussons la configuration dans un fichier *json*. Alternativement, nous pouvons aussi *créer un paquet de configuration WAPT* qui sera appelé lors du déploiement de l'agent WAPT. Nous appelons cette méthode certificate stuffing.

En plus d'éviter les erreurs, comme le déploiement d'un certificat privé par erreur, la méthode a l'avantage de ne plus nécessiter la construction d'un agent WAPT personnalisé. Cette méthode permet également d'éviter de nombreux problèmes d'antivirus avec les faux positifs.

Lorsque l'agent WAPT sera installé silencieusement, il prendra la configuration **par défaut** : il construira le fichier de configuration `wapt-get.ini` de l'agent WAPT et extraira les certificats dans `wapt/ssl`.

Pour sécuriser cette installation (par exemple avec des GPO), **waptsetup.exe** et sa configuration intégrée *json* ont le nom et le hachage du nom de la configuration sur le serveur WAPT. Lorsque le programme d'installation applique la configuration *json*, il vérifie au préalable avec ce hachage que les données *json* n'ont pas été modifiées.

- Dans la console WAPT, allez dans *Outils* → *Générer un agent WAPT*.

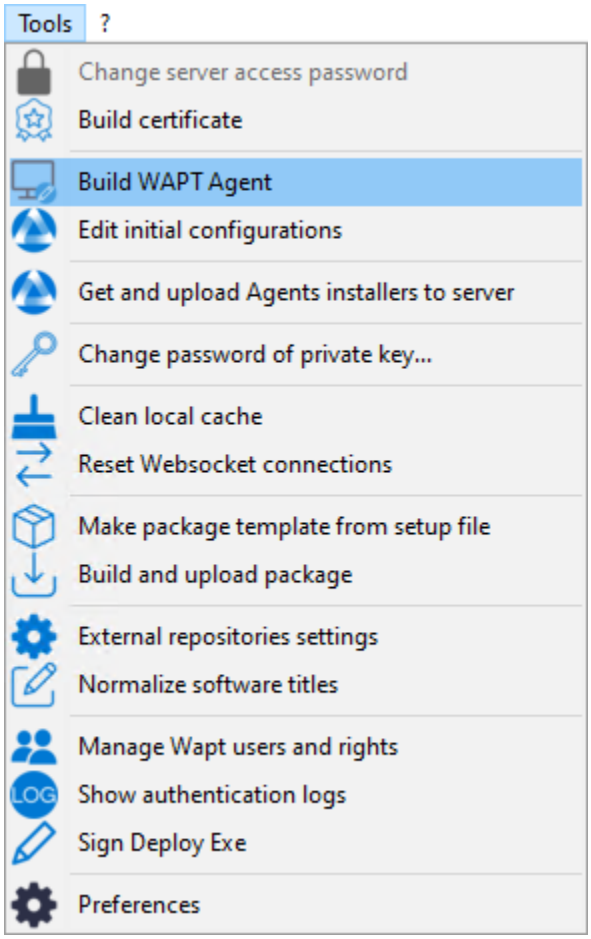


Figure1: Générer l'agent WAPT depuis la console

Avant de construire l'agent WAPT, vous devez choisir comment il s'identifiera auprès du serveur WAPT.

18.1 Choix du mode d'identification unique des agents WAPT

Dans WAPT, vous pouvez choisir le mode d'identification unique des agents WAPT.

Lorsqu'un agent WAPT s'enregistre, le serveur doit savoir s'il s'agit d'une nouvelle machine ou d'une machine qui a déjà été enregistrée.

Pour cela, le serveur WAPT examine l"UUID (Universal Unique IDentifier) de l'inventaire.

WAPT propose 3 modes pour vous aider à distinguer les machines, à vous de choisir le mode qui vous convient le mieux.

Attention: Après avoir choisi un mode de fonctionnement, il est difficile d'en changer, réfléchissez bien !

UUID du BIOS (numéro de série)

Ce mode de fonctionnement permet d'identifier les machines de la console de manière physique.

Si vous remplacez un ordinateur et donnez au nouvel ordinateur le même nom que le précédent, vous aurez deux ordinateurs qui apparaîtront dans la console WAPT puisque vous aurez physiquement deux ordinateurs différents.

Note: Certains fournisseurs font un travail inadéquat et attribuent les mêmes UUID de BIOS à des lots entiers d'ordinateurs. Dans ce cas, WAPT ne verra qu'un seul ordinateur !!!

nom d'hôte

Ce mode de fonctionnement est similaire à celui d'Active Directory. Les machines sont identifiées par leur nom d'hôte.

Note: Ce mode ne fonctionne pas si plusieurs machines du parc portent le même nom.

Nous savons tous que cela ne devrait jamais arriver.

uUID généré aléatoirement

Ce mode de fonctionnement permet d'identifier les PC par leur installation WAPT. Chaque installation de WAPT génère un numéro aléatoire unique. Si vous désinstallez WAPT puis le réinstallez, vous verrez apparaître un nouveau périphérique dans votre console.

Note: Dans ce mode, les UUIDs ont le préfixe RMD

18.2 Construire

- Dans la console WAPT, allez dans *Outils* → *Générer un agent WAPT*

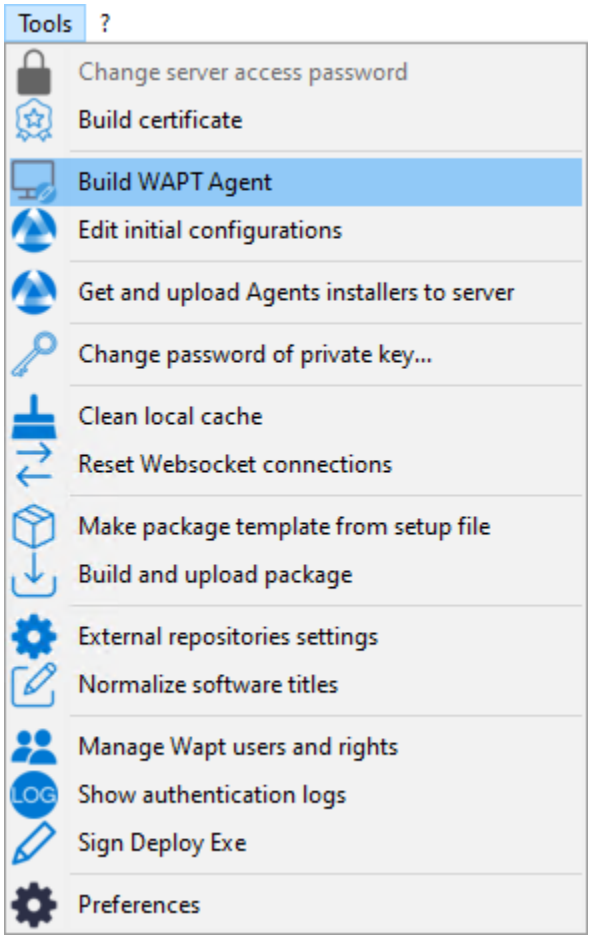


Figure2: Générer l'agent WAPT depuis la console

- Remplissez les informations qui sont nécessaires pour l'installateur.

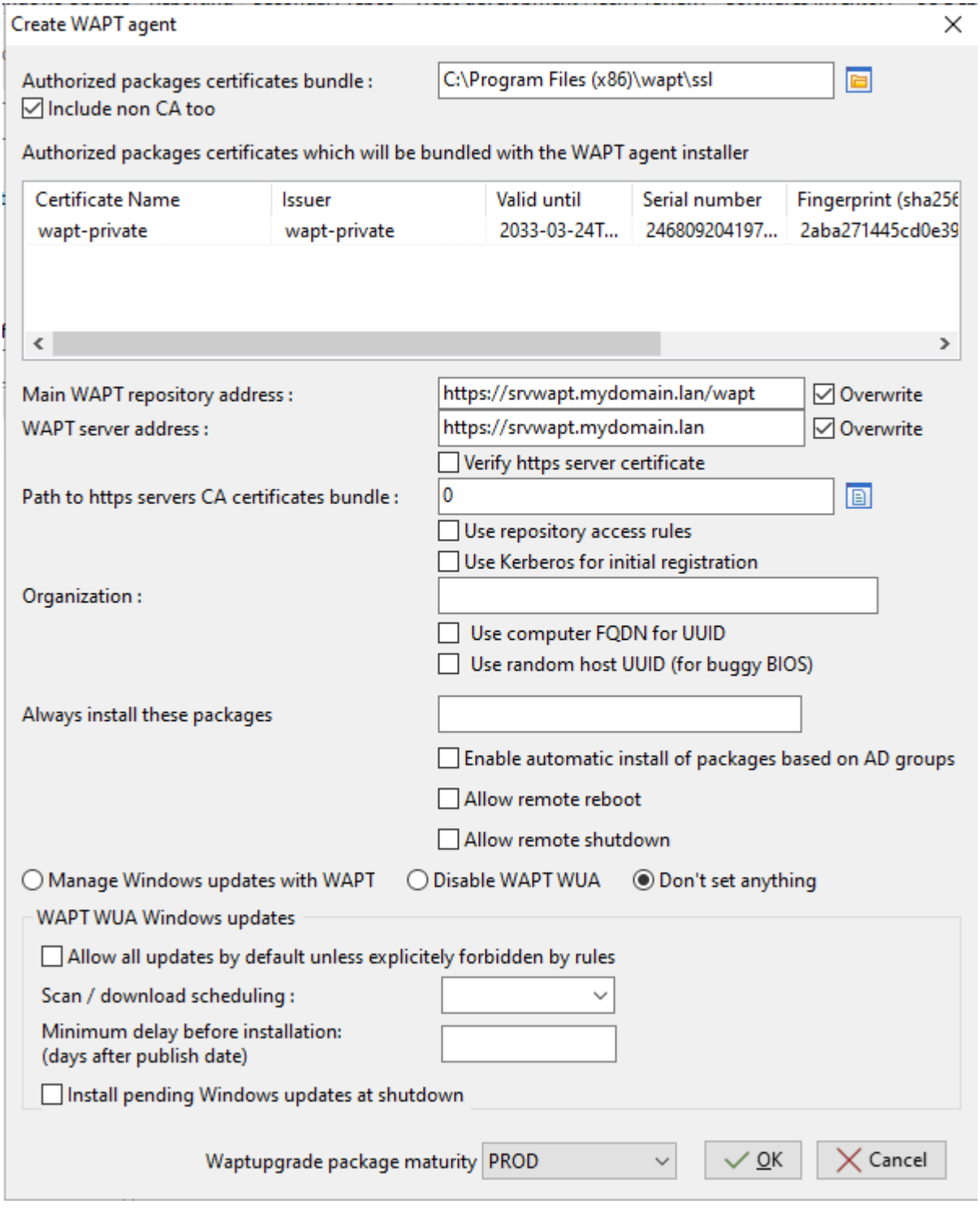


Figure3: Remplir les informations sur votre organisation

Table1: Affectation du certificat

Valeur	Description	Requis	Enterprise
Liste de certificats autorisés	Dossier du certificat de confiance.	✓	
Liste de certificats autorisés	Définit si le certificat WAPT local doit être inclus.	✗	
Adresse du serveur WAPT	Définit l'URL du référentiel WAPT principal.	✓	
Adresse du serveur WAPT	Définit l'URL du serveur WAPT.	✓	
Vérifier le certificat https serveur	Définit si l'authentification client du certificat *HTTPS* est activée sur le serveur WAPT.	✗	
Utiliser les règles d'accès au référentiel	Définit si les règles d'accès aux référentiels doivent être utilisées pour *répliquer les référentiels distants*.	✗	+
Chemin vers le paquet de certificats CA des serveurs https WAPT	Définit le chemin vers les certificats utilisés pour la vérification HTTPS.	✗	
Utiliser Kerberos pour l'enregistrement initial	Définit si l'authentification *Kerberos* des agents WAPT doit être utilisée avec le serveur WAPT.	✗	
Organisation	Définit le nom de l'organisation pour identifier l'origine des packages WAPT.	✗	
Utiliser le FQDN de l'ordinateur comme UUID	Définit si les FQDN doivent être utilisés pour *identifier les agents WAPT*.	✗	
Utiliser un UUID hôte aléatoire (pour les BIOS bogués)	Définit si des UUID aléatoires doivent être utilisés pour *identifier les agents WAPT*.	✗	
Il faut toujours installer ces packages	Définit s'il faut installer automatiquement *les packages du groupe* lors de l'installation de l'agent WAPT.	✗	+
Autoriser l'installation automatique de packages basés sur les groupes AD	Permet l'installation des *packages de profil*. **Cette fonctionnalité peut dégrader les performances de WAPT**.	✗	+
Organisation	Définit si les redémarrages à distance sont autorisés à partir de la console WAPT.	✗	+
Adresse du serveur WAPT	Définit si les arrêts à distance sont autorisés à partir de la console WAPT.	✗	+
Gérer les mises à jour de Windows avec WAPT | Désactiver WAPT WUA | Ne rien définir	Active ou désactive *WAPT WUA*.	✓	+
Autoriser toutes les mises à jour par défaut, sauf si elles sont explicitement interdites par les règles	Définit s'il faut autoriser toutes les mises à jour de Windows si elles ne sont pas interdites par des packages de règles WUA.	✗	+
Organisation	Définit la périodicité de l'analyse de Windows Update.	✗	+
Délai minimum avant installation (jours après la date de publication)	Définit un délai d'installation différée avant la publication.	✗	+
Installer les mises à jour Windows en attente à l'arrêt	Force les mises à jour à s'installer lorsque l'hôte s'éteint.	✗	+
Maturité du packaging de Waptupgrade	Permet de choisir la maturité du paquet waptupgrade.	✗	+

Pour plus d'informations sur la section Windows update, consultez *cette article sur la configuration de WAPTWUA sur l'agent WAPT*

Danger:

- La case à cocher *Utiliser kerberos pour l'enregistrement initial* doit être cochée **UNIQUEMENT SI** vous avez suivi la documentation pour *configurer l'authentification kerberos*.
- La case à cocher *Vérifier le certificat HTTPS du serveur WAPT* doit être cochée **SEULEMENT SI** vous avez suivi la documentation pour *activer la vérification du certificat SSL / TLS*.

- Fournissez le mot de passe pour déverrouiller la clé privée.

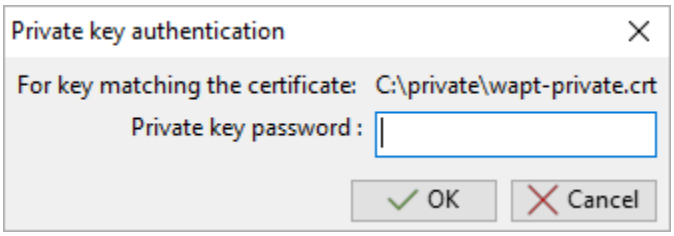


Figure4: Fournir le mot de passe pour déverrouiller la clé privée

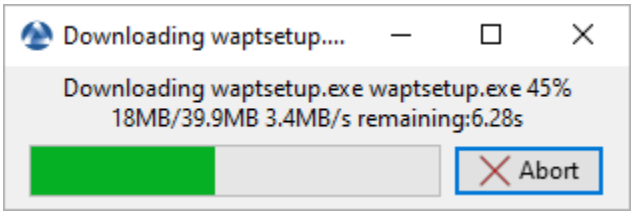


Figure5: Progression de l'installation de l'agent WAPT

Une fois que le programme d'installation de l'agent WAPT a terminé sa construction, une boîte de dialogue de confirmation apparaît pour indiquer que le binaire **`waptagent`** a été téléchargé avec succès sur https://srvwapt.mydomain.lan/wapt/.

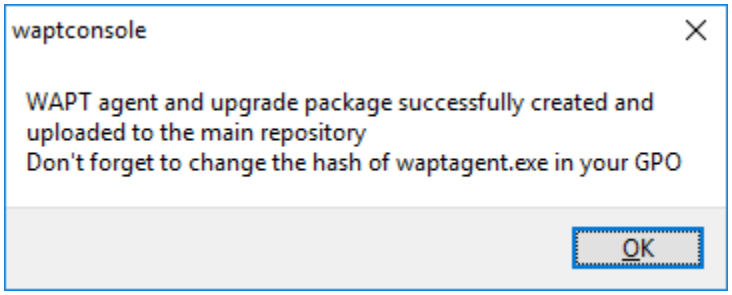


Figure6: Confirmation du chargement de l'agent WAPT sur le référentiel WAPT

Un avertissement s'affiche indiquant que la valeur de hachage de la GPO doit être modifiée. Les GPO peuvent être utilisés pour déployer l'agent WAPT sur l'ordinateur de votre Organisation.

Attention: Après avoir construit l'agent sur votre ordinateur de gestion, quittez la console WAPT et *installer* le **nouvel agent WAPT** qui a été généré sur votre ordinateur de gestion WAPT.

CHAPTER 19

Configuration initiale

Il est possible de configurer l'Agent WAPT avec des options standard et avancées via une interface graphique. Très similaire à *créer un package de configuration*, nous vous recommandons **fortement** de voir la section au préalable. La configuration initiale vise à configurer des paramètres importants dans l'agent WAPT, qu'il s'agisse de Windows, Linux ou macOS. La méthode est très utile pour installer un *Agent WAPT sur Linux ou macOS*.

- Dans la console WAPT, allez dans *Outils* → *Générer un agent WAPT*
- Remplir les informations qui sont nécessaires pour la configuration

Table1: En-tête

Valeur	Description
Organisation	Affiche *les options de configuration de l'Agent WAPT* comme dans `wapt-get.ini`.
Vérifier le certificat https serveur	Ajoute le certificat à la configuration.
Adresse du serveur WAPT	Charge une configuration précédemment créée.
Vérifier le certificat https serveur	Rafraîchit la liste des configurations disponibles.
Organisation	Crée une nouvelle configuration.
Organisation	Supprime une configuration.

mondial

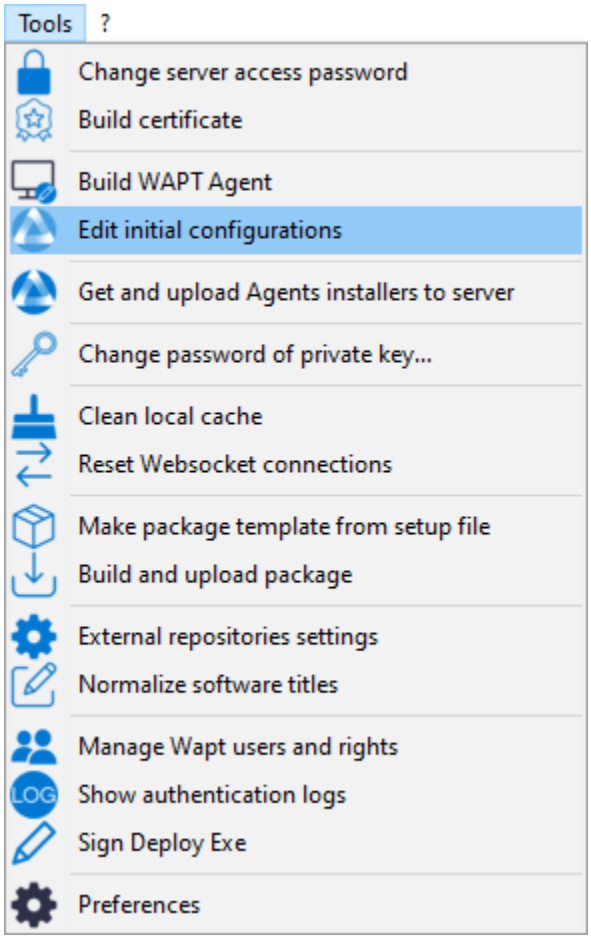


Figure1: Création de la configuration initiale

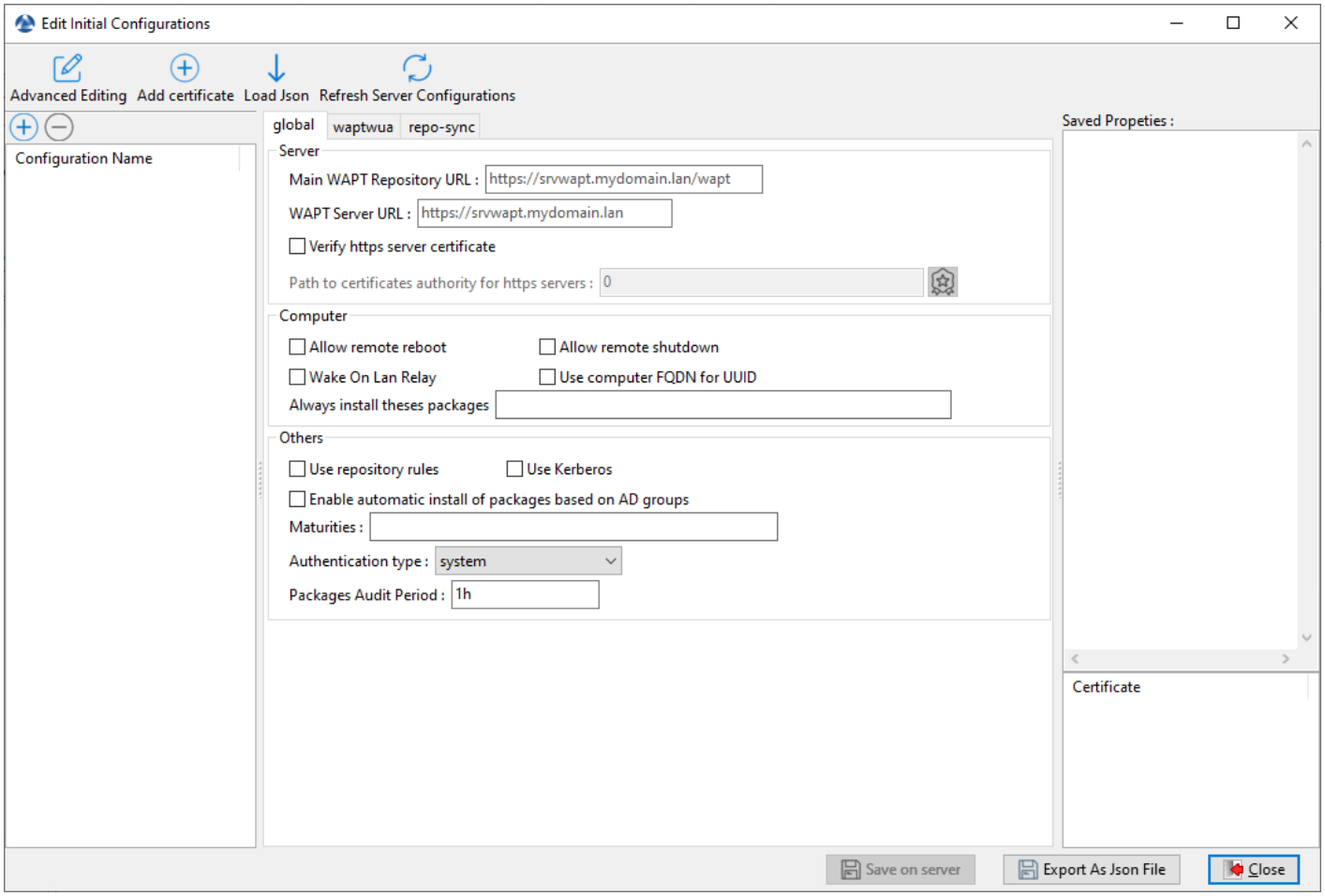


Figure2: Éditer la configuration initiale

Valeur	Description	Requis	Enterprise
Adresse du serveur WAPT	Définit l'URL du référentiel WAPT principal.	✓	
Organisation	Définit l'URL du serveur WAPT.	✓	
Vérifier le certificat https serveur	Définit si l'authentification client du certificat *HTTPS* est activée sur le serveur WAPT.	✗	
Chemin vers l'autorité de certification pour les serveurs https	Définit le chemin vers les certificats utilisés pour la vérification HTTPS.	✗	
Organisation	Définit si les redémarrages à distance sont autorisés à partir de la console WAPT.	✗	+
Adresse du serveur WAPT	Définit si les arrêts à distance sont autorisés à partir de la console WAPT.	✗	+
Organisation	Active la fonctionnalité WoL (Wake-on-Lan) sur les référentiels secondaires.	✗	+
Utiliser le FQDN de l'ordinateur comme UUID	Définit si les FQDN doivent être utilisés pour *identifier les agents WAPT*.	✗	
Il faut toujours installer ces packages	Définit s'il faut installer automatiquement *les packages du groupe* lors de l'installation de l'agent WAPT.	✗	+
Adresse du serveur WAPT	Définit si les dépôts *sont répliqués*.	✗	+
Organisation	Définit si l'authentification *Kerberos* des agents WAPT doit être utilisée avec le serveur WAPT.	✗	
Autoriser l'installation automatique de packages basés sur les groupes AD	Permet l'installation des *packages de profil*. **Cette fonctionnalité peut dégrader les performances de WAPT**.	✗	+
Organisation	Liste des maturités de packaging qui peuvent être visualisées et installées par l'agent WAPT. La valeur par défaut est `PROD`. Seules les valeurs `DEV`, `PREPROD` et `PROD` sont utilisées par Tranquil IT, cependant toute valeur peut être utilisée pour s'adapter à vos processus internes.	✗	
Vérifier le certificat https serveur	Définit le mode de fonctionnement de l'authentification en libre-service. Les valeurs possibles sont : `system`, `waptserver-ldap` ou `waptagent-ldap`.	✓	
Organisation	Définit la fréquence à laquelle les audits sont déclenchés.	✓	

waptwua

Valeur	Description	Requis
Gérer les mises à jour de Windows avec WAPT	Active ou désactive *WAPT WUA*.	✓
Autoriser toutes les mises à jour par défaut, sauf si elles sont explicitement interdites par les règles	Définit s'il faut autoriser toutes les mises à jour de Windows si elles ne sont pas interdites par des packages de règles WUA.	✗
Organisation	Définit une liste de gravité qui sera automatiquement acceptée pendant un scan WAPT de mise à jour de Windows. ex : *Important*, *Critique*, *Modéré*.	✗
Télécharger les mises à jour depuis les serveurs Microsoft	Définit si les mises à jour sont téléchargées directement à partir des serveurs Microsoft.	✗
Organisation	Définit la récurrence de l'analyse de Windows Update (ne fera rien si la règle du paquet *waptwua* ou le fichier `wsusscn2.cab` n'ont pas changé).	✗
Installer les mises à jour Windows en attente à l'arrêt	Force les mises à jour à s'installer lorsque l'hôte s'éteint.	✗
Organisation	Définit la récurrence de l'installation de Windows Update (ne fera rien si aucune mise à jour n'est en attente).	✗
Délai minimum avant installation (jours après la date de publication)	Définit un délai d'installation différée avant la publication.	✗

repo-sync

Attention: Ces options ne doivent être utilisées que sur un référentiel secondaire.

Valeur	Description	Requis
Organisation	Permet au serveur WAPT de servir de référentiel.	✓
Adresse du serveur WAPT	Définit les dossiers à synchroniser	✓
Synchroniser uniquement lorsque cela est demandé	Activer ou désactiver la synchronisation automatique	✗
Organisation	Définit la périodicité de la synchronisation	✓
Heure locale du référentiel pour le début de la synchronisation	Définit l'heure de début de la synchronisation (HH:MM / format 24h)	✗
Heure locale du référentiel pour la fin de la synchronisation	Définit l'arrêt du début de la synchronisation (HH:MM / format 24h)	✗

Table2: Colonne

Valeur	Description
Organisation	Liste des *options* avec la configuration.
Vérifier le certificat https serveur	Liste des certificat avec la configuration.

Table3: Pied de page

Valeur	Description
Organisation	Fenêtre pour la configuration basique de la console WAPT
Adresse du serveur WAPT	Exporter la configuration en JSON
Organisation	Fermer la fenêtre

- Après la configuration, il est possible de copier les commandes en cliquant avec le bouton droit de la souris sur la configuration

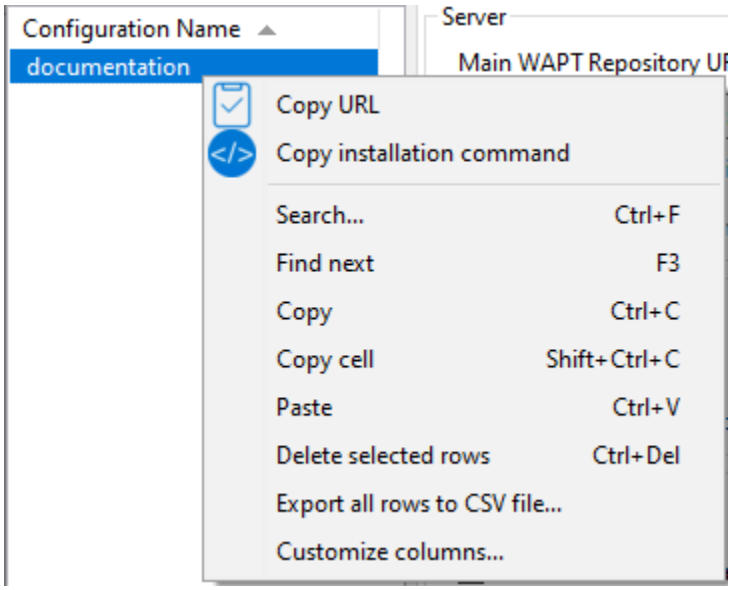


Figure3: Commande de copie

Table4: Options de copie

Valeur	Description
Organisation	Donne une URL de téléchargement pour récupérer le `.json` du serveur.
Organisation	Donne une commande pour installer la configuration d'un agent WAPT.

Note: Il est possible d'installer un agent vierge et de lui donner la commande d'installation copiée pour fournir la configuration.

Cette section de la documentation couvre l'utilisation quotidienne de WAPT.

Toutes les fonctionnalités de WAPT sont expliquées en détail pour les *Administrateurs*, les *Utilisateurs* et les Déployeurs de packages.

CHAPTER 20

Gestion de l'agent WAPT sous Windows

20.1 Déploiement de l'agent WAPT sur Windows

Note: Pour installer WAPT sur un client Windows, la configuration minimale requise est la suivante :

- 512Mo Ram;
- 1 CPU;
- 300Mo d'espace disque (sans tenir compte de l'espace de cache pour les paquets WAPT).

Attention: Si vous installez l'Agent WAPT sur **Windows Server 2012r2**, ces fonctionnalités doivent être activées avant d'installer l'agent WAPT :

- `KB2919442`.
- `KB2919355`.
- `vcredist2015`

Deux méthodes sont disponibles pour déployer le **`waptagent.exe`**.

- La première méthode est manuelle et la procédure **Doit** être appliquée sur chaque hôte.
- La seconde est automatisée et s'appuie sur un GPO.

Note: Le programme d'installation **`waptagent.exe`** est disponible sur la page d'accueil du site WAPT serveur. Le lien de téléchargement direct est par exemple : https://srvwapt.mydomain.lan/wapt/waptagent.exe.

Avertissement: Si vous ne signez pas le programme d'installation **`waptagent.exe`** avec un certificat commercial **Code Signing** ou un certificat **Code Signing** émis par l'*Autorité de Certification* de votre Organisation, les navigateurs web afficheront un message d'avertissement lors du téléchargement du programme d'installation.

Pour supprimer le message d'avertissement, vous **DEVEZ**signer le :mimetype:`.exe` avec un certificat **Code Signing** qui peut être vérifié par un certificat d'autorité stocké dans le magasin de certificats de la machine.

20.1.1 Manuellement

Attention: L'installation manuelle de l'agent WAPT nécessite des droits d'administrateur local sur l'ordinateur.

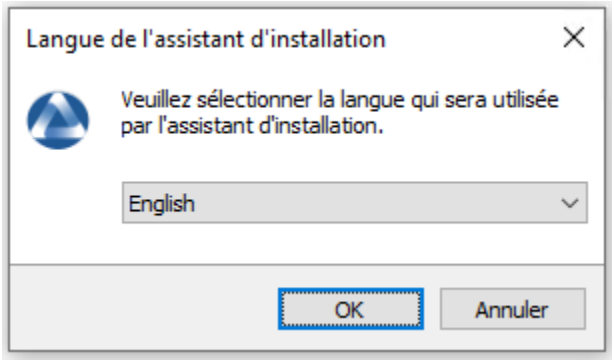
L'installation manuelle de l'agent WAPT à l'aide d'un compte d'administrateur de domaine ne fonctionnera pas.

Indication: Quand déployer l'agent WAPT manuellement ?

La méthode de déploiement manuel est efficace dans ces cas :

- Test WAPT.
- Utilisation de WAPT dans une organisation avec un petit nombre d'ordinateurs.
- Si vous ne disposez pas d'un moyen de déploiement de masse.

- Téléchargez l'agent WAPT depuis votre serveur WAPT puis lancez le programme d'installation.
- Choisissez la langue du programme d'installation de WAPT.



- Cliquez sur *OK* pour passer à l'étape suivante.

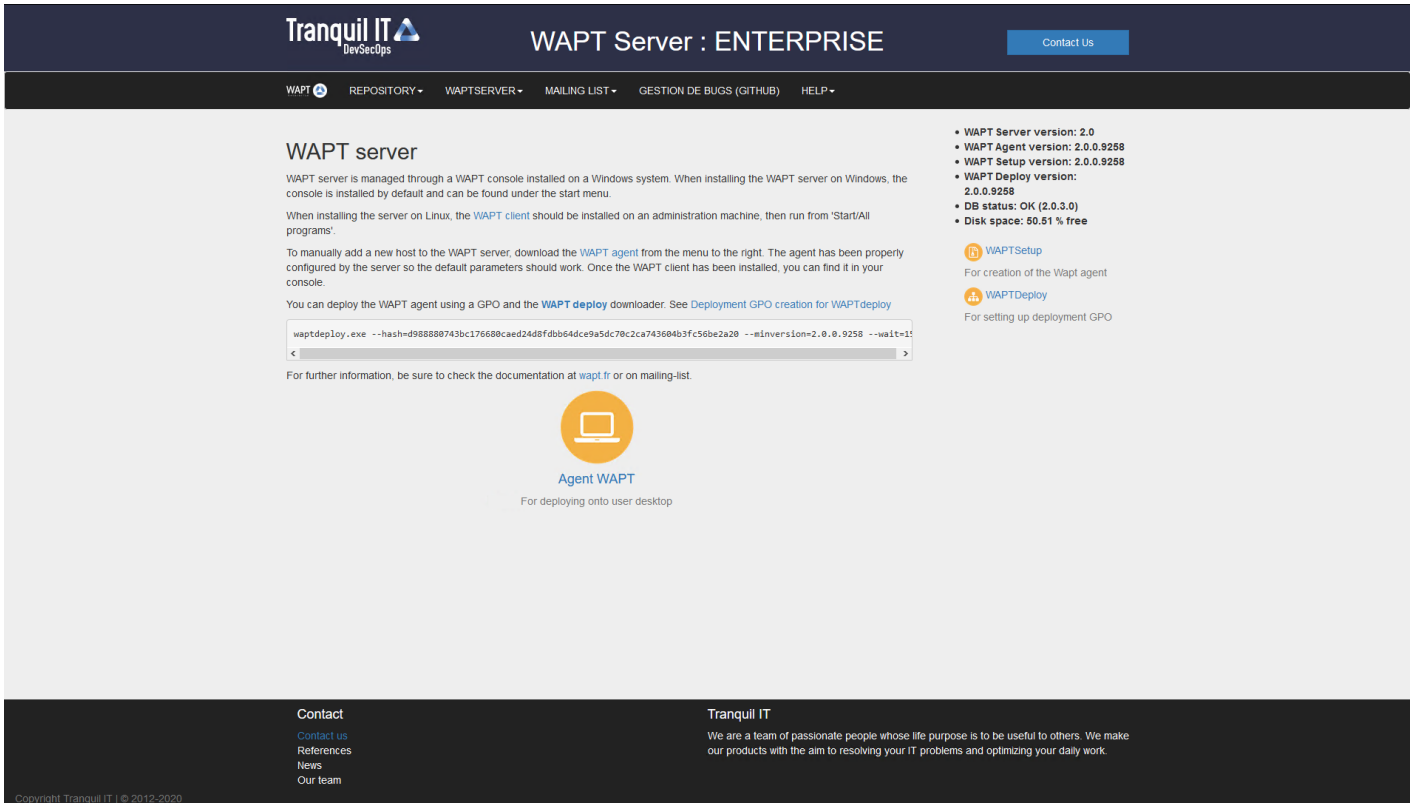
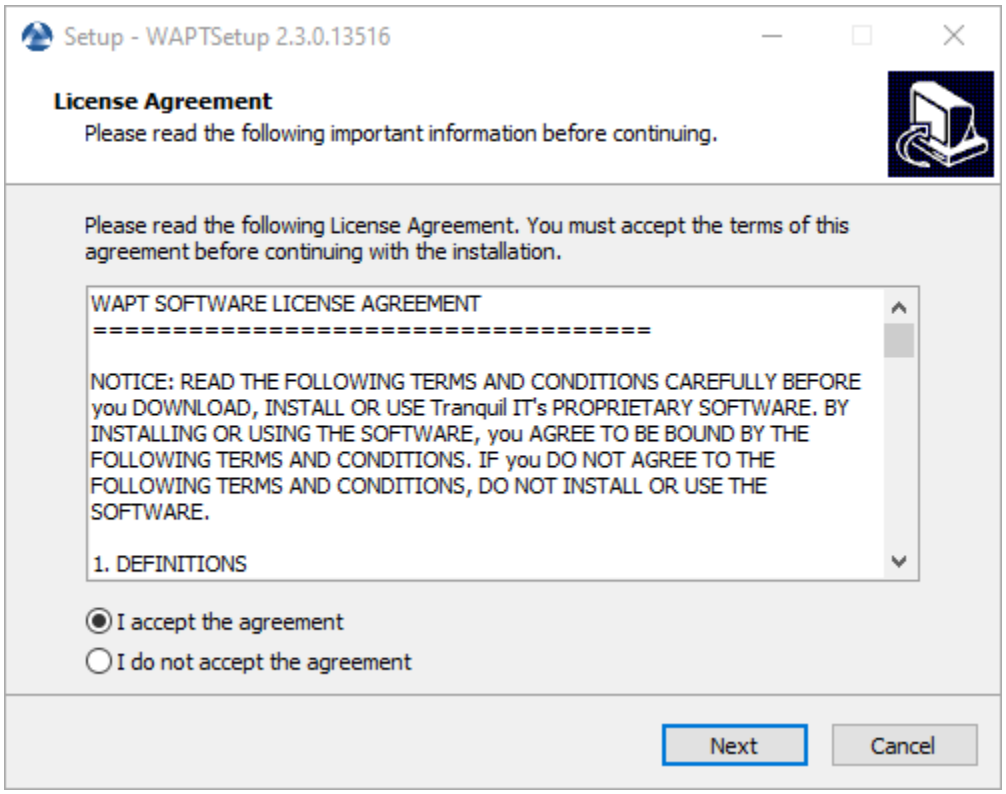


Figure1: L'interface du serveur WAPT dans un navigateur web



- Acceptez les conditions de la licence et cliquez sur *Next* pour passer à l'étape suivante.
- Choisissez des tâches de configuration supplémentaires (laissez la valeur par défaut si vous n'êtes pas sûr).

Table1: Options disponibles

Paramètres	Description	Valeur par défaut
Installer le service WAPT	Ajoute le service WAPT sur l'ordinateur.	Vérifié
L'icône de notification de lancement à l'ouverture de la session	Lance l'agent WAPT dans la barre d'état système au démarrage.	Non vérifié
Désactiver hiberboot, et augmenter le délai d'arrêt de la GPO (recommandé)	Désactive le démarrage rapide de Windows pour des raisons de stabilité, augmente le délai d'attente pour l'utilitaire WAPT Exit.	Vérifié
Utiliser un UUID aléatoire pour identifier l'ordinateur au lieu du BIOS	Résout les éventuels bogues *BIOS UUID*.	Non vérifié

- Choisissez le référentiel WAPT et le serveur WAPT et cliquez sur *Next* pour passer à l'étape suivante.
- Installez l'agent WAPT en cliquant sur *Install*.

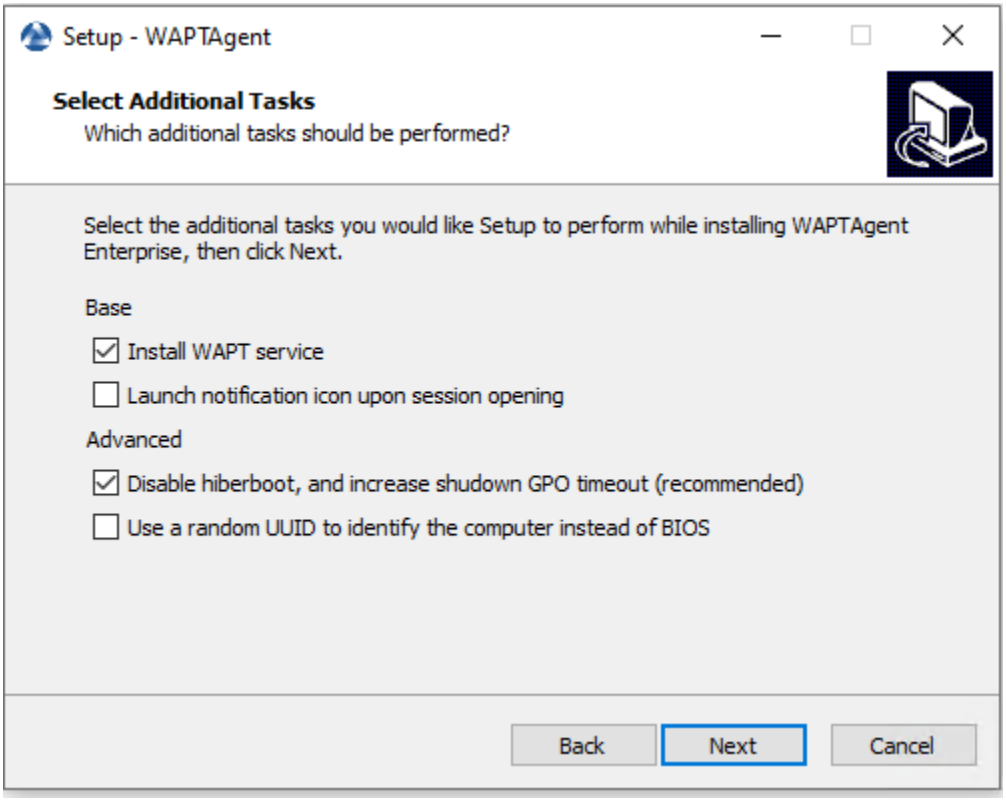


Figure2: Choix des options du programme d'installation pour le déploiement de l'agent WAPT

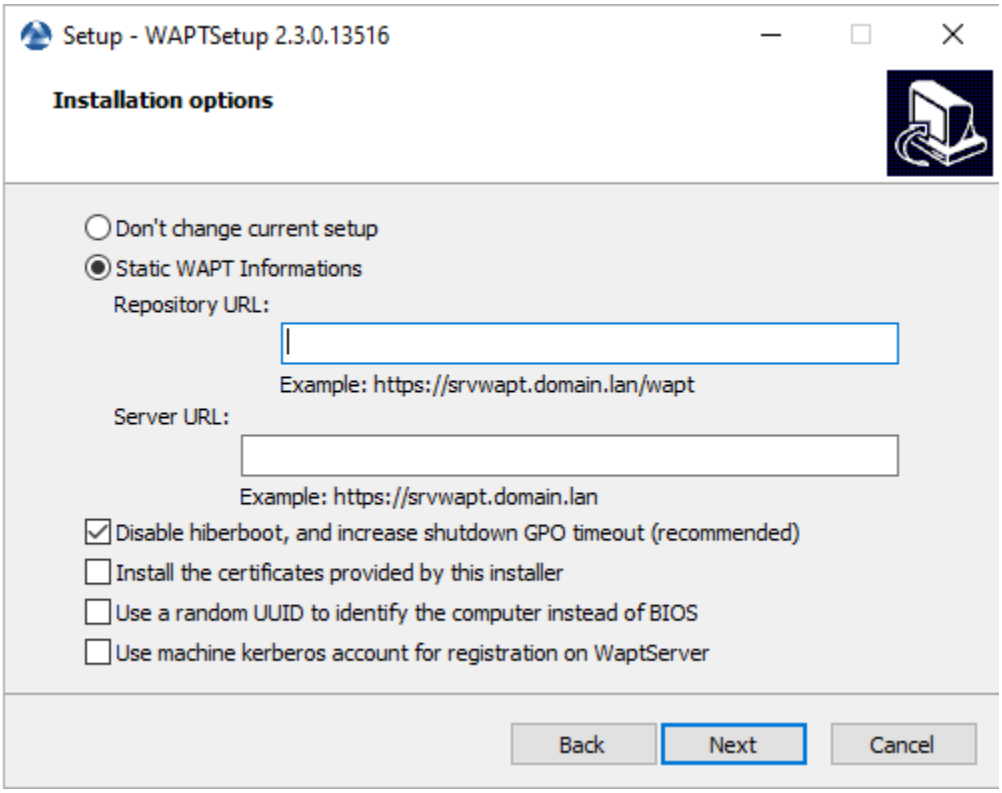
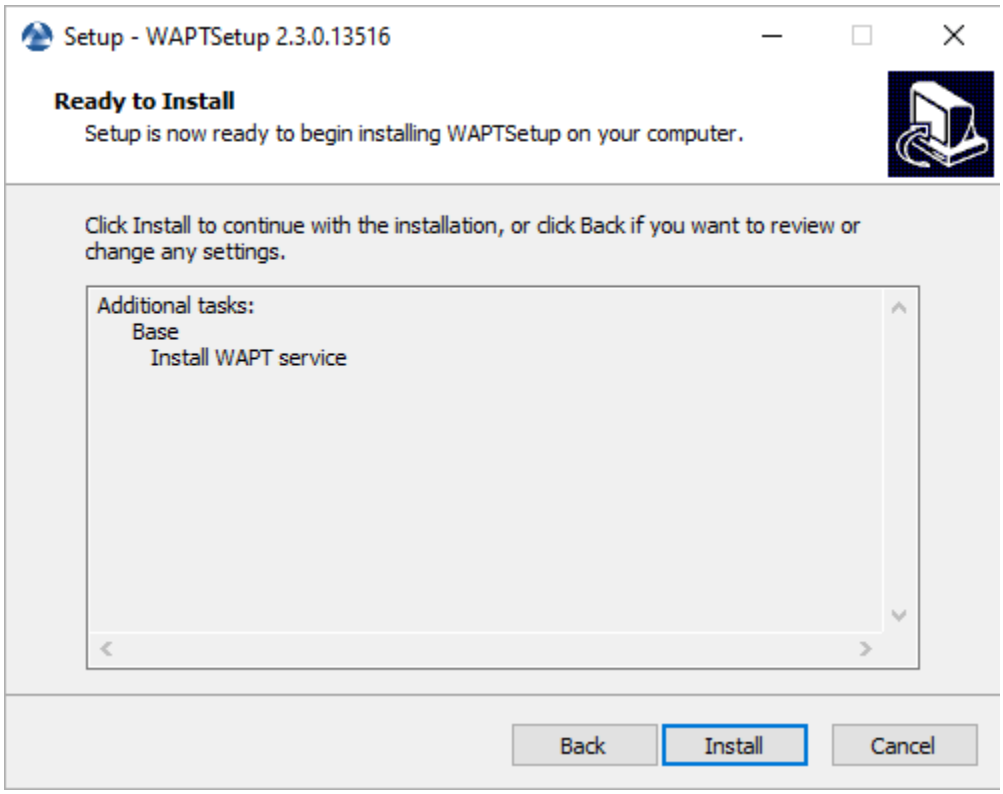
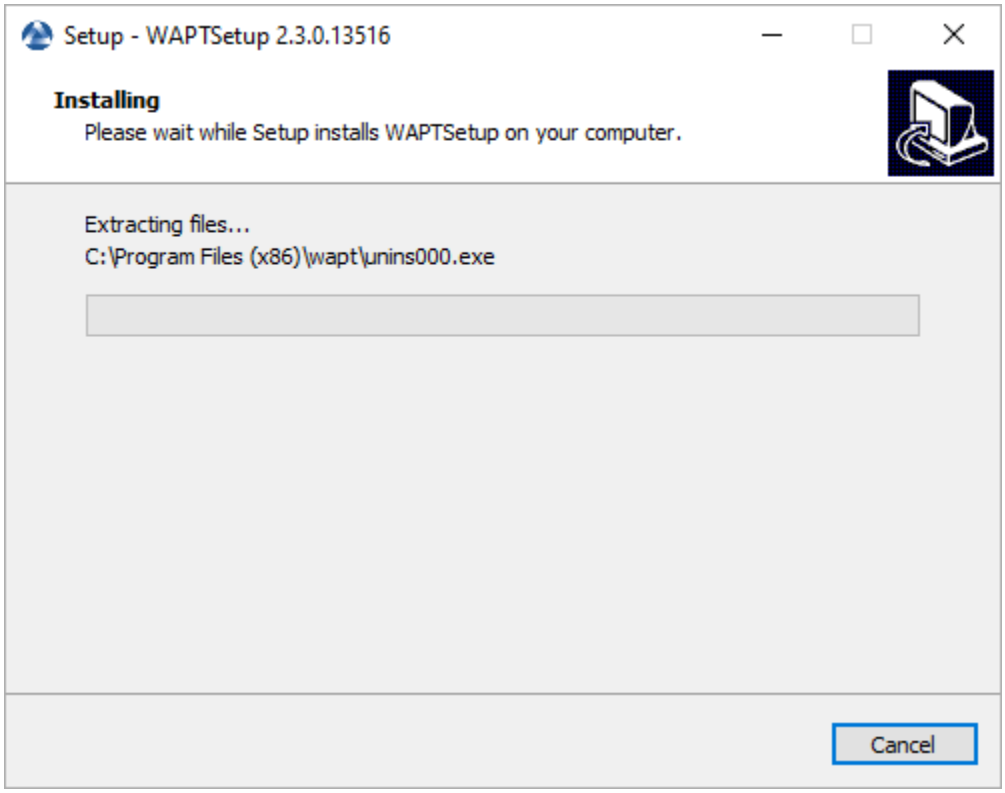


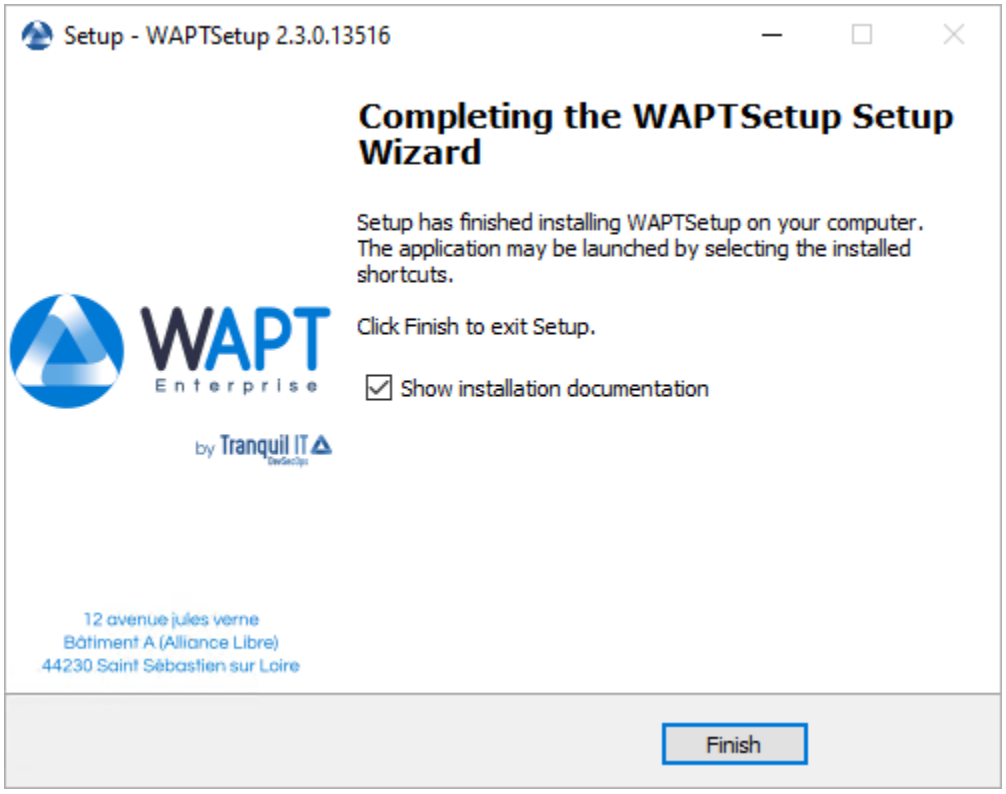
Figure3: Choisir le référentiel WAPT et le serveur WAPT



- Attendez que l'installation de l'agent WAPT se termine, puis cliquez sur *Finish* pour quitter.



L'installation de l'agent WAPT est terminée. L'enregistrement de l'hôte avec le serveur WAPT se fait automatiquement.



Pour gérer les clients WAPT de votre organisation, consultez la *documentation sur l'utilisation de la console WAPT*.

20.1.2 Automatiquement

Important: Pré-requis techniques

Des connaissances avancées en matière d'administration de réseaux et de systèmes sont nécessaires pour mener à bien cette procédure. Un réseau correctement configuré en assurera le succès.

Indication: Quand déployer automatiquement l'agent WAPT ?

La méthode suivante est utile dans ces cas :

- Une grande organisation avec de nombreux ordinateurs.
- Un Samba Active Directory ou un Microsoft Active Directory pour lequel vous disposez de suffisamment de privilèges d'administration.
- La sécurité et la traçabilité des actions sont importantes pour vous ou pour votre *Organisation*.

Avec l'utilitaire WAPT Deployment

waptagent.exe est un installateur InnoSetup, il peut être exécuté avec ces arguments silencieux :

```
waptagent.exe /VERYSILENT
```

- Des arguments supplémentaires sont disponibles pour l'utilitaire WAPT Deployment.

Table2: Description des options disponibles pour le déploiement silencieux de l'agent WAPT

Options	Description
/dnsdomain = mydomain.lan	Domaine dans wapt-get.ini rempli lors de l'installation.
/wapt_server = https://srvwapt.mydomain.lan	URL du serveur WAPT dans wapt-get.ini rempli lors de l'installation.
/repo_url = https://repo1.mydomain.lan/wapt	URL du référentiel WAPT dans wapt-get.ini renseigné lors de l'installation.
/StartPackages = groupe de base	Groupe de packages WAPT à installer par défaut.
:code:``/verify_cert = ``True` ou chemin relatif ssl\server\srvwapt.mydomain.lan.crt.	Valeur de verify_cert saisie lors de l'installation.
/CopyServersTrustedCA = chemin d'accès à un paquet à copier vers ssl\server	Paquet de certificats pour les connexions https (à définir par verify_cert).
/CopypackagesTrustedCA = chemin vers un paquet de certificats à copier dans ssl	Paquet de certificats pour la vérification des signatures de packaging.

Indication: Le fichier *.iss* pour le programme d'installation InnoSetup est disponible dans C:\Program Files (x86)\wapt\waptsetup\waptsetup.iss.

Vous pouvez choisir de l'adapter à vos besoins spécifiques. Une fois modifié, il vous suffira de recréer un **waptagent**.

Pour en savoir plus sur les options disponibles avec *InnoSetup*, visitez cette documentation

L'utilitaire WAPT Deployment est un petit binaire qui :

- Vérifie la version de l'agent WAPT.
- Télécharge via https le programme d'installation **waptagent.exe**.
- Lance le programme d'installation silencieux avec des arguments (options vérifiées définies pendant la compilation de l'agent WAPT).

```
/VERYSILENT /MERGETASKS= ""useWaptServer""
```

- Met à jour le serveur WAPT avec le statut de l'agent WAPT (version WAPT, statut du packaging).

Avertissement: L'utilitaire de déploiement WAPT **DÉPLOIEMENT** doit être lancé en tant que *Administrateur local*, c'est pourquoi un GPO est une bonne méthode pour déployer l'agent WAPT.

Téléchargez waptdeploy.exe depuis la page d'accueil de votre serveur WAPT.

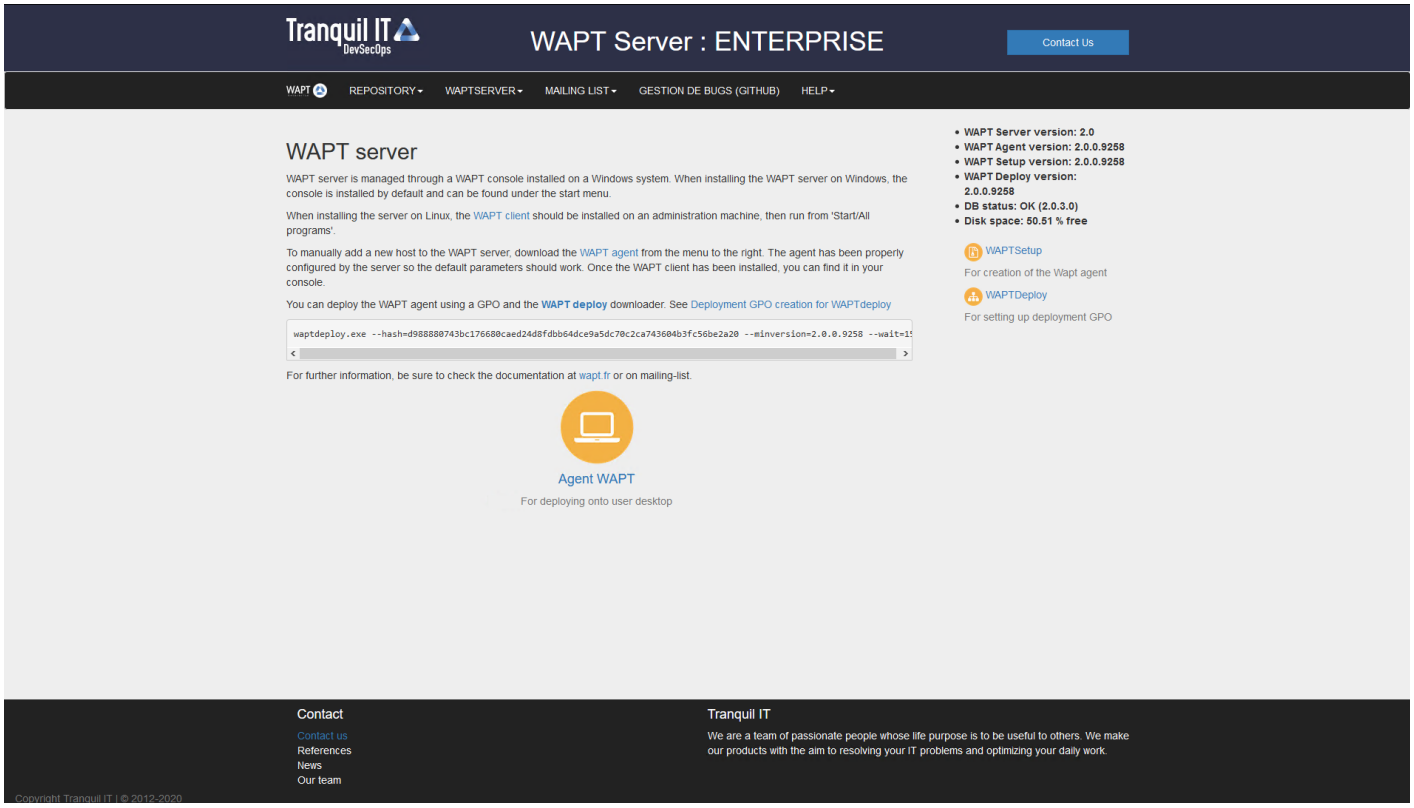


Figure4: L'interface du serveur WAPT dans un navigateur web

Avec un GPO

- Créez une nouvelle stratégie de groupe sur le serveur Active Directory (Microsoft Active Directory ou Samba-AD).
- Ajouter une nouvelle stratégie avec *Configuration de l'ordinateur* → *Stratégies* → *Paramètres Windows* → *Scripts* → *Démarrage* → *Propriétés* → *Ajouter*.

Figure5: Création d'une stratégie de groupe pour déployer l'agent WAPT

- Cliquez sur *Browse* pour sélectionner le `waptdeploy.exe`.

Figure6: Recherche du fichier de l'utilitaire WAPT Deployment sur votre ordinateur

- Copiez `waptdeploy.exe` dans le dossier de destination.

Figure7: Sélection du script de l'utilitaire WAPT Deployment

- Cliquez sur *Open* pour importer le `waptdeploy.exe`.
- Cliquez sur *Open* pour confirmer l'importation du binaire de l'utilitaire WAPT Deployment.

Indication: Il est nécessaire de fournir la somme de contrôle du `waptagent.exe` comme argument à la GPO de l'utilitaire WAPT Deployment. Cela empêchera l'hôte distant d'exécuter un binaire **`waptagent`** erroné / corrompu.

```
--hash=checksum WaptAgent --minversion=1.2.3 --wait=15 --waptsetupurl=http://srvwapt.mydomain.lan/
↪wapt/waptagent.exe
```

Les paramètres et la somme de contrôle **`waptagent.exe`** à utiliser pour la GPO de l'utilitaire de déploiement WAPT sont disponibles sur le serveur WAPT en visitant https://srvwapt.mydomain.lan.

- Copiez les paramètres requis dans la GPO.
- Cliquez sur *OK* pour passer à l'étape suivante.
- Cliquez sur *OK* pour passer à l'étape suivante.
- Appliquer la stratégie GPO résultante aux ordinateurs de l'organisation OU.

Note: Nous recommandons d'ajouter `waptdeploy.exe` aux scripts de démarrage et d'arrêt sur le GPO.

Indication: D'autres arguments sont disponibles pour l'utilitaire WAPT Deployment

---

Figure8: Sélection du script de l'utilitaire WAPT Deployment

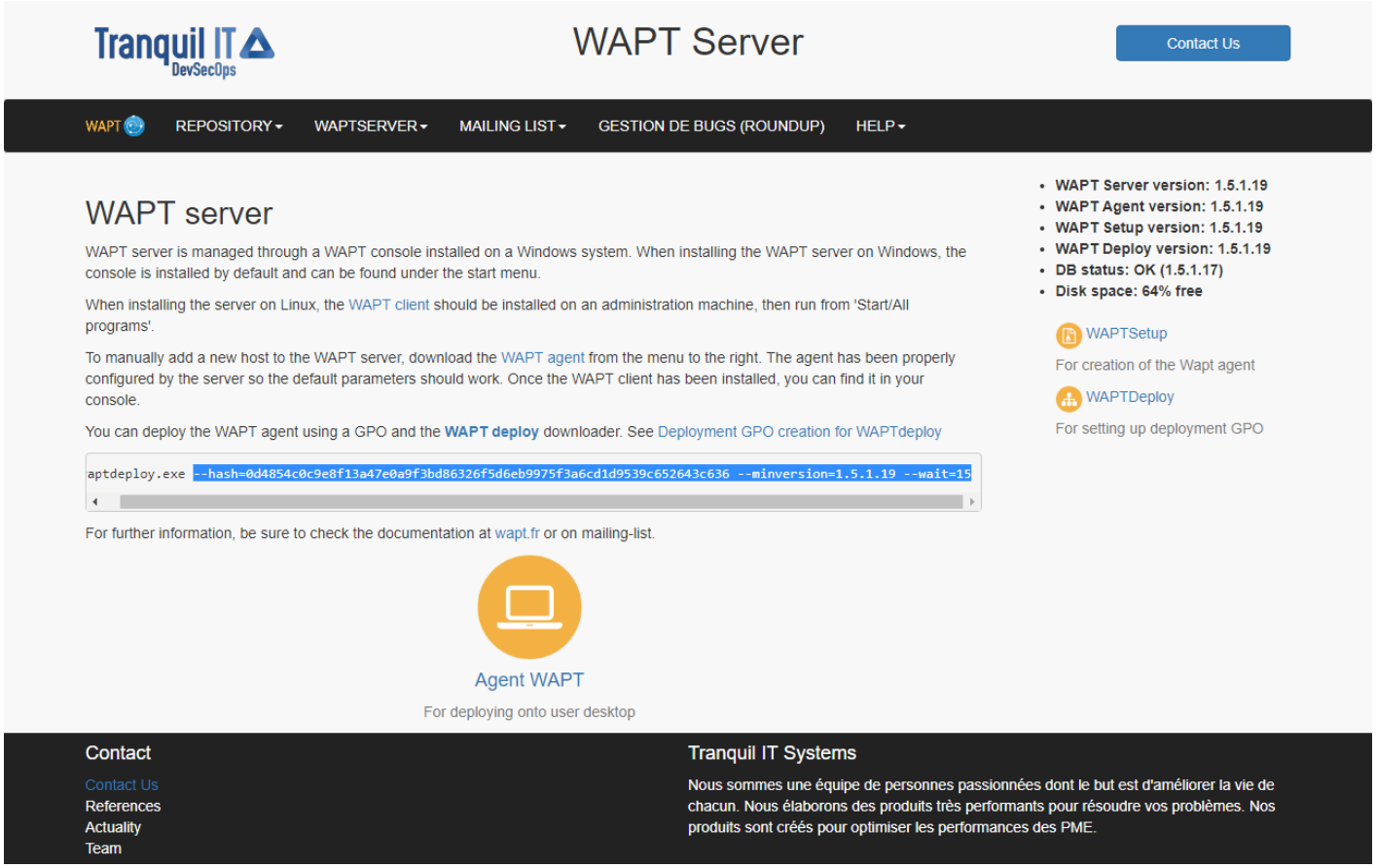


Figure9: Console web du serveur WAPT

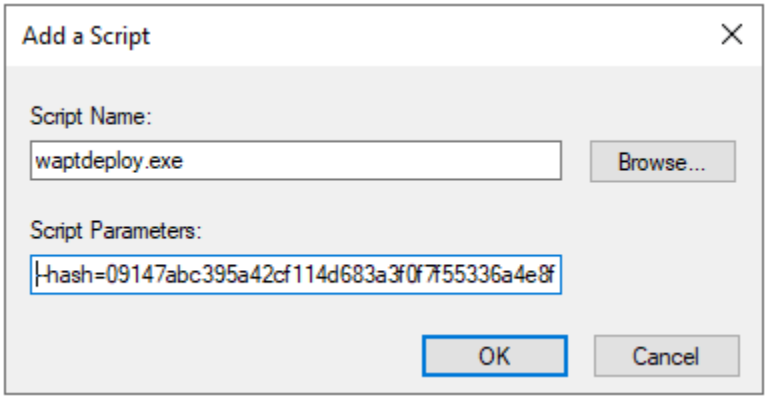


Figure10: Ajout du script de l'utilitaire de déploiement WAPT à la GPO de démarrage

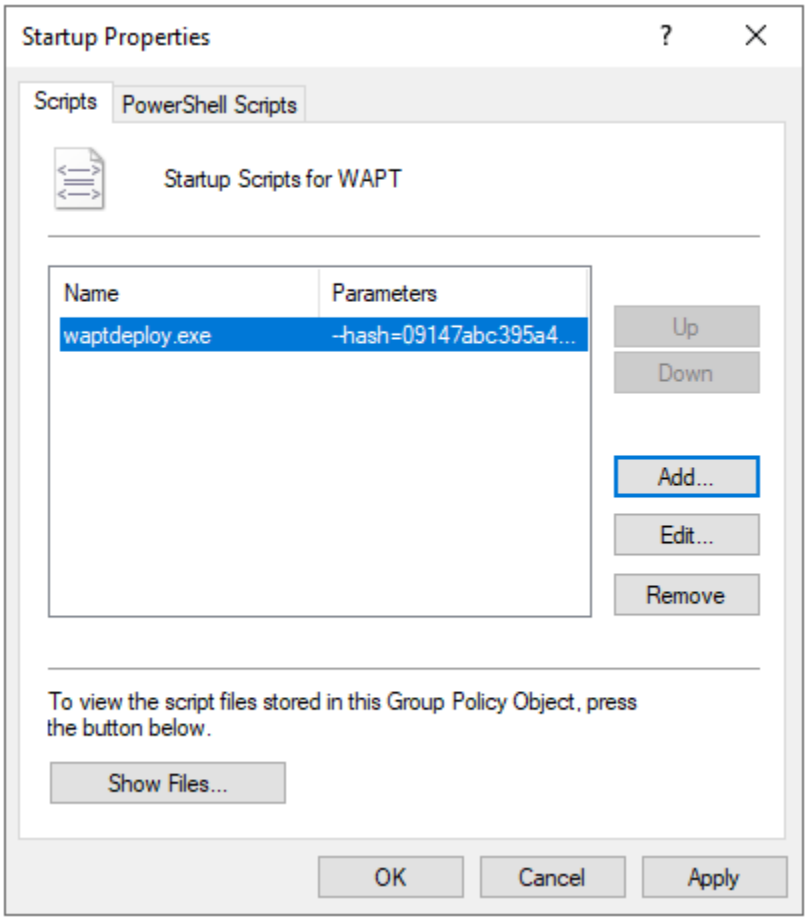


Figure11: La GPO de l'utilitaire de déploiement de WAPT à déployer au prochain démarrage

Table3: Description des options disponibles pour l'utilitaire WAPT Deployment

Options	Description
`--force`	Force l'installation de **`waptagent.exe`** même s'il est déjà installé.
`--hash` = <sha256hash>	Vérifiez que le hash sha256 du setup **`waptagent.exe`** téléchargé correspond au hash.
`--help`	Affiche les options
`--minversion` = <version>	Installer **`waptagent.exe`** si la version installée est inférieure à la minversion.
`--tâches` = autorunTray,installService,installredist2008,autoUpgradePolicy	Si elle est donnée, elle passe les arguments aux options /TASKS de **`waptagent`** (par défaut `installService, installredist2008, autoUpgradePolicy`).
`--repo_url` = <repo_url>`	Emplacement du dépôt pour obtenir **`waptagent.exe`** (par défaut <repo_url>/wapt)
`--setupargs` = <setupargs>	Ajoute des arguments à la ligne de commande de **`waptagent.exe`**.
`--wait` = <minutes>	Définit le délai d'exécution des tâches en cours et en attente si **`waptservice`** est en cours d'exécution avant l'installation.
:code:` —waptsetupurl` = <waptsetupurl>	Emplacement explicite pour télécharger l'exécutable d'installation. Il peut s'agir d'un chemin local (par défaut <repo_url>/waptagent.exe).

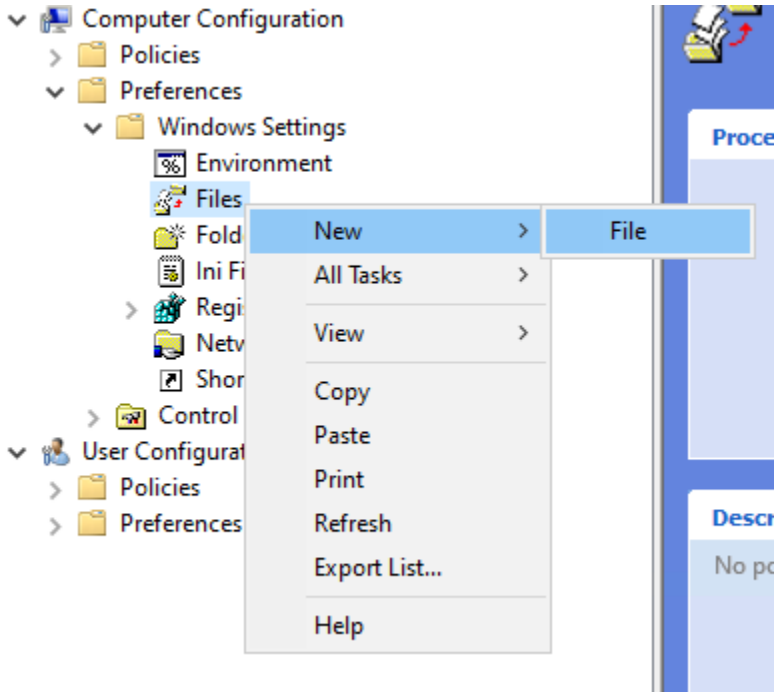
Avec une tâche planifiée

Vous pouvez également choisir de lancer l'utilitaire de déploiement WAPT à l'aide d'une tâche planifiée qui a été définie par GPO.

Indication: Cette méthode est particulièrement efficace pour déployer WAPT sur des postes de travail lorsque le réseau n'est pas disponible au démarrage ou à l'arrêt.

La méthode consiste à utiliser une GPO pour copier localement `waptdeploy.exe` et `waptagent.exe` et créer une tâche planifiée pour l'installation.

- Copiez `waptdeploy.exe` et `waptagent.exe` dans le partage netlogon de votre serveur Active Directory (`\mydomain.lan\netlogon\waptagent.exe`).
- Créez une nouvelle stratégie de groupe sur le serveur Active Directory (Microsoft Active Directory ou Samba-AD).
- Ajoutez une nouvelle stratégie avec *Configuration de l'ordinateur* → *Préférences* → *Paramètres Windows* → *Fichiers*.
- Créez un nouveau fichier et copiez l'utilitaire WAPT Deployment.



- Définir les paramètres.

Table4: Description des options pour la copie

Options	Valeur
La liste du menu déroulant Action	Remplacer
Fichier(s) source(s) champ	`\mydomain.lan\netlogon\waptdeploy.exe`
Destination File champ	`C:\Temp\waptdeploy.exe`
Suppression des erreurs sur les actions de fichiers individuels case à cocher	non vérifié
Case à cocher pour la lecture seule	non vérifié
Chef caché case à cocher	non vérifié
Archive case à cocher	vérifié

- Créez une nouvelle GPO et copiez le fichier **`waptagent.exe`**.

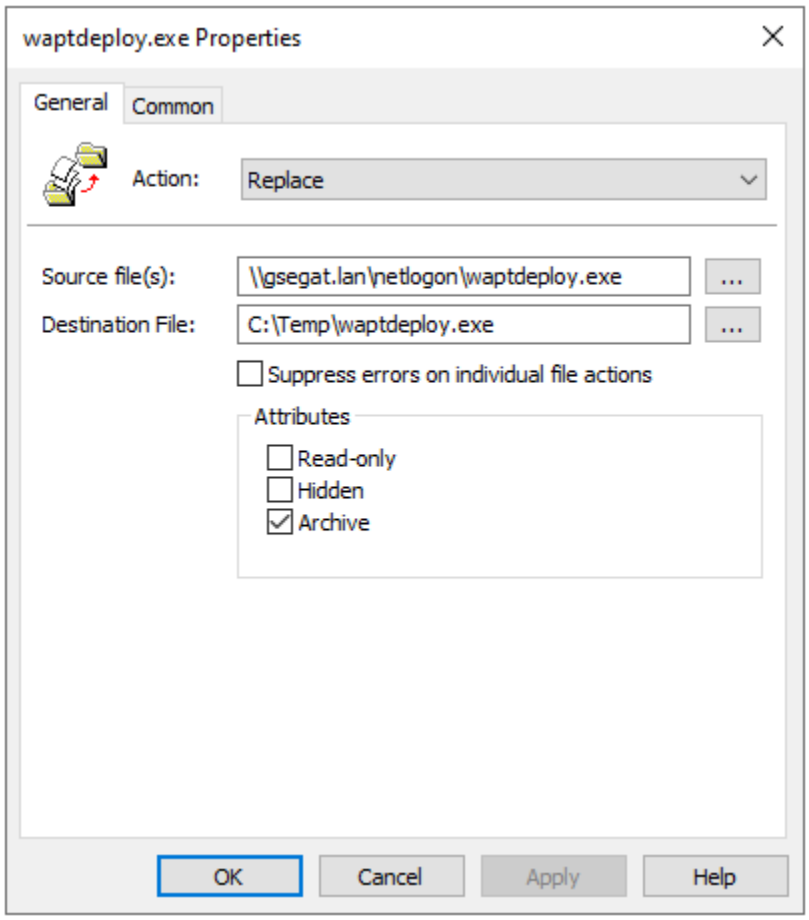
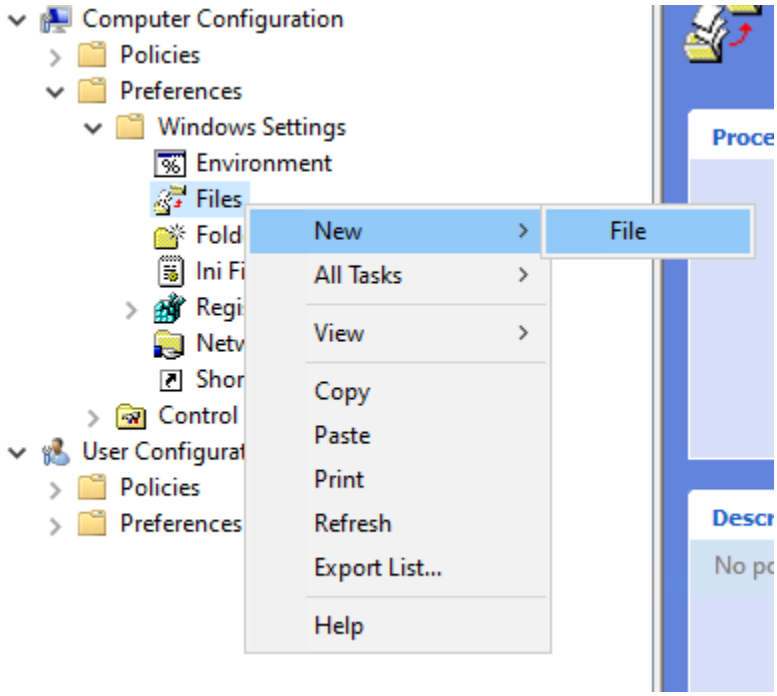


Figure12: Progression de l'installation de l'agent WAPT



- Définir les paramètres.

Table5: Description des options pour la copie

Options	Valeur
La liste du menu déroulant Action	Remplacer
Fichier(s) source(s) champ	`\mydomain.lan\netlogon\waptagent.exe`
Destination File champ	`C:\Temp\waptagent.exe`
Suppression des erreurs sur les actions de fichiers individuels case à cocher	non vérifié
Case à cocher pour la lecture seule	non vérifié
Chef caché case à cocher	non vérifié
Archive case à cocher	vérifié

- Ensuite, allez dans le menu des tâches planifiées avec *Configuration de l'ordinateur → Préférences → Paramètres du panneau de configuration → Tâches planifiées*.
- Créez une nouvelle tâche programmée avec *Clic droit → Nouveau → Tâche programmée (au moins Windows 7)*.
- Définissez *Action* sur `Replace`.
- Pour *Lorsque vous exécutez la tâche, utilisez le compte utilisateur suivant* paste *S-1-5-18* (compte système). Vous pouvez visiter pour plus d'informations.
- Vérifier *Exécuter si l'utilisateur est connecté ou non*.
- Cochez *Exécuter avec les plus hauts privilèges*, puis passez à l'onglet *Déclencheurs*.
- Créez un nouveau déclencheur.
- Vérifiez *Daily*, sélectionnez *today's date*.
- Cochez *Répéter la tâche tous les* et sélectionnez *1 heure* et *pour une durée de* sélectionnez *1 jour*.
- Vérifiez *Arrêter la tâche si elle dure plus de* et sélectionnez *2 heures*.

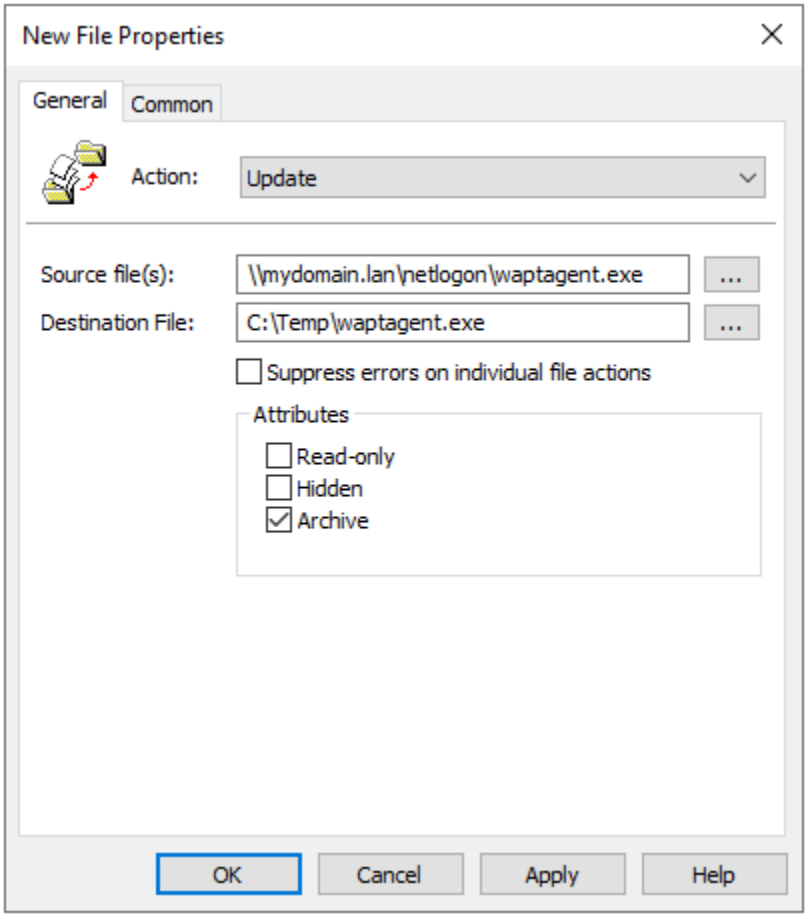


Figure13: Préparation de la GPO de mise à jour WAPT

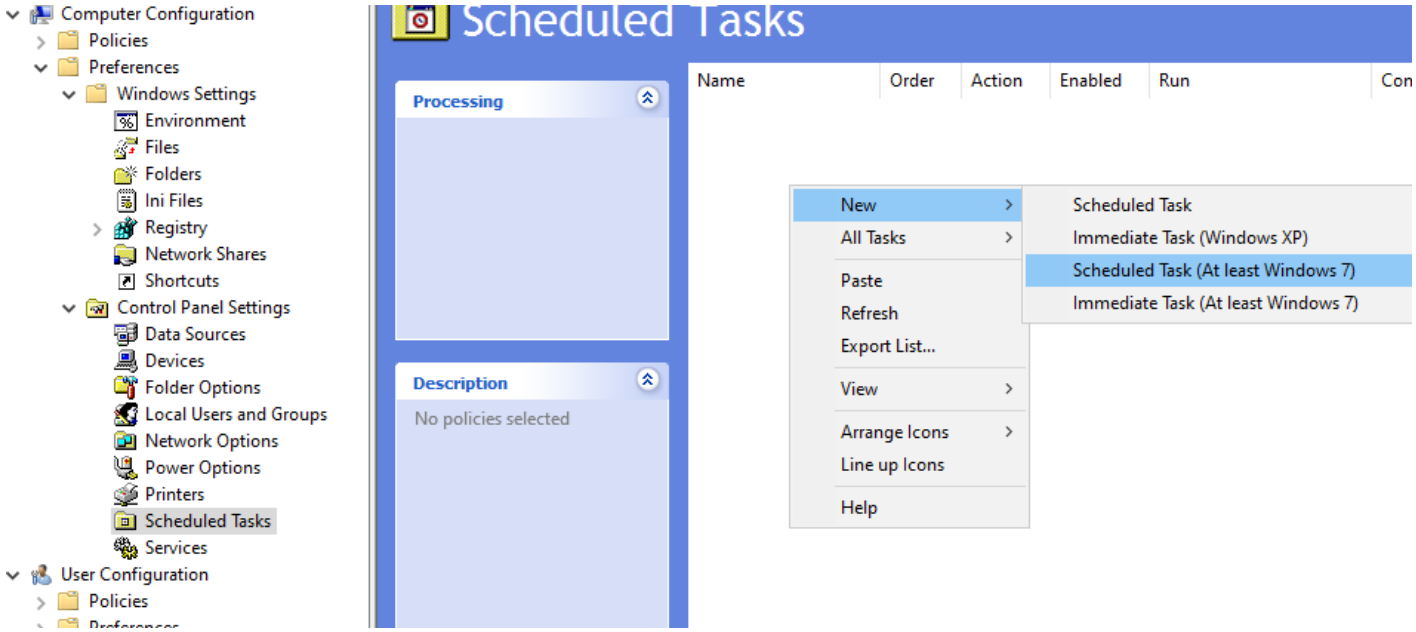


Figure14: Créez la tâche planifiée pour la fenêtre des propriétés de l'utilitaire de déploiement WAPT dans RSAT

- Vérifiez que *Enabled* est coché, puis allez dans l'onglet *Actions*.
- Créez une nouvelle action *Démarrer un programme* pour `waptdeploy.exe`.

Table6: Description des options à copier

Options	Valeur
Action	Lancer un programme
Programme / script	C:\Temp\waptagent.exe
Ajouter des arguments (facultatif)	Voir le point suivant
Début en (optionnel)	vide

Indication: Il est nécessaire de fournir la somme de contrôle du `waptagent.exe` comme argument à l'utilitaire WAPT Deployment. Cela empêchera l'hôte distant d'exécuter un binaire **`waptagent`** erroné / corrompu.

```
--hash=checksum WaptAgent --minversion=1.2.3 --wait=15 --waptsetupurl=http://srvwapt.mydomain.lan/
↪wapt/waptagent.exe
```

Les paramètres et la somme de contrôle **`waptagent.exe`** à utiliser pour la GPO de l'utilitaire de déploiement WAPT sont disponibles sur le serveur WAPT en visitant https://srvwapt.mydomain.lan.

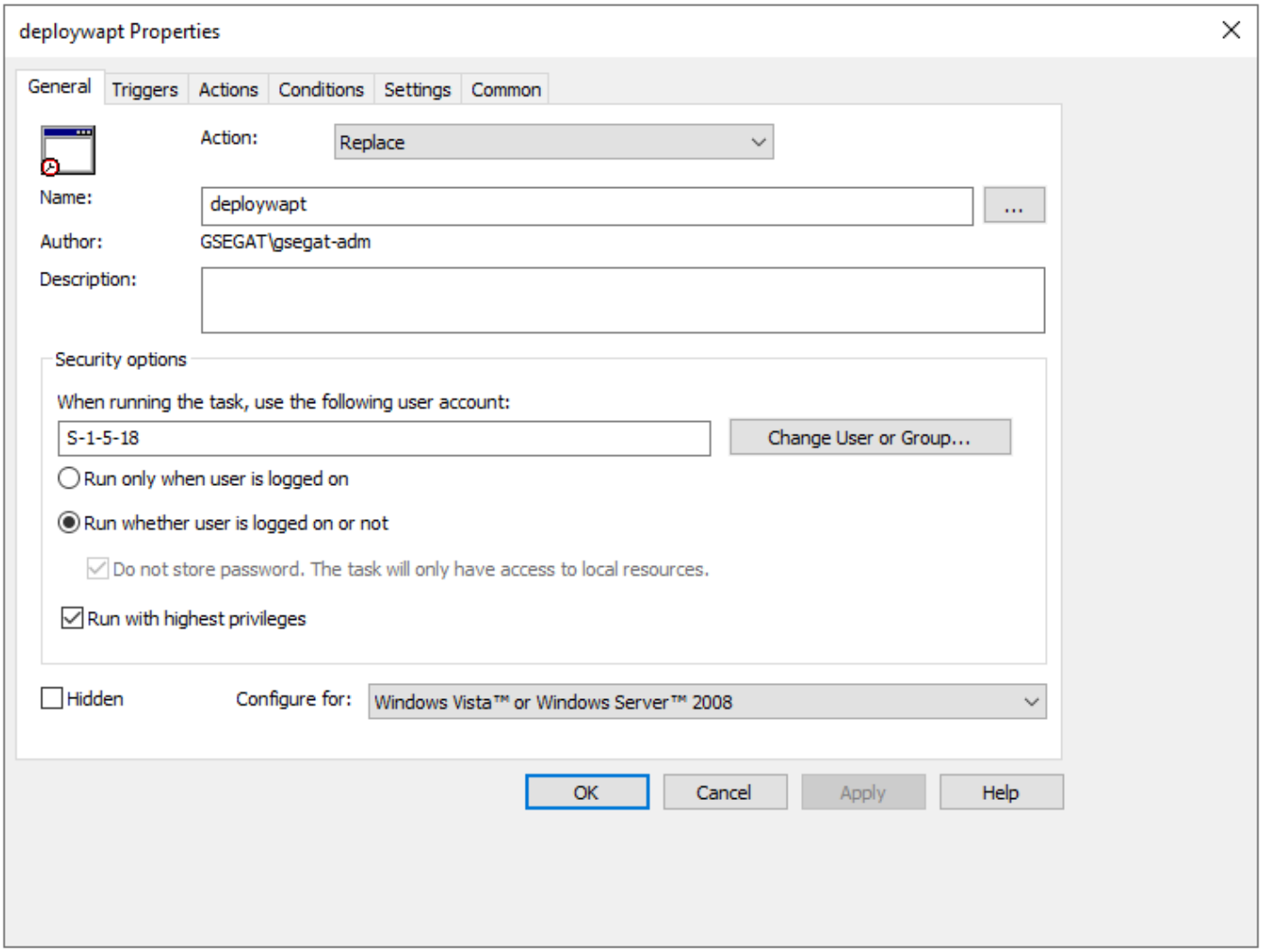


Figure15: Onglet Général de la fenêtre Propriétés dans RSAT

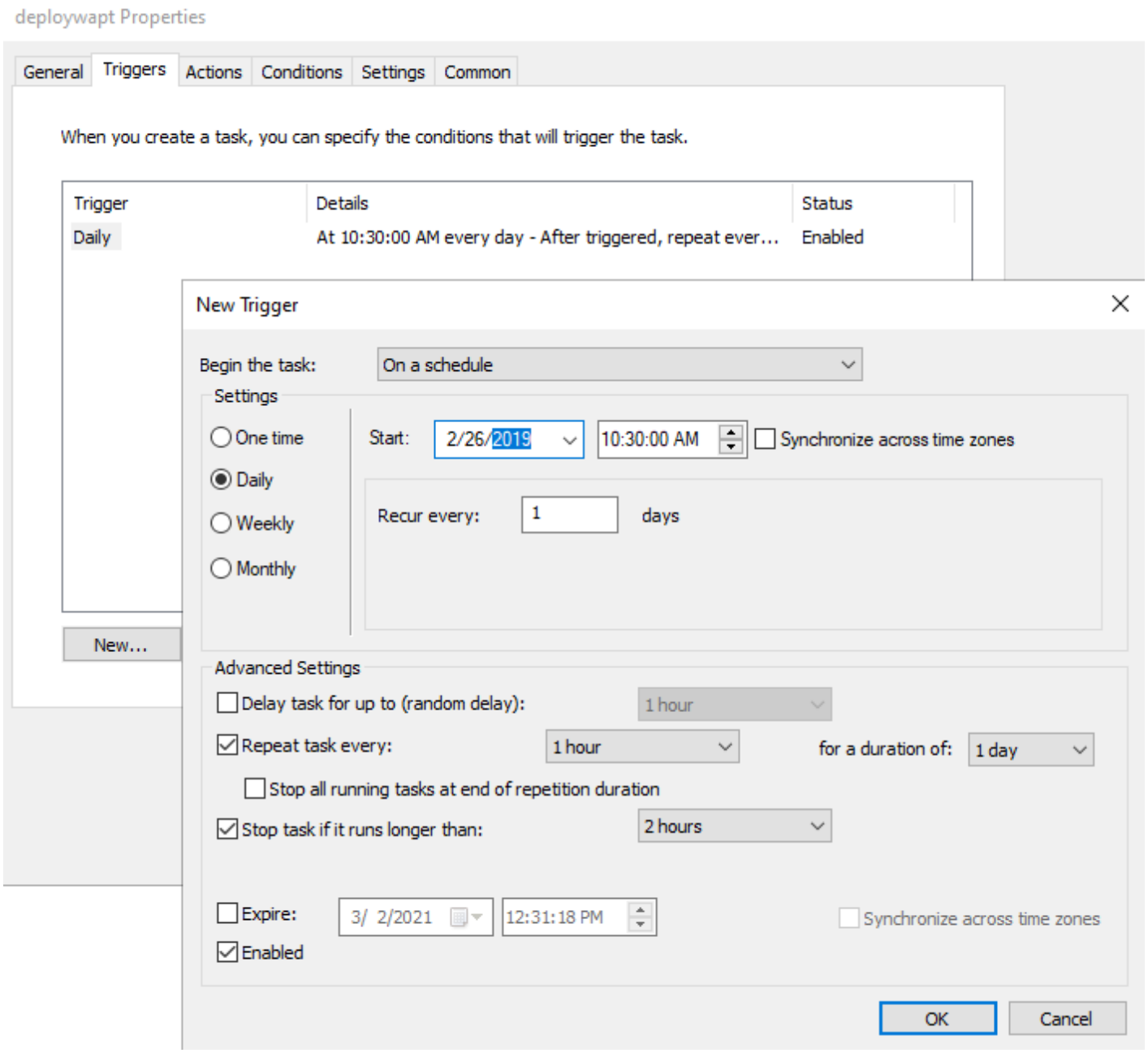


Figure16: Onglet Déclencheur dans la fenêtre Propriétés dans RSAT

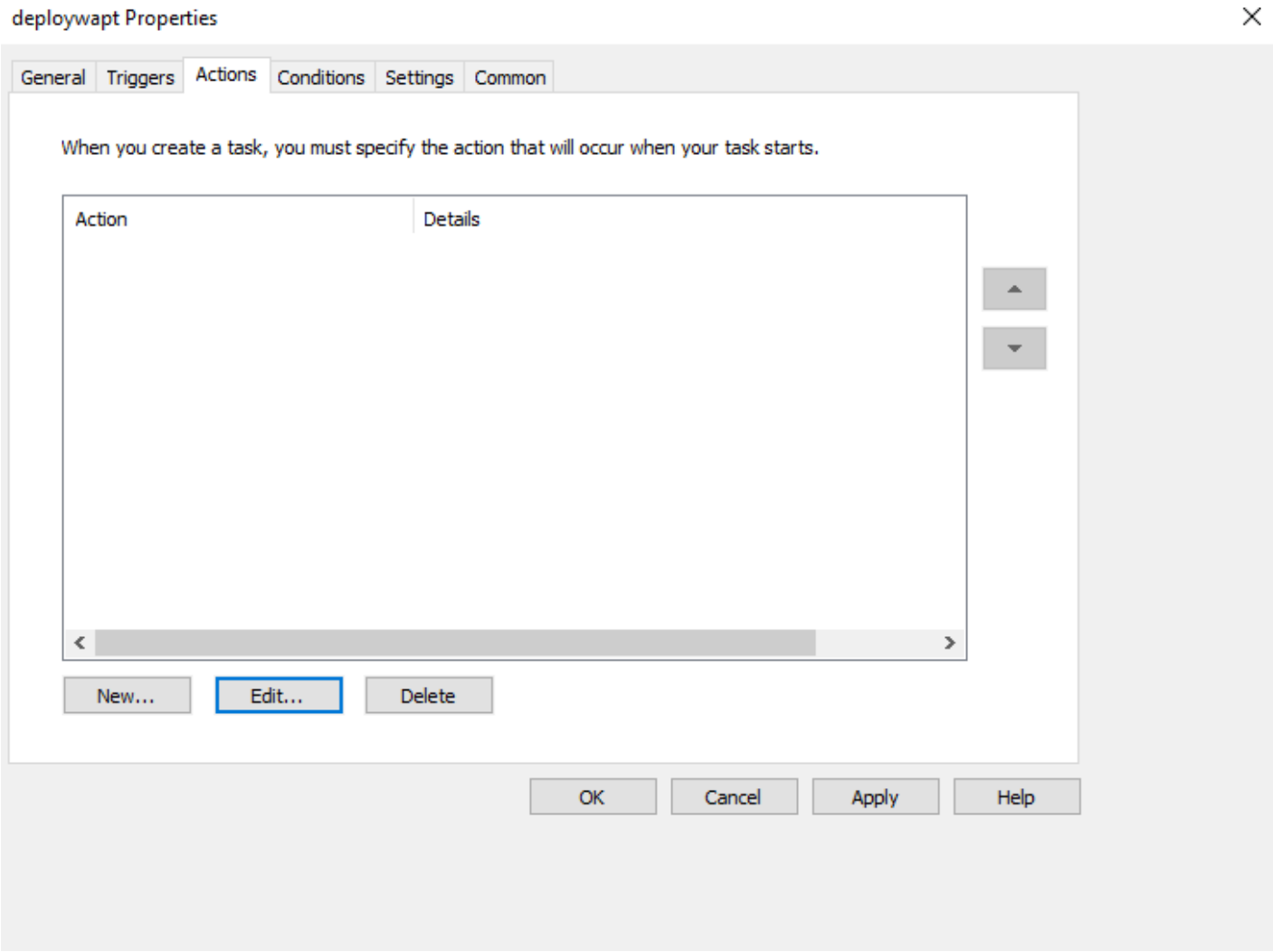


Figure17: Onglet Actions dans la fenêtre Propriétés de RSAT

Figure18: Onglet Actions dans la fenêtre Propriétés de RSAT

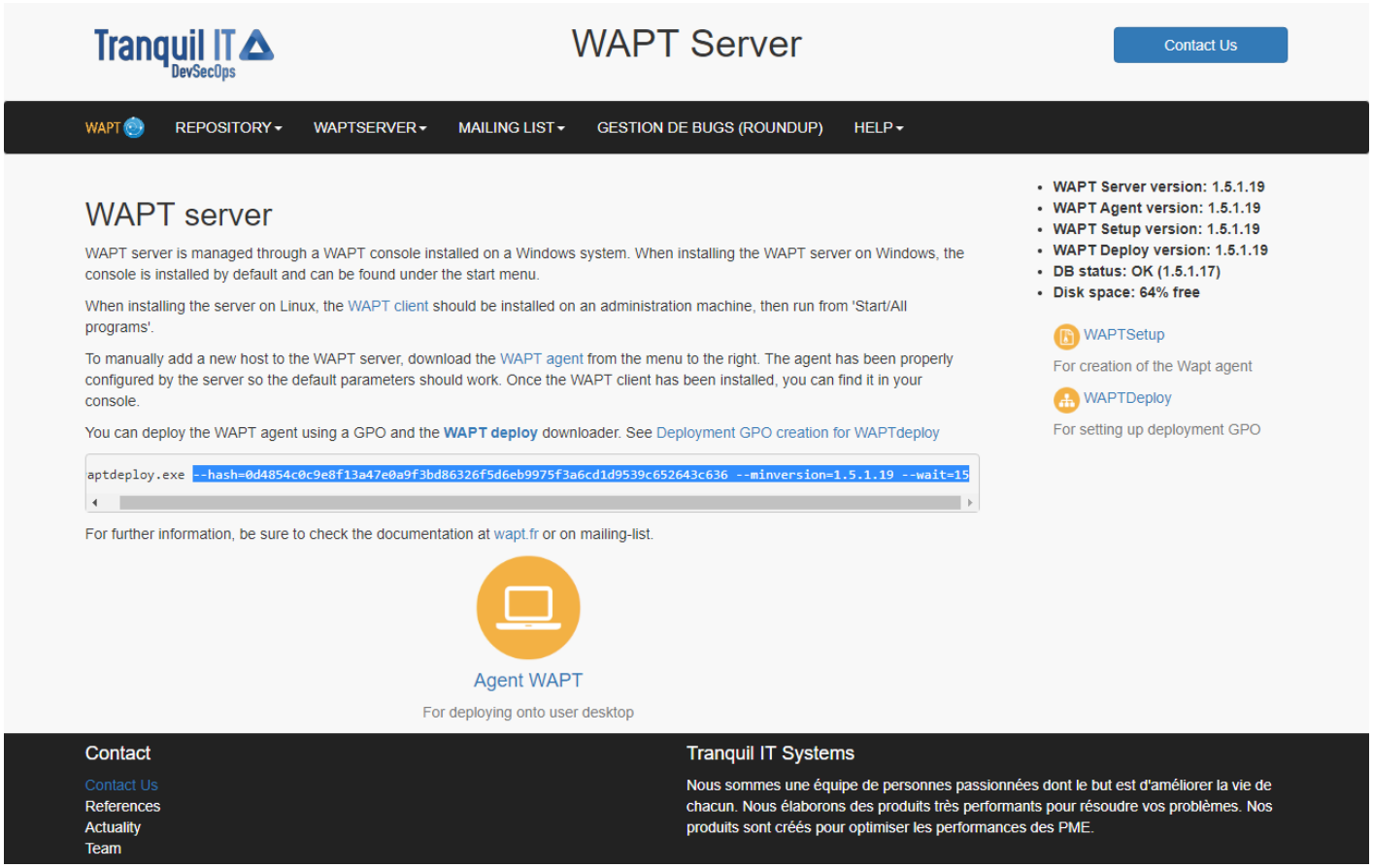


Figure19: Console web du serveur WAPT

- Copiez les paramètres requis et changez `waptsetupurl` en `C:\Temp\waptagent.exe`.

```
--hash=checksum WaptAgent --minversion=1.2.3 --wait=15 --waptsetupurl=C:\Temp\waptagent.exe
```

Table7: Description des options disponibles pour l'utilitaire WAPT Deployment

Options	Description
-force	Installe waptagent.exe même si ce n'est pas nécessaire
--hash = <sha256hash>	Vérifie que le hachage sha256 de l'installation de waptagent.exe téléchargée correspond au hachage.
--help	Affiche les options.
--minversion = 1.2.3	Installe waptagent.exe si la version installée est inférieure à la minversion.
--tâches = autorunTray,installService,installredist2008,autoUpgradePolicy	Si donné, passe ces arguments aux options /TASKS de l'installateur de waptagent. Défaut=autorunTray, installService, installredist2008, autoUpgradePolicy
--repo_url = https://srvwapt.mydomain.lan/wapt	Définit l'emplacement du référentiel pour obtenir le waptagent.exe.
--setupargs = <options>	Ajoute des arguments à la ligne de commande de waptagent.exe.
--attente = <minutes>	Définit la durée maximale autorisée pour l'achèvement des tâches en cours et en attente si le service WAPT est en cours d'exécution avant l'installation.
--waptsetupurl = https://srvwapt.mydomain.lan/wapt/waptagent.exe	Définit un emplacement explicite pour télécharger l'exécutable d'installation. Cela peut être un chemin local (par défaut=:file:<*repo_url*>*/waptagent.exe*).

- Passez à l'onglet *Paramètres*.
- Dans l'onglet *Paramètres*, cochez uniquement *Exécuter la tâche dès que possible après un démarrage programmé manqué*.

Indication: Pour vérifier que le GPO fonctionne, vous pouvez exécuter la commande **gpupdate /force** et vérifier que la tâche planifiée est présente sur l'ordinateur en lançant **Task Scheduler** en tant qu'administrateur local.

20.2 Mise à jour de l'agent WAPT sous Windows

Pour chaque *upgrade* du serveur WAPT, vous devrez mettre à niveau les agents WAPT.

Pour ce faire, vous devez *générer l'agent WAPT* et le déployer.

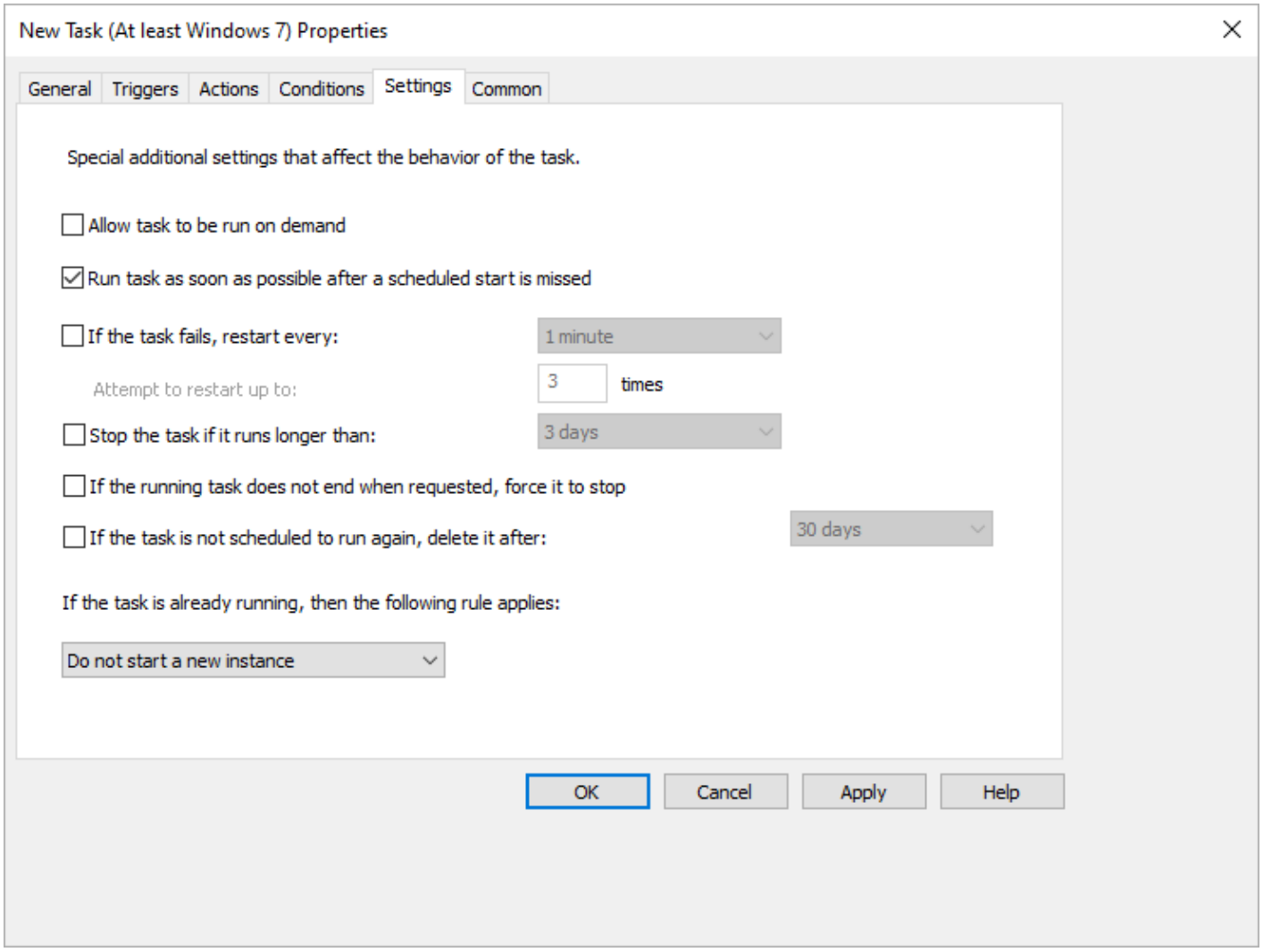


Figure20: Onglet Paramètres dans la fenêtre Propriétés de RSAT

20.2.1 Manuellement

Vous pouvez le faire manuellement *en suivant cette documentation sur l'installation de l'agent WAPT*.

Indication: Il s'agit de la seule solution de mise à niveau disponible pour l'instant pour macOS et Linux.

20.2.2 Via waptupgrade

Pendant que vous *générer* l'agent WAPT, un packaging nommé `waptupgrade` est créé.

Ce packaging est un packaging WAPT standard conçu pour mettre à niveau les agents WAPT sur des hôtes distants.

Indication: Pour l'instant, `waptupgrade` ne fonctionne que pour Windows. Waptupgrade ne met pas à jour l'agent WAPT si la version du serveur WAPT et la version de l'agent WAPT sont les mêmes.

La mise à niveau des agents WAPT à l'aide du packaging `waptupgrade` est un processus en deux étapes :

- D'abord le paquet copie le fichier `waptsetup.exe` sur l'ordinateur client et crée une tâche planifiée qui exécutera **`waptsetup.exe`** avec des paramètres d'installation prédéfinis deux minutes après la création de la tâche planifiée. À ce moment-là, le paquetage lui-même est installé et l'inventaire du serveur WAPT indique que l'installation du paquetage est *OK*, avec la bonne version installée, mais l'inventaire montrera toujours l'ancienne version car l'Agent WAPT n'est pas encore mis à jour.
- Après deux minutes, la tâche planifiée démarre et exécute **`waptsetup.exe`** avec la configuration prédéfinie créée dans la Console WAPT. Cette nouvelle méthode conserve le **`waptsetup.exe`** signé par Tranquil IT, mais la configuration de l'agent WAPT viendra du serveur WAPT. **`waptsetup.exe`** arrête le service WAPT local, met à jour WAPT localement, puis redémarre le service WAPT. La tâche planifiée est alors automatiquement supprimée et l'agent WAPT commence à renvoyer son inventaire au serveur WAPT. A partir de ce moment, l'inventaire sur le serveur WAPT montrera la nouvelle version de l'agent WAPT.

Il est recommandé d'installer `waptupgrade` sur tous les hôtes pour que les agents WAPT se mettent à jour automatiquement.

CHAPTER 21

Gérer l'agent WAPT sous Linux et MacOS

21.1 Déployer l'agent WAPT sur Linux et MacOS

Note: Pour installer WAPT sur un client Windows, la configuration minimale requise est la suivante :

- 512Mo Ram;
- 1 CPU;
- 300Mo d'espace disque (sans tenir compte de l'espace de cache pour les paquets WAPT).

Le processus dépend de votre système d'exploitation :

Distributions basées sur Debian / Ubuntu

Indication: The WAPT Agent for Debian has been tested on Debian 9, 10 and 11.

L'agent WAPT pour Ubuntu n'a été testé que sur Ubuntu Bionic et Ubuntu Focal.

- Mettez à jour la distribution sous-jacente et vérifiez que le transport apt https est installé

```
sudo apt update && apt upgrade -y
sudo apt install apt-transport-https lsb-release gnupg -y
```

- Récupérer la clé *.gpg*, l'ajouter au référentiel Tranquil IT et installer l'agent WAPT.

```
sudo wget -O - https://wapt.tranquil.it/$(lsb_release -is)/tiswapt-pub.gpg | apt-key add -
sudo echo "deb https://wapt.tranquil.it/$(lsb_release -is)/wapt-2.3/ $(lsb_release -cs) main" > /
↪etc/apt/sources.list.d/wapt.list
```

(suite sur la page suivante)

(suite de la page précédente)

```
export DEBIAN_FRONTEND=noninteractive
sudo apt update
sudo apt install tis-waptagent -y
unset DEBIAN_FRONTEND
```

Distributions basées sur RedHat 7

Indication: The WAPT Agent for Redhat based system has been tested on Redhat 7/8/9 and derivatives on x86_64 platforms

- Mettre à jour la distribution sous-jacente.

```
yum update
```

- Récupérez la clé `.gpg` et configurez le référentiel WAPT.

```
RH_VERSION=$(cat /etc/system-release-cpe | awk -F: '{ print $5}')
wget -q -O /tmp/tranquil_it.gpg "https://wapt.tranquil.it/redhat${RH_VERSION}/RPM-GPG-KEY-TISWAPT-{
↪$RH_VERSION}"; rpm --import /tmp/tranquil_it.gpg

cat > /etc/yum.repos.d/wapt.repo <<EOF
[wapt]
name = WAPT Server Repo
baseurl = https://wapt.tranquil.it/redhat${RH_VERSION}/wapt-2.3/
enabled = True
gpgcheck = True
EOF
```

- installer l'agent WAPT en utilisant yum :

```
yum install tis-waptagent
```

macOS

Indication: The WAPT Agent has only been tested on **Intel architecture** and **Apple Silicon M1** processors:

- Mojave (10.14) ;
- Catalina (10.15) ;
- Big Sur (11.x) ;
- Monterey (12.x).
- Ventura (13.x).

- Download and install the WAPT Agent (note: the hash string may change, to get the latest, point your brower on the url https://wapt.tranquil.it/wapt/releases/wapt-2.3/). Choose the version depending on your processor architecture (intel or m1):

```
# for mac m1
curl -o tis-waptagent-2.3.0.13516-69968974-macos-all-arm64.pkg https://wapt.tranquil.it/wapt/
↪releases/wapt-2.3.0.13516-69968974/tis-waptagent-2.3.0.13516-69968974-macos-all-arm64.pkg
```

(suite sur la page suivante)

(suite de la page précédente)

```
# for mac intel
curl -o tis-waptagent-2.3.0.13516-69968974-macos-all-x86_64.pkg https://wapt.tranquil.it/wapt/
↪releases/wapt-2.3.0.13516-69968974/tis-waptagent-2.3.0.13516-69968974-macos-all-x86_64.pkg

sudo installer -target / -pkg tis-waptagent*.pkg
```

21.1.1 Créer le fichier de configuration de l'agent WAPT

Avant d'installer le fichier de configuration de l'agent WAPT, vous devez créer une *configuration initiale de l'Agent WAPT* dans la console WAPT.

Lorsque vous avez terminé, copier la commande avec *Copy installation command*.

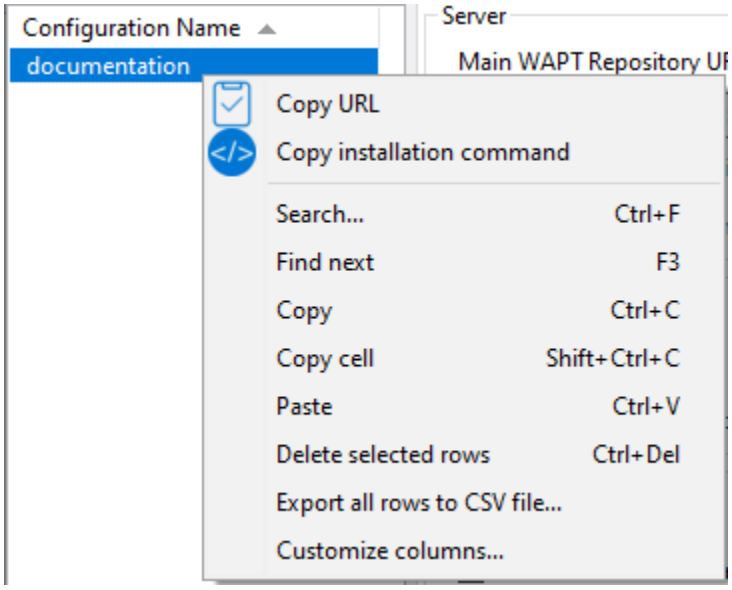


Figure1: Liste de menus montrant l'option *Copy installation command*

Utilisez ensuite cette copie de l'invite de commande sur l'agent Linux / macOS.

```
wapt-get add-config-from-url https://srvwapt.mydomain.lan/wapt/conf.d/default_
↪f0288df2131b8dce667b8c34b9999959bdc2d253b3934fcb3be2eabad8a50021.json␣
↪f0288cf2131b9dce667b8c34b9999959bdc2d253b3934fcb3be2eabad8a50020
```

Enfin, exécuter la commande suivante pour enregistrer votre hôte Linux avec le serveur WAPT :

```
sudo wapt-get register
```

Lorsque vous avez modifié la configuration de l'agent WAPT, vous devez redémarrer l'agent WAPT en utilisant la commande suivante :

```
sudo wapt-get restart-waptservice
```

Matrice des caractéristiques

Certaines fonctionnalités sont actuellement indisponibles sur Linux et macOS :

- l'installation des mises à jour à l'arrêt (WAPT Exit) ;
- toutes les fonctions spécifiques a Windows.

Particularités de la fonctionnalité du domaine

Sur Linux :

- Les tests ont été effectués avec sssd avec un domaine Active Directory et une authentification kerberos.
- Pour intégrer une machine dans le domaine Active Directory, vous pouvez suivre cette documentation.
- Pour que les groupes Active Directory fonctionnent correctement, vous devez vérifier que la commande **`id hostname$`** renvoie la liste des groupes dont la machine est membre.

Attention: Nous avons remarqué que la requête LDAP de kerberos ne fonctionne pas si l'enregistrement DNS inverse n'est pas configuré correctement pour les contrôleurs de domaine. Ces enregistrements **DOIVENT** donc être créés s'ils n'existent pas.

21.2 Ancienne méthode pour configurer l'Agent WAPT fonctionnant sur Linux / macOS

Attention: Veuillez consulter la nouvelle méthode *pour déployer le fichier de configuration* à la place.

21.2.1 Création du fichier de configuration de l'agent

Indication: Utilisez l'adresse de votre serveur dans **repo_url** et **wapt_server**.

```
sudo cat > /opt/wapt/wapt-get.ini <<EOF
[global]
repo_url = https://srvwapt.mydomain.lan/wapt
wapt_server = https://srvwapt.mydomain.lan
use_hostpackages = True
use_kerberos = False
verify_cert = False
EOF
```

21.2.2 Copie du certificat de signature de paquet

Vous devez copier manuellement, ou par script, le certificat public de votre Autorité de Certification de signature de paquet.

Le certificat doit se trouver sur votre machine Windows dans `C:\Program Files (x86)\wapt\ssl\`.

Copiez votre ou vos certificats dans `/opt/wapt/ssl` en utilisant **WinSCP** ou **rsync** si vous déployez sur Linux ou macOS.

21.2.3 Copie du certificat SSL/TLS

Si vous avez déjà configuré votre serveur WAPT pour utiliser les *certificats SSL/TLS avec Nginx*, vous devez copier le certificat dans votre agent WAPT Linux.

Le certificat doit se trouver sur votre ordinateur Windows, dans `C:\Program Files (x86)\wapt\ssl\server\`.

- Copiez votre ou vos certificats dans `/opt/wapt/ssl/server/` en utilisant **WinSCP** ou **rsync** si vous déployez sur Linux ou MacOS.
- Ensuite, modifiez dans votre fichier de configuration `/opt/wapt/wapt-get.ini` le chemin vers votre certificat.
- Et donnez le chemin absolu de votre certificat.

```
verify_cert = /opt/wapt/ssl/server/YOURCERT.crt
```

Indication: Changez `YOURCERT.crt` par le nom de votre certificat.

21.3 Mise à jour de l'agent WAPT sur Linux et MacOS

Pour chaque *upgrade* du serveur WAPT, vous devrez mettre à niveau les agents WAPT.

Pour ce faire, vous devez *générer l'agent WAPT* et le déployer.

21.3.1 Manuellement

Vous pouvez le faire manuellement *en suivant cette documentation sur l'installation de l'agent WAPT*.

Indication: Il s'agit de la seule solution de mise à niveau disponible pour l'instant pour macOS et Linux.

CHAPTER 22

Désinstallation de l'agent WAPT des clients

22.1 Windows

Si vous devez désinstaller les agents WAPT des clients, le programme de désinstallation est automatiquement créé dans l'emplacement d'installation de WAPT. Par défaut, il s'agit de `C:\Program Files (x86)\wapt\unins000.exe`.

- La désinstallation silencieuse par défaut d'un agent WAPT peut être réalisée avec la commande suivante.

```
unins000.exe /VERYSILENT
```

- Un argument supplémentaire peut être passé à **unins000.exe** pour tout nettoyer.

```
unins000.exe /VERYSILENT /purge_wapt_dir=1
```

Table1: Liste complète des arguments de ligne de commande pour **unins000.exe**

Paramètres	Description
/VERYSILENT	Lance unins000.exe en silencieux.
/purge_wapt_dir = **1**	Purge le répertoire WAPT (supprime tous les dossiers et fichiers).

- Il est possible d'utiliser un paquet pour cela.

```
# -*- coding: utf-8 -*-
from setuphelpers import *

def install():

  print("Creation of the task")
  task = create_onetime_task('removewapt', "unins000.exe", "/VERYSILENT /purge_wapt_dir = True")
  print(task)
```

22.1.1 Réactivation des mises à jour de Windows avant la désinstallation

Dans le cas où vous avez utilisé WAPT pour gérer les mises à jour de Windows, vous voudrez peut-être réactiver le comportement par défaut de Windows Updates avant de désinstaller l'agent WAPT.

Pour ce faire, voici un exemple de paquet à pousser avant de désinstaller l'agent WAPT :

```
# -*- coding: utf-8 -*-
from setuphelpers import *

def install():
  print('Disable WAPT WUA')
  inifile_writestring(WAPT.config_filename,'waptwua','enabled','false')

  print('DisableWindowsUpdateAccess registry to 0')
  registry_set(HKEY_LOCAL_MACHINE,r'Software\Policies\Microsoft\Windows\WindowsUpdate',
↪'DisableWindowsUpdateAccess',0,REG_DWORD)

  print('AUOptions registry to 0')
  registry_set(HKEY_LOCAL_MACHINE,r'SOFTWARE\Microsoft\Windows\CurrentVersion\WindowsUpdate\Auto␣
↪Update','AUOptions',0,REG_DWORD)

  print('Enable wuauserv')
  run_notfatal('sc config wuauserv start= auto')
  run_notfatal('net start wuauserv')

  print('Reload WAPT configuration')
  WAPT.reload_config_if_updated()
```

22.2 Linux

- La désinstallation par défaut d'un agent WAPT peut être réalisée avec la commande suivante, selon votre OS Linux :

Debian / Ubuntu

```
apt remove --purge tis-waptagent
```

Redhat and derivatives

```
yum remove tis-waptagent
```

- Une étape supplémentaire peut être effectuée à l'aide de ces commandes (WIP).

Debian / Ubuntu

```
rm -f /opt/wapt/
rm /etc/apt/sources.list.d/wapt.list
```

Redhat and derivatives

```
rm -f /opt/wapt/
rm /etc/yum/yum.repos.d/wapt.list
```

22.3 MacOS

La désinstallation par défaut d'un agent WAPT peut être réalisée avec la commande suivante :

```
pkgutil --only-files --files it.tranquil.waptservice > file_list
sudo pkgutil --forget it.tranquil.waptservice
```

CHAPTER 23

Upgrade WAPT Server

Si votre serveur WAPT est une machine virtuelle, prenez un instantané de la VM. De cette façon, vous pourrez revenir en arrière facilement dans le cas rare où la mise à jour échoue.

Avertissement: Après chaque mise à jour du serveur, mettez à jour votre *console* puis *regénérer* l'agent WAPT.

Avant de mettre à niveau le serveur WAPT, veuillez consulter le tableau de compatibilité de mise à niveau suivant :

Table1: Possibilités de mise à niveau WAPT disponibles

	Vers WAPT 2.3
Depuis WAPT 1.8.2	✅
Depuis WAPT 2.0	✅
Depuis WAPT 2.1	✅
Depuis WAPT 2.2	✅

Avertissement: Si vous mettez à niveau à partir d'une version antérieure à WAPT 2.1, le processus *d'activation de licence* a changé.

23.1 Changement de l'édition WAPT (Community, Discovery, Enterprise)

WAPT Community is no longer supported. If you want to upgrade from WAPT 1.8.2 Community you can upgrade to WAPT Discovery or WAPT Enterprise. Please note that WAPT Discovery is limited to 300 clients.

Pour mettre à niveau WAPT Community vers WAPT Enterprise, suivez la documentation standard 1.8.2 to 2.3 upgrade documentation.

Le serveur effectuera les modifications appropriées.

Pour mettre à niveau WAPT Discovery vers WAPT Enterprise, il suffit de modifier votre *licence*.

Si votre licence Enteprise expire, WAPT basculera sur l'édition Discovery. Si vous utilisez WAPT Discovery et que vous avez plus de 300 ordinateurs dans votre inventaire, la console WAPT cessera de fonctionner et ne vous donnera que la possibilité de supprimer des entrées d'ordinateur de l'inventaire. La console WAPT redeviendra opérationnelle lorsque l'inventaire repassera sous la limite des 300 ordinateurs.

23.2 Minor upgrade

23.2.1 Mise à niveau de WAPT de 2.3 à la dernière 2.3

To do a minor upgrade please follow the procedure corresponding to your server operating system.

Debian / Ubuntu

- Mettez à jour la distribution sous-jacente et mettez à niveau le serveur WAPT.

```
export DEBIAN_FRONTEND=noninteractive
apt update && apt upgrade -y
apt install tis-waptserver tis-waptsetup -y
unset DEBIAN_FRONTEND
```

- Lancer l'étape de post-configuration *étape de post-configuration*
- Une fois terminé, votre serveur est prêt.

RedHat et dérivés

- Mettez à jour la distribution sous-jacente et mettez à niveau le serveur WAPT.

```
yum update -y
yum install tis-waptserver tis-waptsetup -y
```

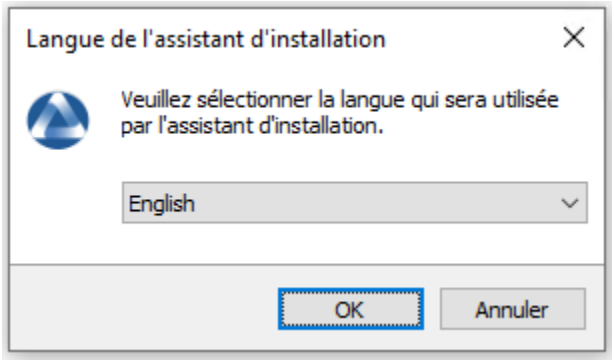
- Lancer l'étape de post-configuration *étape de post-configuration*
- Une fois terminé, votre serveur est prêt.

Windows

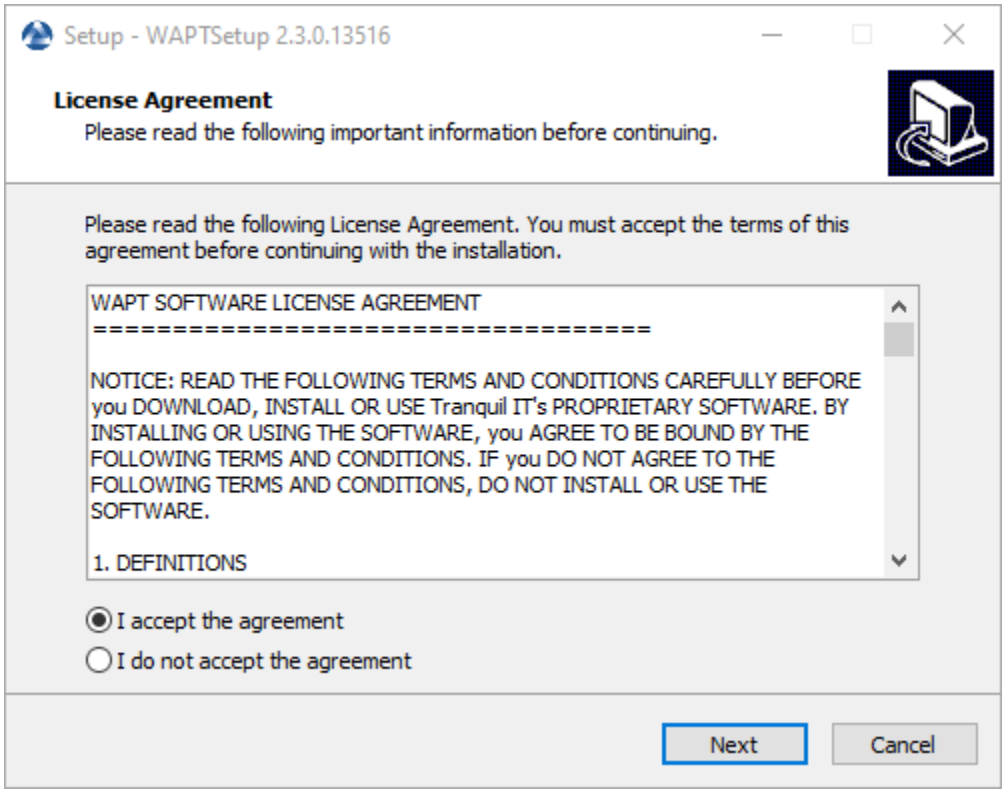
- Télécharger et exécuter `waptserversetup.exe`.

Attention: L'installation du serveur WAPT doit être effectuée à l'aide d'un compte **Administrateur local** sur l'hôte

- Choisir la langue de l'installeur WAPT.



- Acceptez la licence publique GNU et cliquez sur *Suivant* pour passer à l'étape suivante.



- Acceptez la licence publique GNU et cliquez sur *Suivant* pour passer à l'étape suivante.
- Choisissez une tâche supplémentaire (laissez la valeur par défaut).
- Ne pas modifier le mot de passe du serveur WAPT (si cela n'est pas nécessaire).

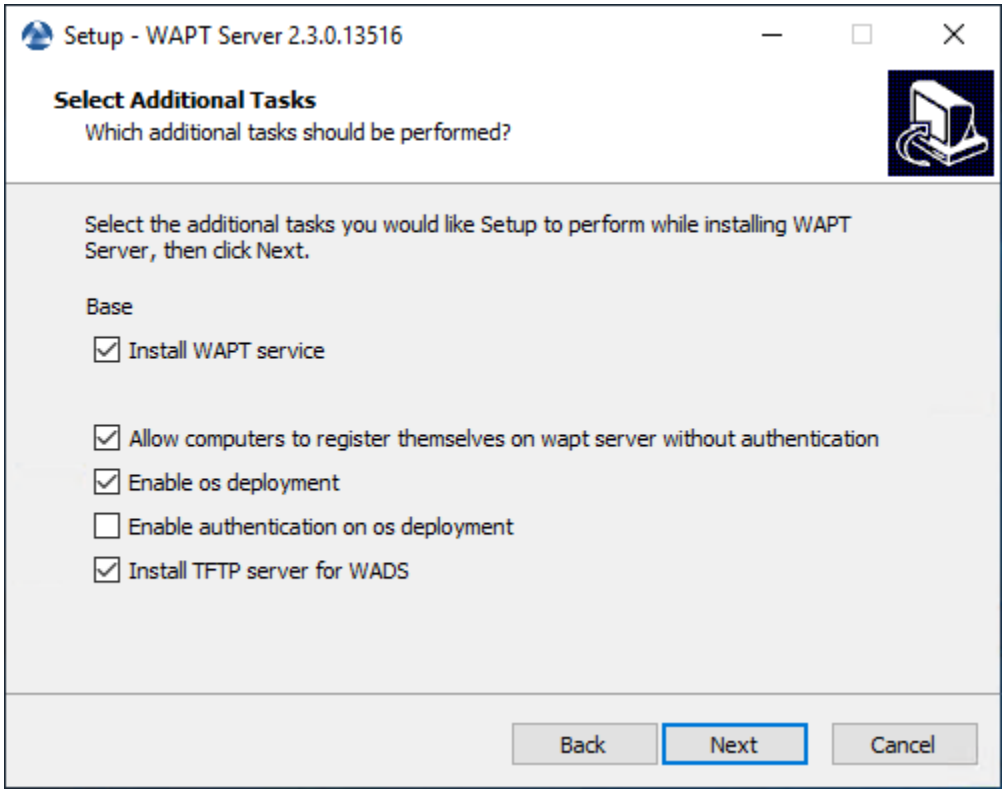
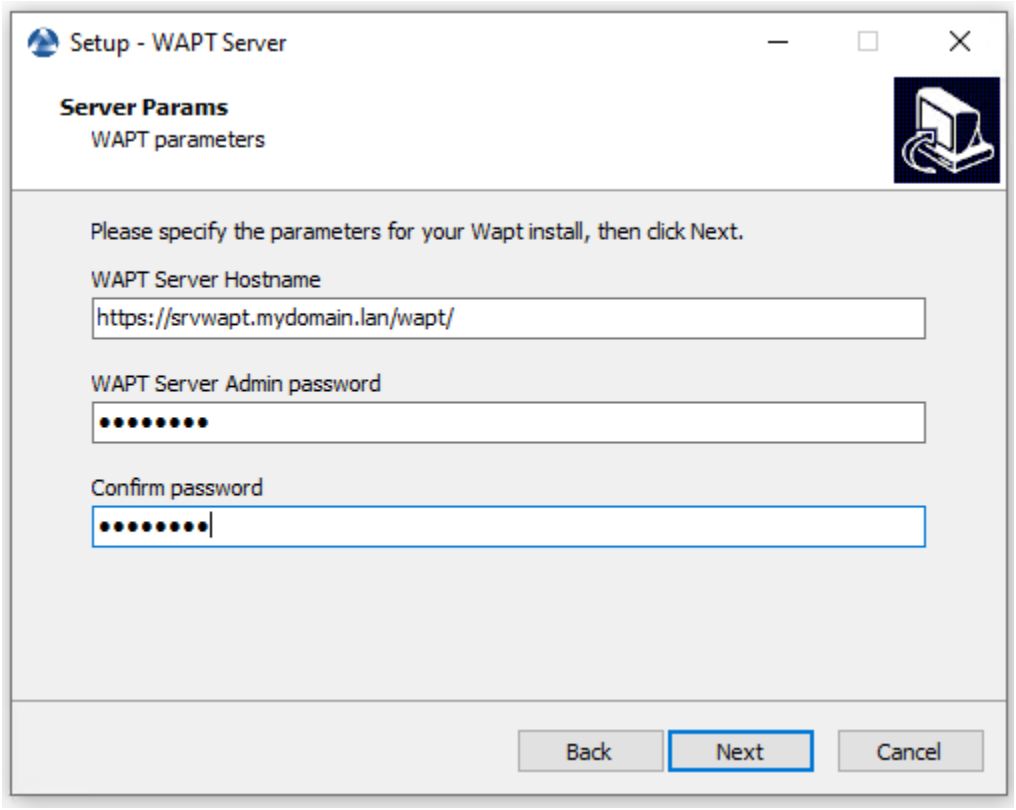
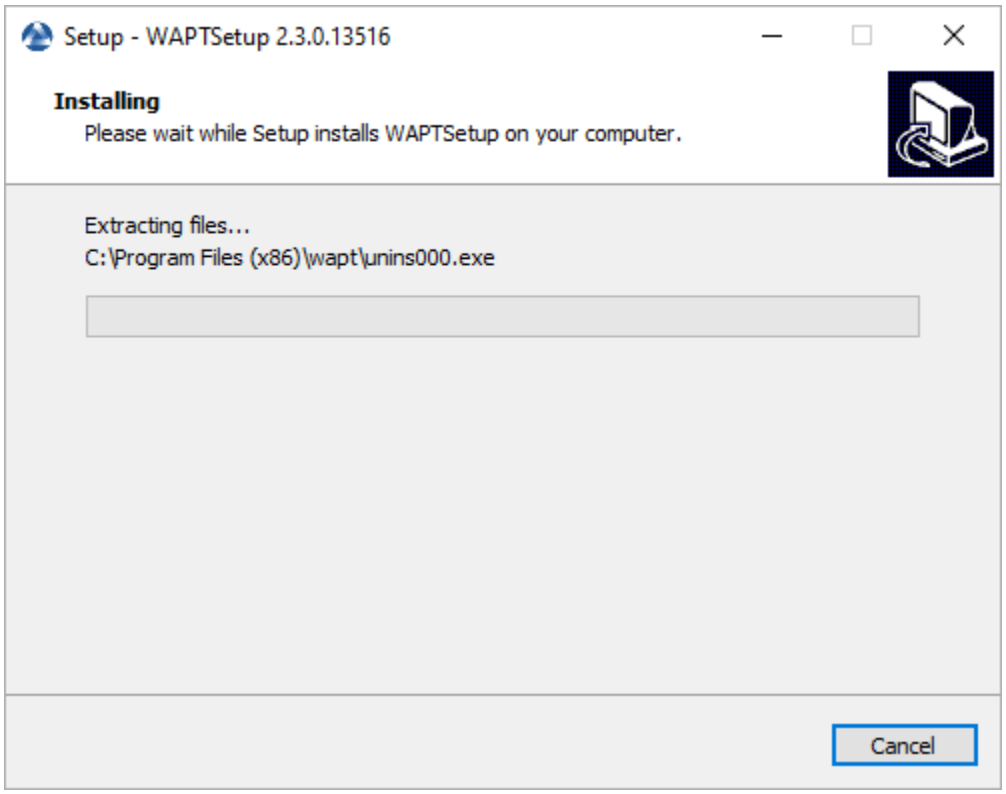


Figure1: Choix des options du programme d'installation pour le déploiement du serveur WAPT



- Cliquez sur le bouton *Installer* pour lancer l'installation, attendez que l'installation soit terminée.



- Cliquez sur *Terminer* pour fermer la fenêtre.

- Une fois terminé, votre serveur est prêt.

23.2.2 Mise à niveau de WAPT de 2.x à 2.3

Note: Avant de procéder à la mise à niveau, lisez les *exigences d'installation*.

WAPT 2.3 needs PostgreSQL 10 or above. If you have upgraded from a older Debian or Ubuntu version with PostgreSQL 9.6, be sure to follow the OS documentation to upgrade PostgreSQL to latest version.

Debian / Ubuntu

- Tout d'abord, mettez à jour la distribution.

```
apt update && apt upgrade -y
apt install apt-transport-https lsb-release gnupg
```

- Ajouter ou mettre à jour le dépôt de paquets Debian, importer la clé GPG du dépôt et installer les paquets du serveur WAPT.

```
wget -O - https://wapt.tranquil.it/$(lsb_release -is)/tiswapt-pub.gpg | apt-key add -
echo "deb https://wapt.tranquil.it/$(lsb_release -is)/wapt-2.3/ $(lsb_release -c -s) main" > /etc/
↪apt/sources.list.d/wapt.list
```

- Mettre à jour le dépôt et installer les paquets.

```
export DEBIAN_FRONTEND=noninteractive
apt update
apt install tis-waptserver tis-waptsetup -y
unset DEBIAN_FRONTEND
```

- Lancer l'étape de post-configuration *étape de post-configuration*

RedHat et dérivés

- Tout d'abord, mettez à jour la distribution.

```
yum update -y
yum install epel-release redhat-lsb-core -y
```

- Ajouter ou mettre à jour le dépôt de paquets Debian, importer la clé GPG du dépôt et installer les paquets du serveur WAPT.

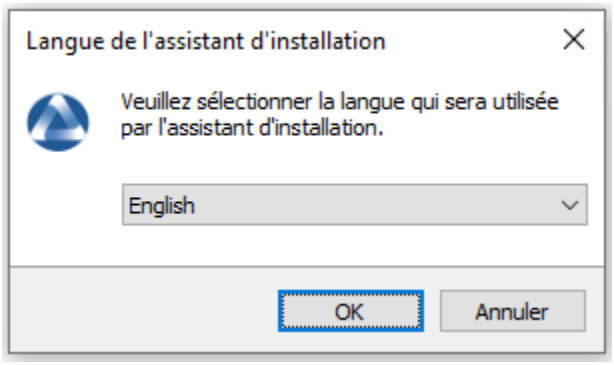
```
RH_VERSION=$(cat /etc/system-release-cpe | awk -F: '{ print $5}')

cat > /etc/yum.repos.d/wapt.repo <<EOF
[wapt]
name=WAPT Server Repo
baseurl=https://wapt.tranquil.it/redhat${RH_VERSION}/wapt-2.3/
enabled=1
gpgcheck=1
EOF

wget -q -O /tmp/tranquil_it.gpg "https://wapt.tranquil.it/redhat${RH_VERSION}/RPM-GPG-KEY-TISWAPT-$
↪{RH_VERSION}"; rpm --import /tmp/tranquil_it.gpg
```

- Et enfin, mettre à jour le serveur WAPT.

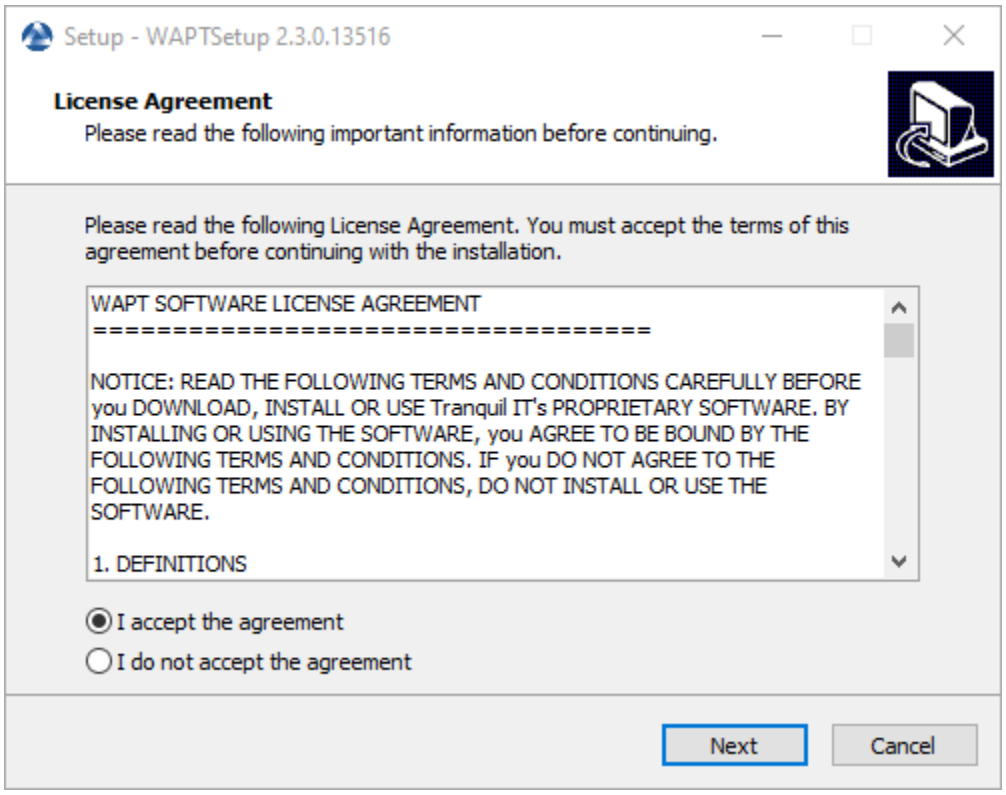
```
yum install tis-waptserver tis-waptsetup cabextract -y
```

Windows

- Télécharger et exécuter `waptserversetup.exe`.
- Choisir la langue de l'installeur WAPT.



- Acceptez la licence publique GNU et cliquez sur *Suivant* pour passer à l'étape suivante.



- Acceptez la licence publique GNU et cliquez sur *Suivant* pour passer à l'étape suivante.
- Choisissez le répertoire d'installation (laissez la valeur par défaut) et cliquez sur *Suivant* pour passer à l'étape suivante.

- Sélectionnez une tâche supplémentaire si nécessaire.

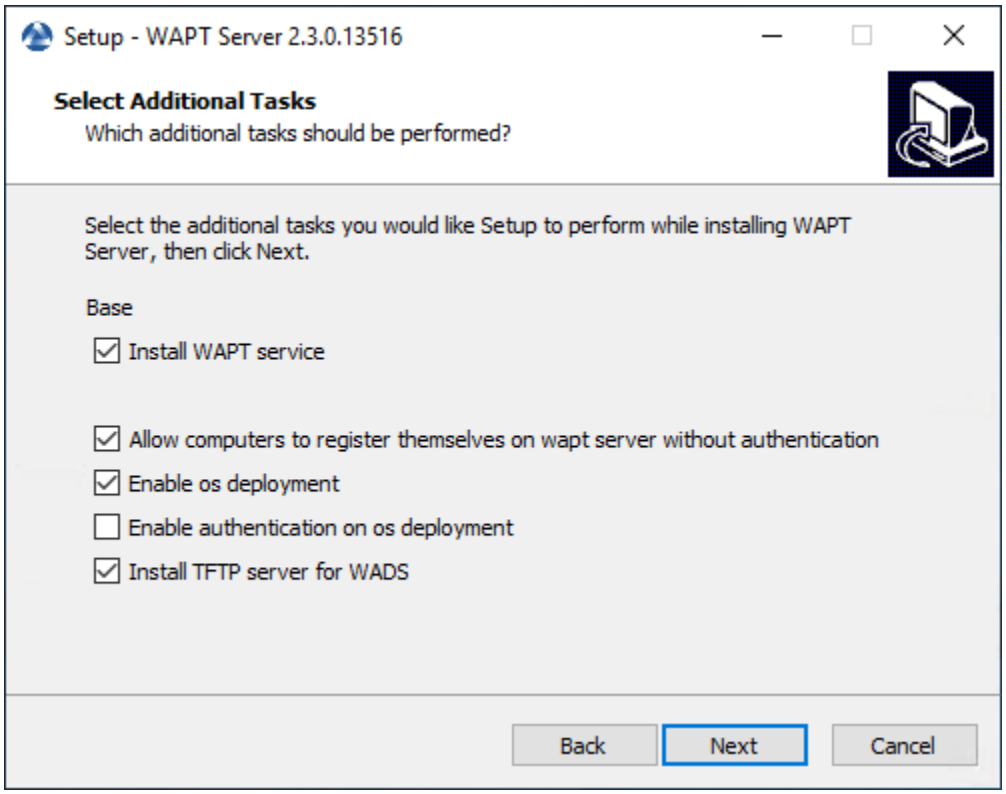
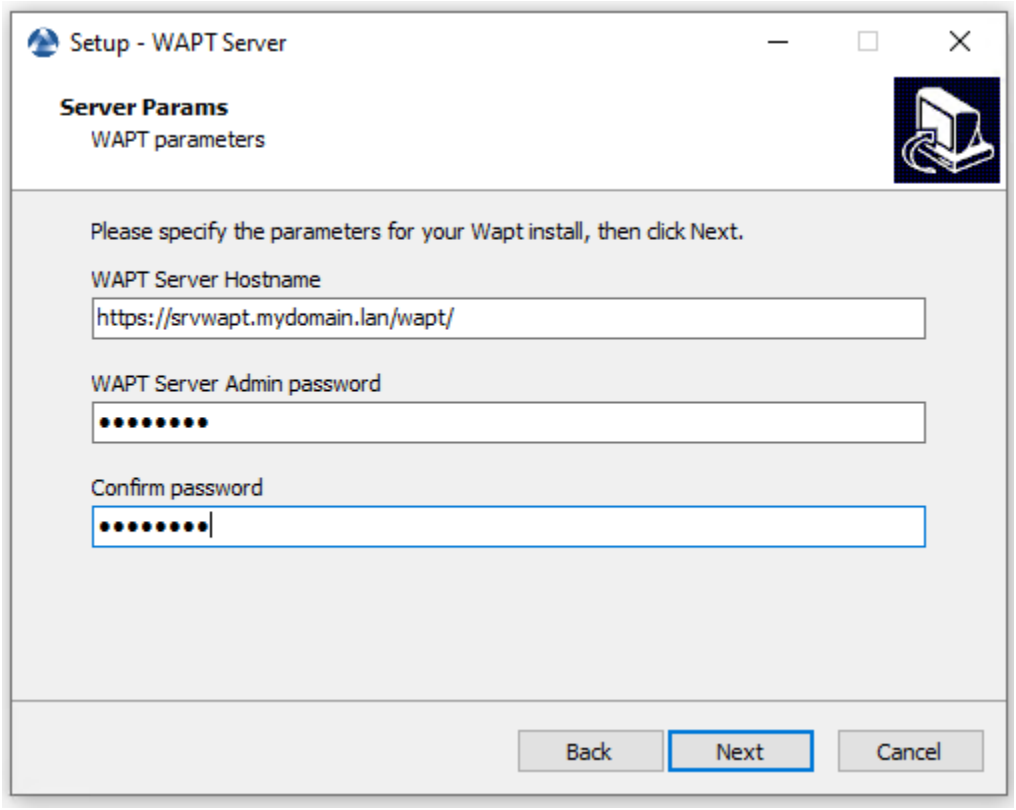
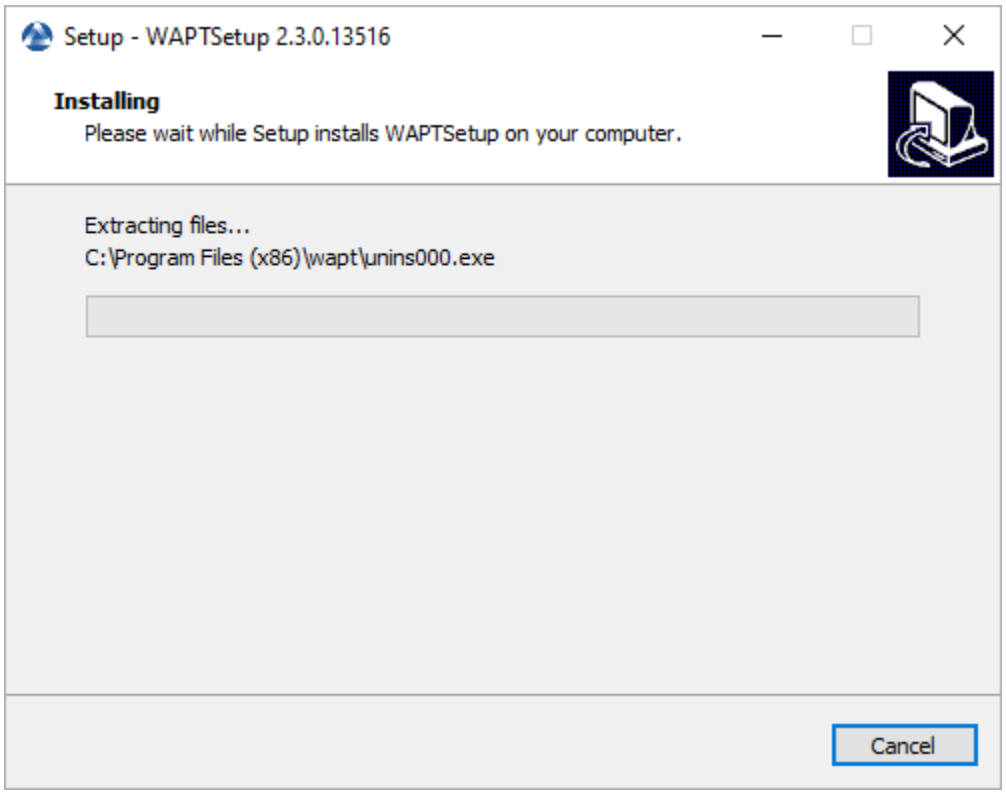


Figure2: Choix des options du programme d'installation pour le déploiement du serveur WAPT

- Modifiez le mot de passe du serveur WAPT si nécessaire, puis appuyez sur *Suivant*.



- Cliquez sur le bouton *Installer* pour lancer l'installation, attendez que l'installation soit terminée.



- Cliquez sur *Terminer* pour fermer la fenêtre.

Attention: **NE PAS** utiliser la console WAPT sur le serveur WAPT. **N'installez PAS** et n'exécutez pas vos outils de développement de paquets WAPT sur le serveur WAPT.

Le serveur WAPT sur votre serveur ou station de travail Windows est prêt.

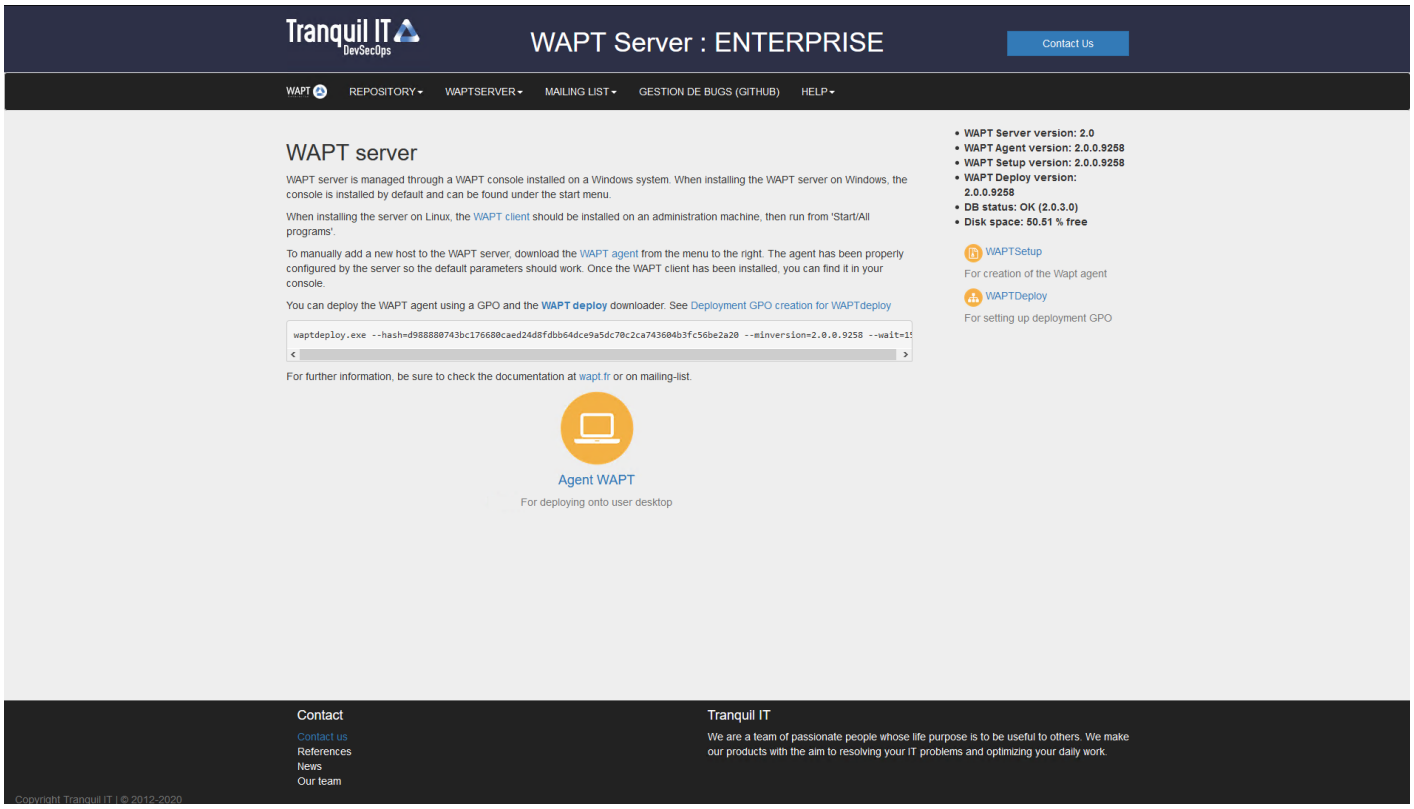


Figure3: L'interface du serveur WAPT dans un navigateur web

Votre serveur est maintenant prêt. Vous pouvez maintenant consulter la documentation sur *Installation de la console de gestion WAPT*.

23.3 Mise à niveau de WAPT de 1.8.2 à 2.3

The changement between WAPT 1.8.2 (**not possible from a older version, please upgrade to 1.8.2 beforehand**) and 2.3 are numerous. First of all, 1.8.2 was in Python2, we changed to Python3. A lot of new features are available too (essentially in Enterprise version).

23.3.1 Before upgrading

Ensure that *installation requirements* are met.

Backup your WAPT Private and Public certificates which allows you to deploy your WAPTAgent and packages. Usually, it is located in `C:\private` on your computer where the WAPT Console is set. If you don't remember what this key is please refer to the section about *generating the Administrator's certificate for signing WAPT packages* for better understandment.

In this documentation, your WAPT certificate's name will be **wapt-private.crt**.

23.3.2 Upgrading

Debian / Ubuntu

Note: Si vous êtes sous Debian9 Stretch, vous devez d'abord mettre à niveau vers Debian10 Buster avant de mettre à niveau vers WAPT 2.x. **Le Serveur WAPT 2.x n'est pas disponible pour Debian9**.

Il est même recommandé de mettre à niveau vers Debian 11 Bullseye. Dans ce cas, il faut passer de Debian 9 => Debian 10 => Debian 11.

- Tout d'abord, mettez à jour la distribution.

```
apt update && apt upgrade -y
apt install apt-transport-https lsb-release gnupg
```

- Mettre à jour le dépôt et installer les paquets.

```
wget -O - https://wapt.tranquil.it/$(lsb_release -is)/tiswapt-pub.gpg | apt-key add -
echo "deb https://wapt.tranquil.it/$(lsb_release -is)/wapt-2.3/ $(lsb_release -c -s) main" > /etc/
↪apt/sources.list.d/wapt.list
```

- Mettre à jour le dépôt et installer les paquets.

```
export DEBIAN_FRONTEND=noninteractive
apt update
apt install tis-waptserver tis-waptsetup -y
unset DEBIAN_FRONTEND
```

- Lancer *l'étape de post-configuration*.

RedHat et dérivés

- Tout d'abord, mettez à jour la distribution.

```
yum update -y
yum install epel-release -y
```

- Ajouter ou mettre à jour le dépôt de paquets Debian, importer la clé GPG du dépôt et installer les paquets du serveur WAPT.

```
RH_VERSION=$(cat /etc/system-release-cpe | awk -F: '{ print $5}')

cat > /etc/yum.repos.d/wapt.repo <<EOF
[wapt]
```

(suite sur la page suivante)

(suite de la page précédente)

```
name=WAPT Server Repo
baseurl=https://wapt.tranquil.it/redhat${RH_VERSION}/wapt-2.3/
enabled=1
gpgcheck=1
EOF

wget -q -O /tmp/tranquil_it.gpg "https://wapt.tranquil.it/redhat${RH_VERSION}/RPM-GPG-KEY-TISWAPT-$
↪{RH_VERSION}"; rpm --import /tmp/tranquil_it.gpg
```

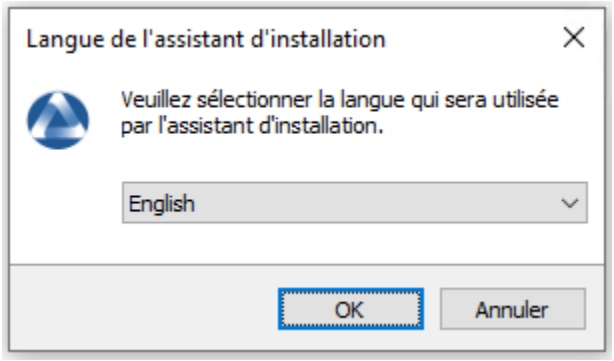
- Et enfin, mettre à jour le serveur WAPT.

```
yum install tis-waptserver tis-waptsetup cabextract -y
```

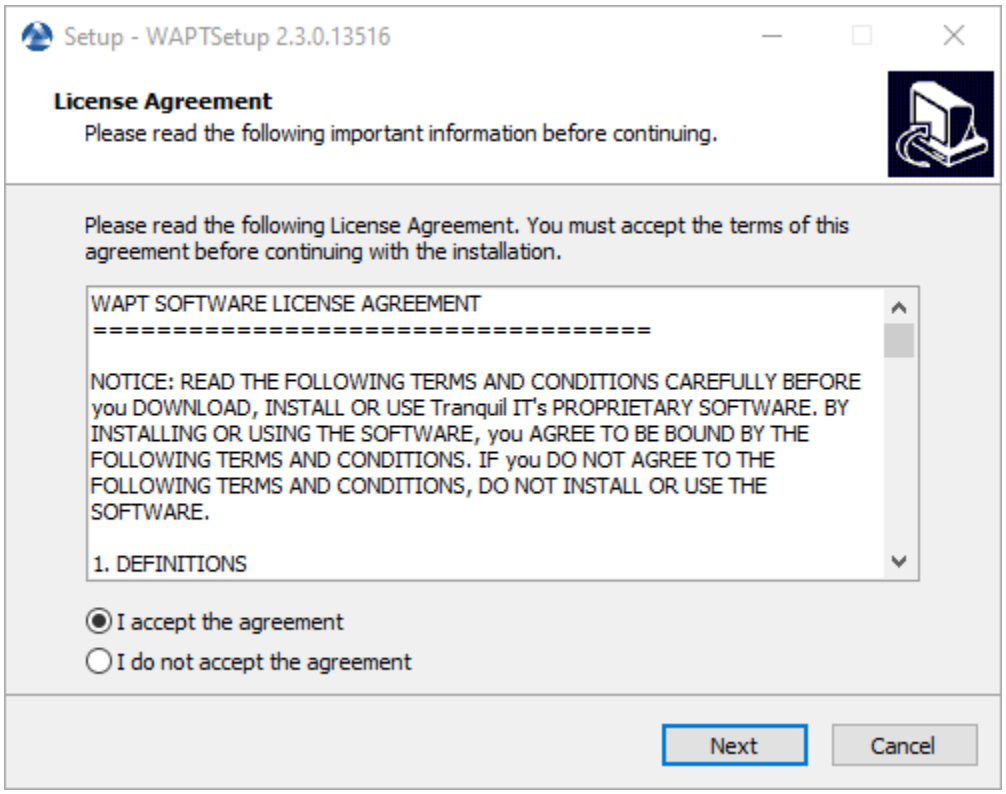
- Lancer *l'étape de post-configuration*.

Windows

- Télécharger et exécuter `waptserversetup.exe`.
- Choisir la langue de l'installeur WAPT.



- Acceptez la licence publique GNU et cliquez sur *Suivant* pour passer à l'étape suivante.



- Acceptez la licence publique GNU et cliquez sur *Suivant* pour passer à l'étape suivante.
- Si un ancien dossier d'installation est trouvé, ce message apparaît. Cliquez sur *Oui* pour passer à l'étape suivante.

- Sélectionnez une tâche supplémentaire si nécessaire.
- Modifiez le mot de passe du serveur WAPT si nécessaire, puis appuyez sur *Suivant*.

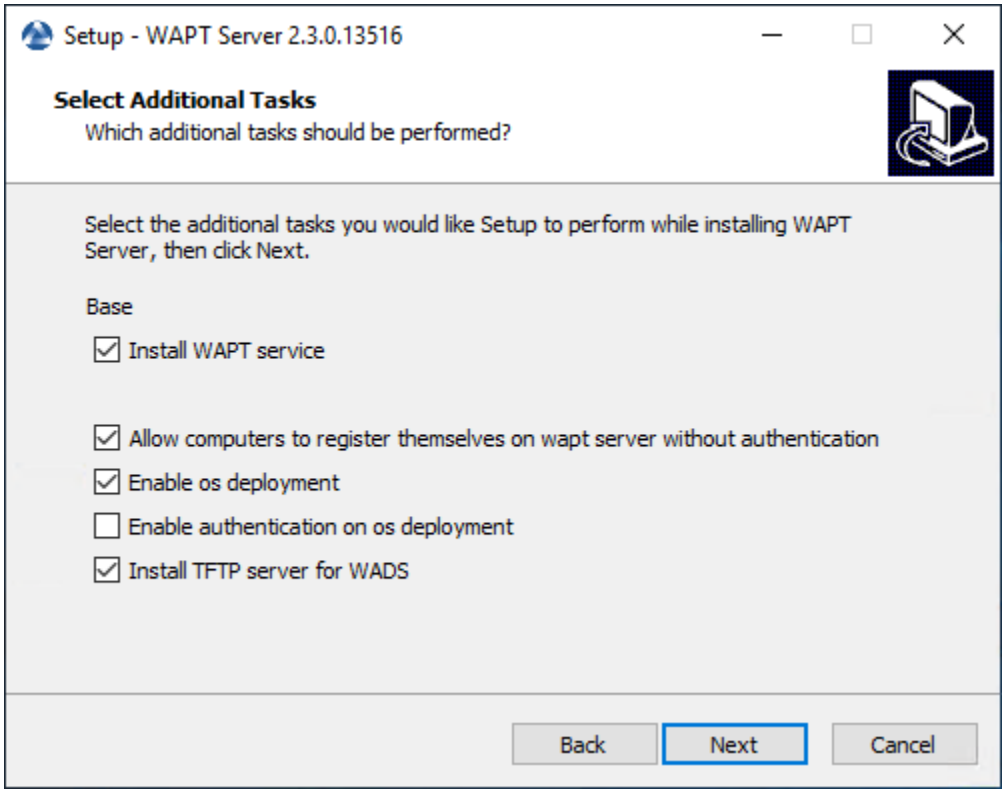
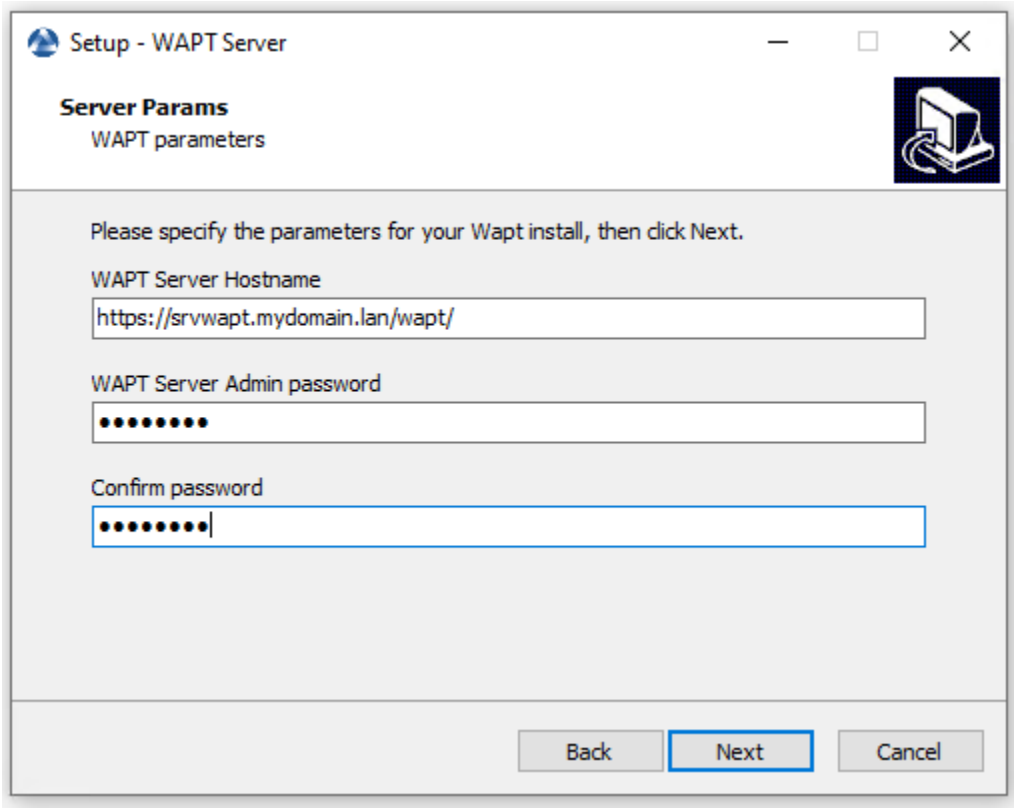
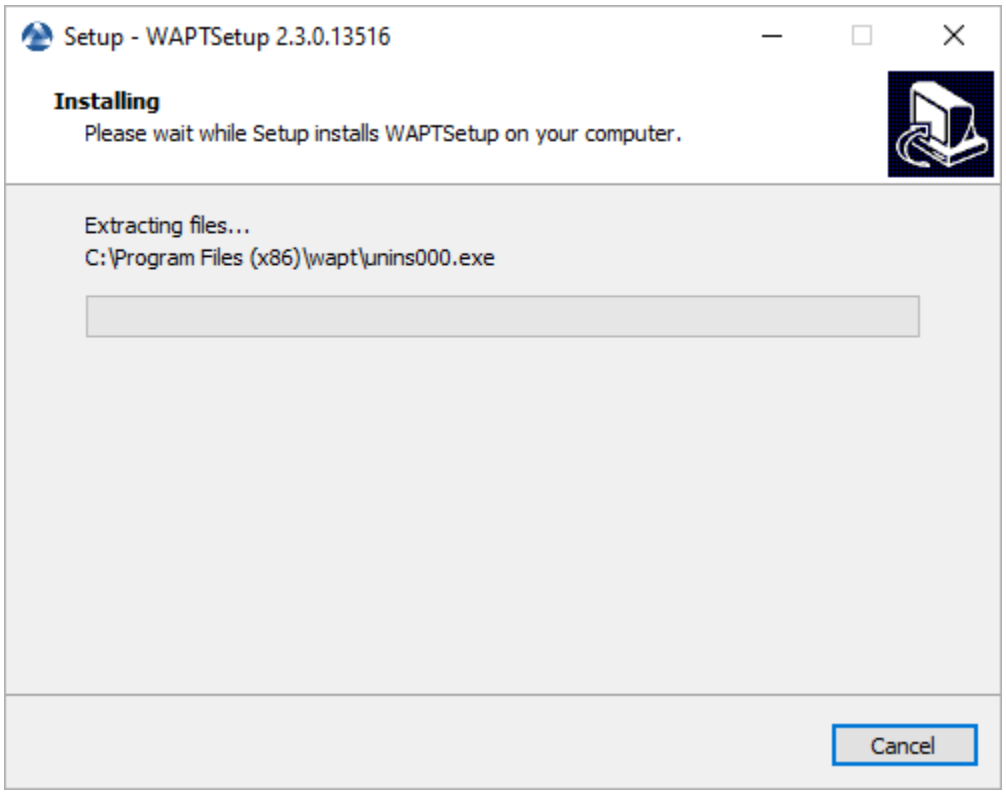


Figure4: Choix des options du programme d'installation pour le déploiement du serveur WAPT



- Cliquez sur le bouton *Installer* pour lancer l'installation, attendez que l'installation soit terminée.



- Cliquez sur *Terminer* pour fermer la fenêtre.

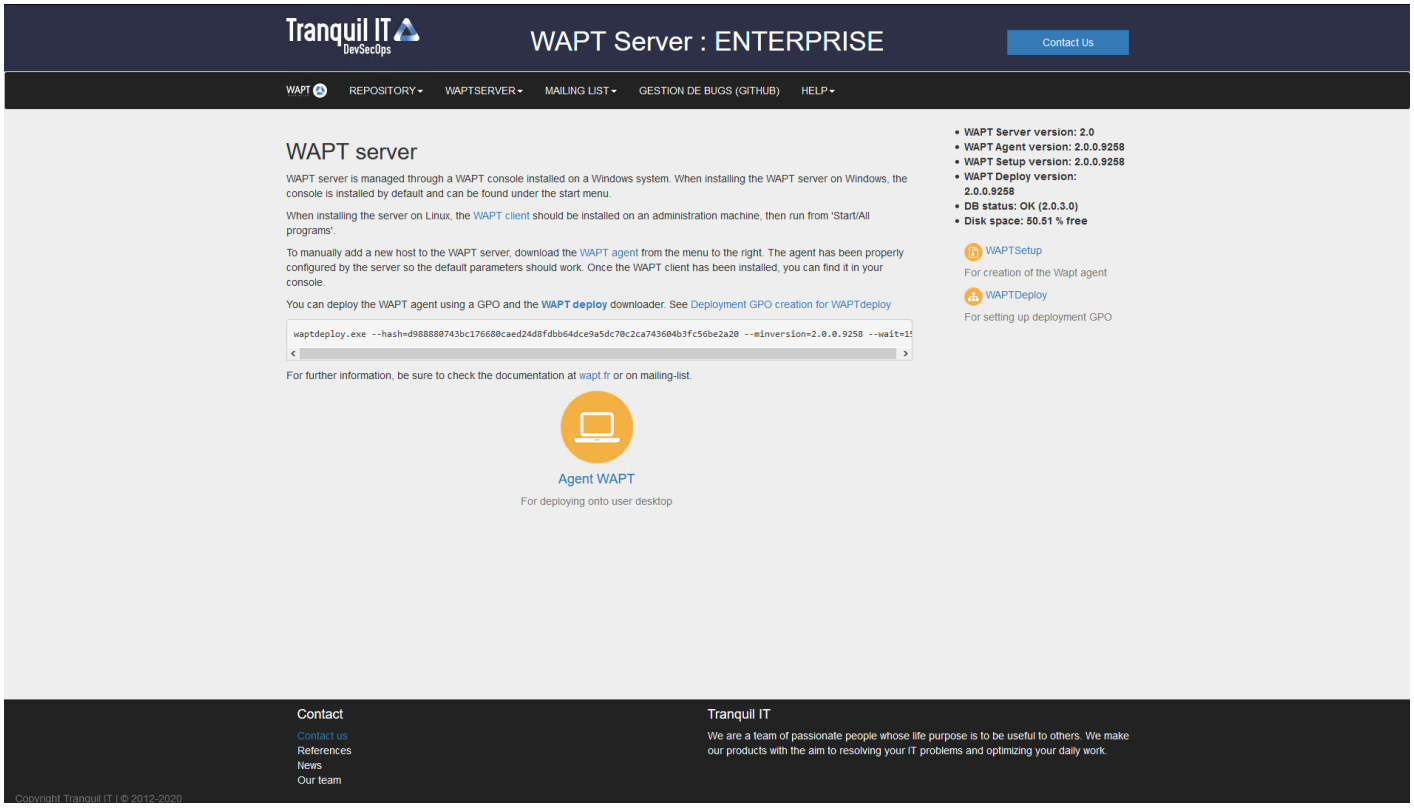


Figure5: L'interface du serveur WAPT dans un navigateur web

Le serveur WAPT sur votre serveur ou station de travail Windows est prêt.

Attention: **NE PAS** utiliser la console WAPT sur le serveur WAPT. **N'installez PAS** et n'exécutez pas vos outils de développement de paquets WAPT sur le serveur WAPT.

Your WAPT Server is now ready.

23.3.3 The WAPT management Console

To download the `waptsetup.exe` file, point your web browser to your waptserver url https://srvwapt.mydomain.lan, then click on the *WAPTSetup* link on the right-hand side of the WAPT Server web page. The WAPT Server home page only provides basic server status information and the download link for the WAPT Console.

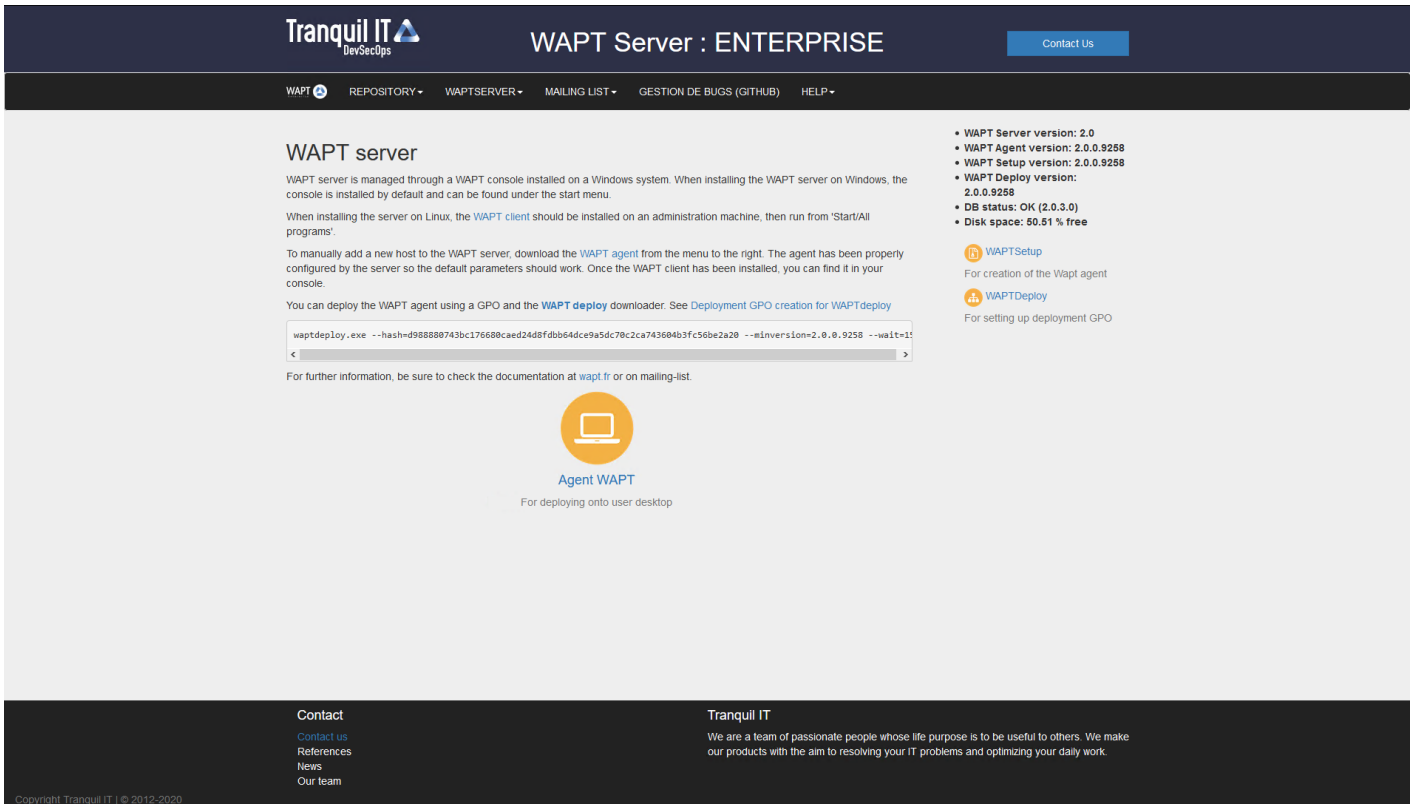
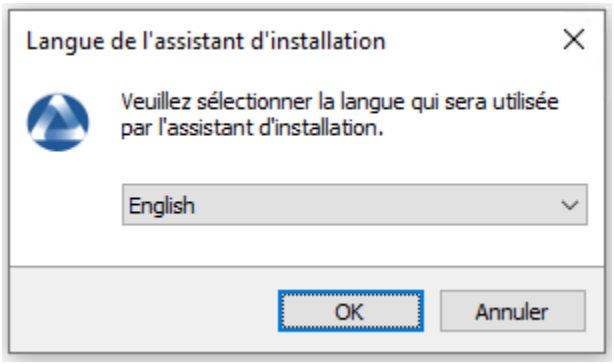


Figure6: L'interface du serveur WAPT dans un navigateur web

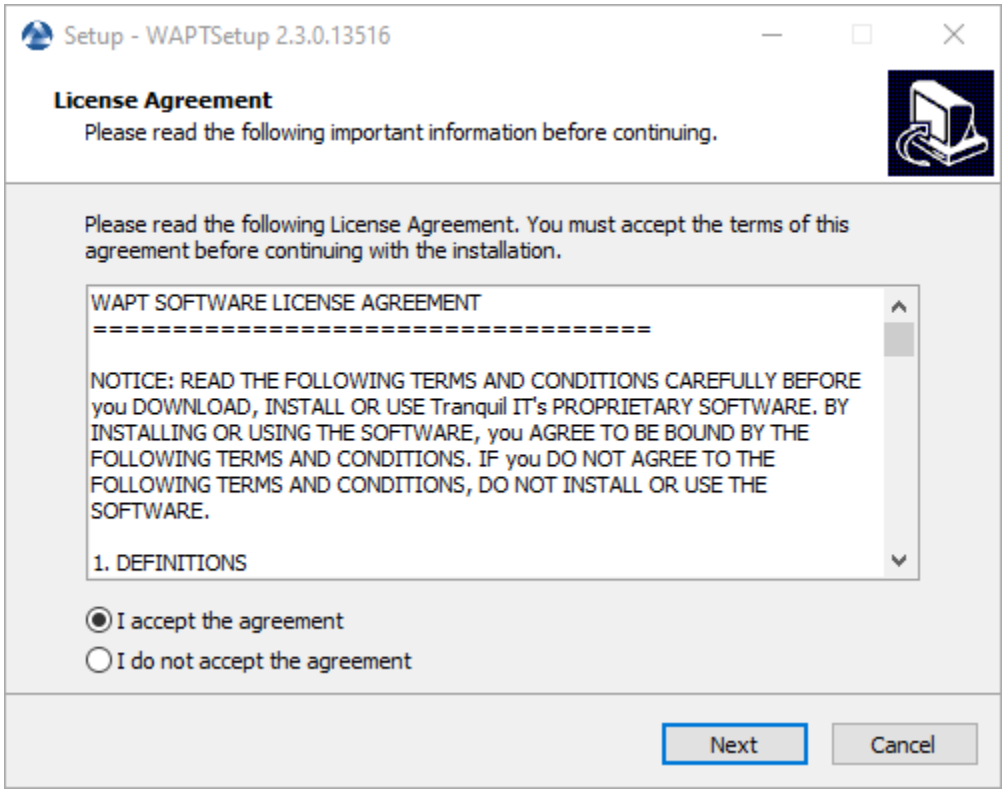
Installing the WAPT Agent on the Administrator's computer

Attention: If **the WAPT Agent** is not compiled and installed on your computer, you need to run de WAPT Agent installer to open and *configure the WAPT Console*.

- Start the executable installer as *Local Administrator* on the *Administrator*'s workstation.
- Choisir la langue de l'installeur WAPT.



- Acceptez la licence publique GNU et cliquez sur *Suivant* pour passer à l'étape suivante.



- Acceptez la licence publique GNU et cliquez sur *Suivant* pour passer à l'étape suivante.
- Choisissez une tâche supplémentaire (laissez la valeur par défaut).

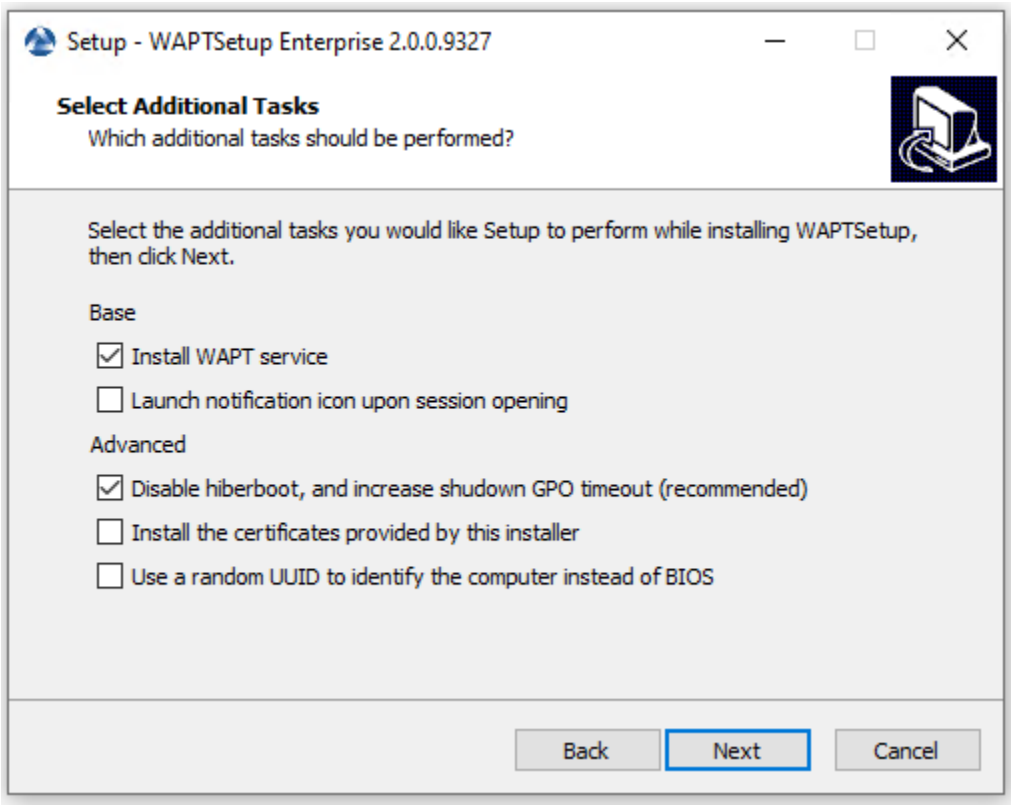


Figure7: Choosing the WAPT Agent installer options

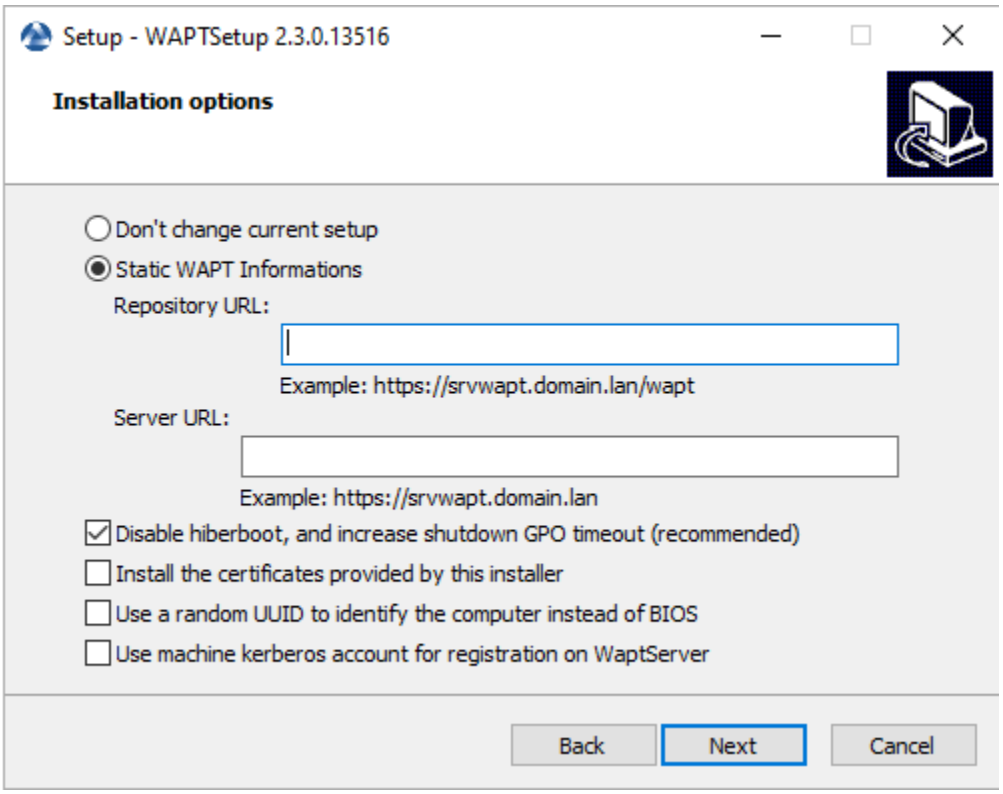
Table2: Available options of the WAPT Agent installer

Settings	Description	Default value
Install WAPT service checkbox	Enables the WAPT service on this computer.	Checked
Launch notification icon upon session opening checkbox	Launches the WAPT Agent in systray on host startup.	Not checked
Disable hiberboot, and increase shutdown GPO timeout (recommended) checkbox	Disables Windows fast startup for stability, it increases the timeout for the WAPT Exit utility.	Checked
Install the certificates provided by this installer checkbox	Installs Tranquil IT certificate on this computer.	Not checked
Use a random UUID to identify the computer instead of BIOS checkbox	For more information, check the documentation on *BIOS UUID bugs*	Not checked

- Set up the WAPT Server URL.

Fisrt installation

- Check *Static WAPT Informations* and set:
  - WAPT repository URL: http://srvwapt.mydomain.lan/wapt.
  - WAPT Server URL: https://srvwapt.mydomain.lan.

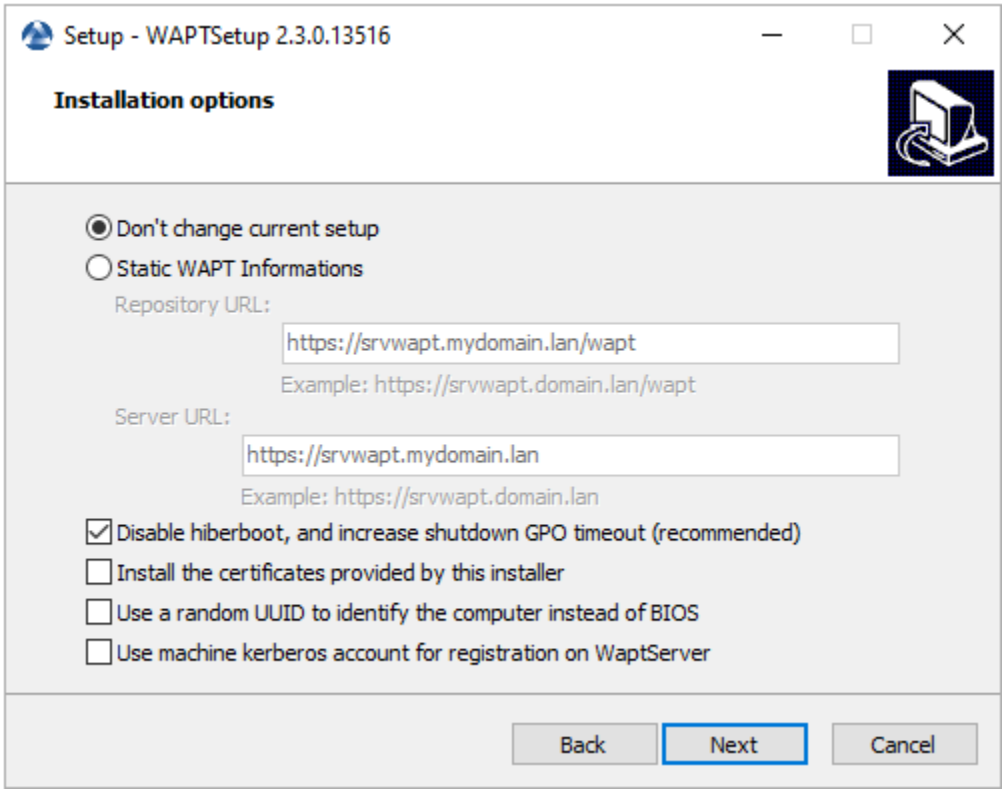


Choosing the WAPT repository and the WAPT Server

- Choose the WAPT repository and the WAPT Server; click *Next*.

Upgrade

- Check *Don't change current setup*, then click *Next*.



The WAPT repository and the WAPT Server are already set

- Get a summary of the WAPT Console installation.
- Click *Install* to launch the installation, wait for the installation to complete, then click on *Finish* (leave default options).

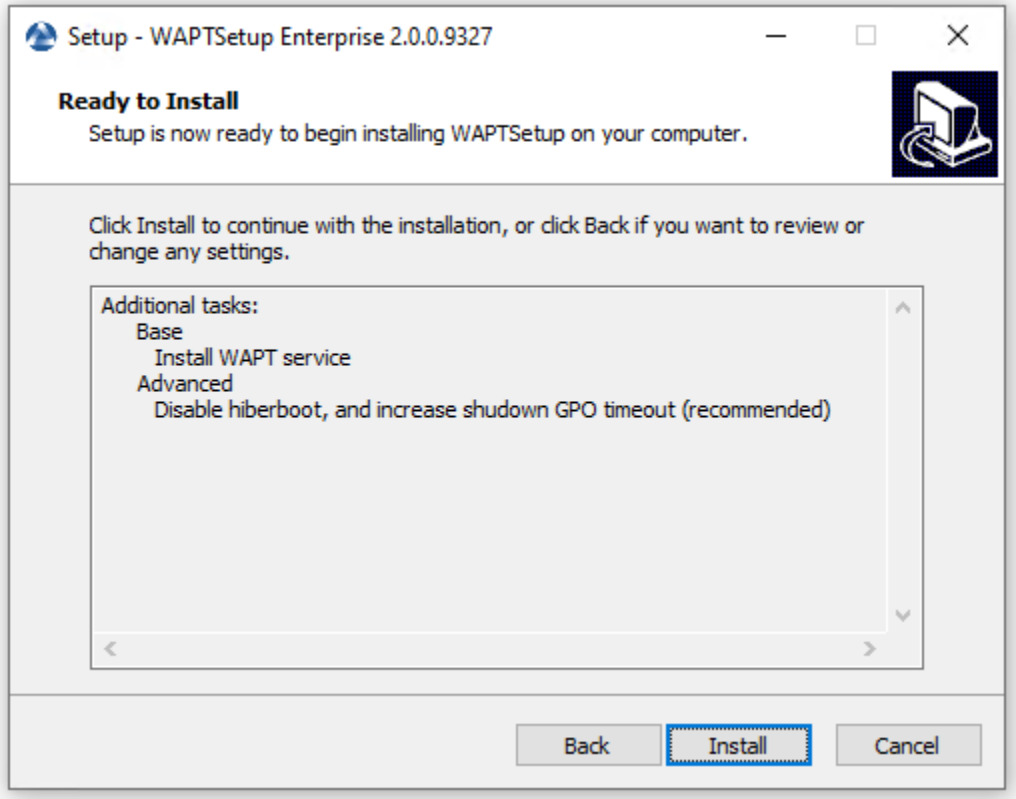
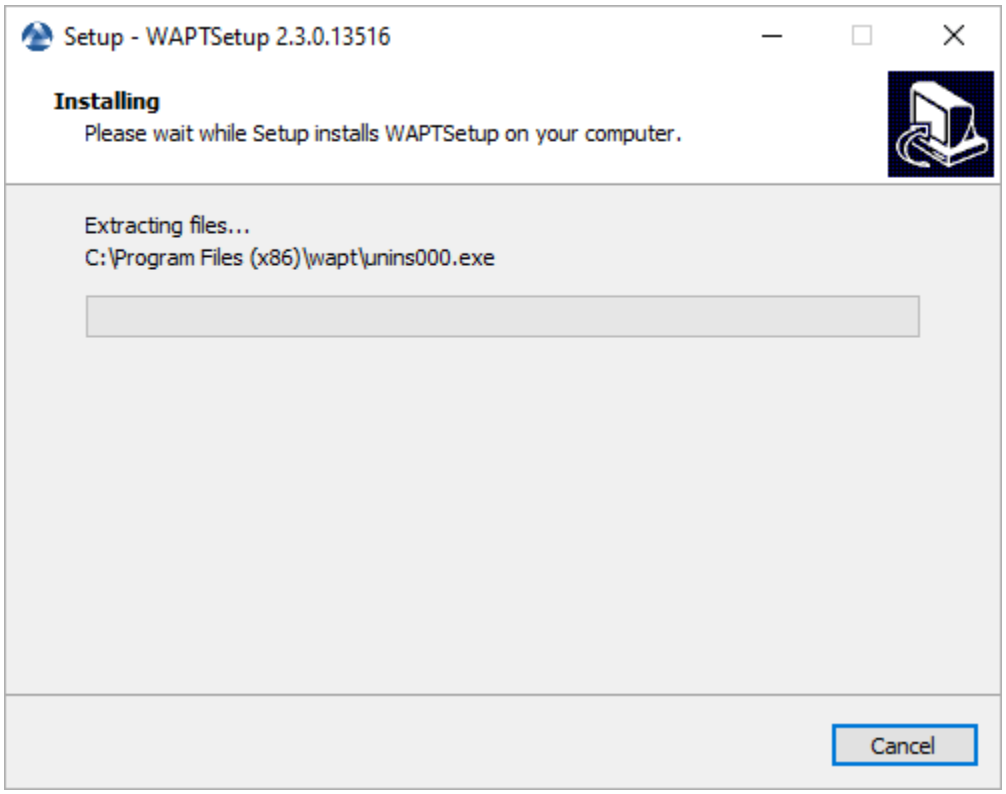
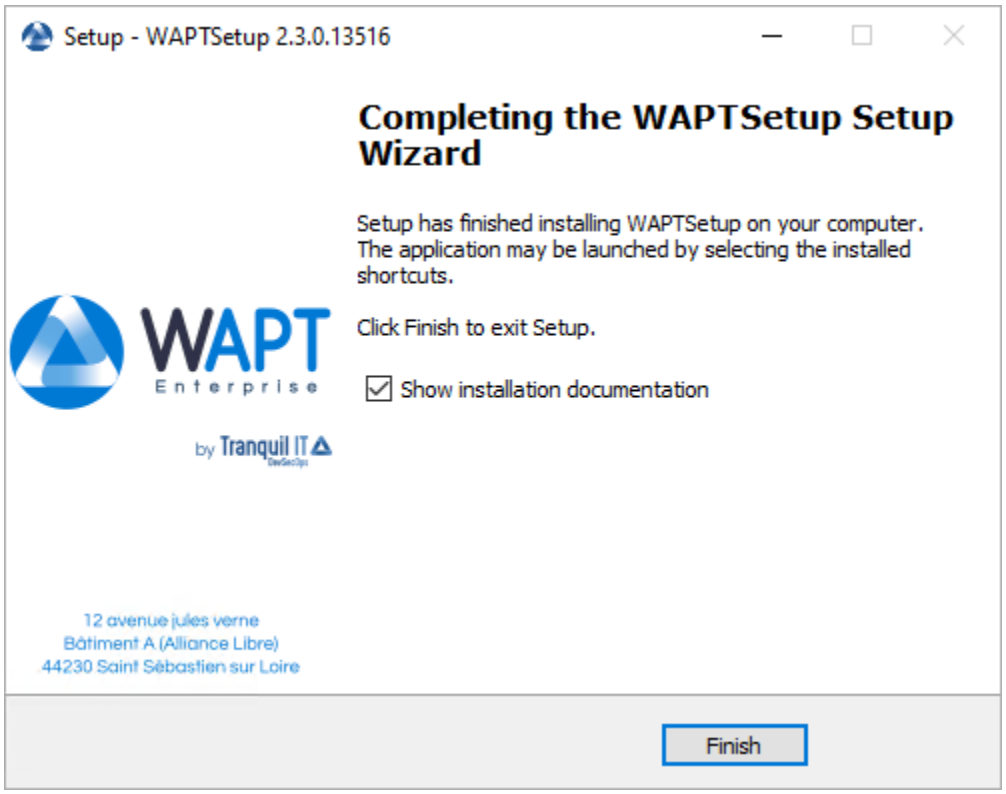


Figure8: Summary of the WAPT installation abstract





- Uncheck *Show installation documentation.*

Starting the WAPT Console

- Launch the WAPT Console:
  - By looking for the binary.

    `C:\Program Files (x86)\wapt\waptconsole.exe`
  - Or using the *Start* Menu.

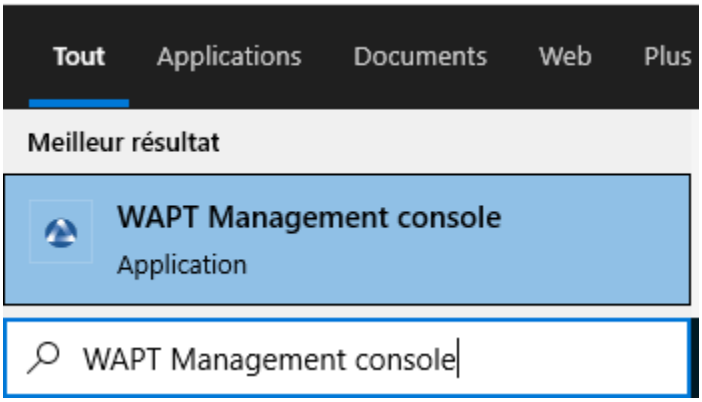


Figure9: Launching the WAPT Console from the Windows Start Menu

- Log into the WAPT Console with the *SuperAdmin* login and password.

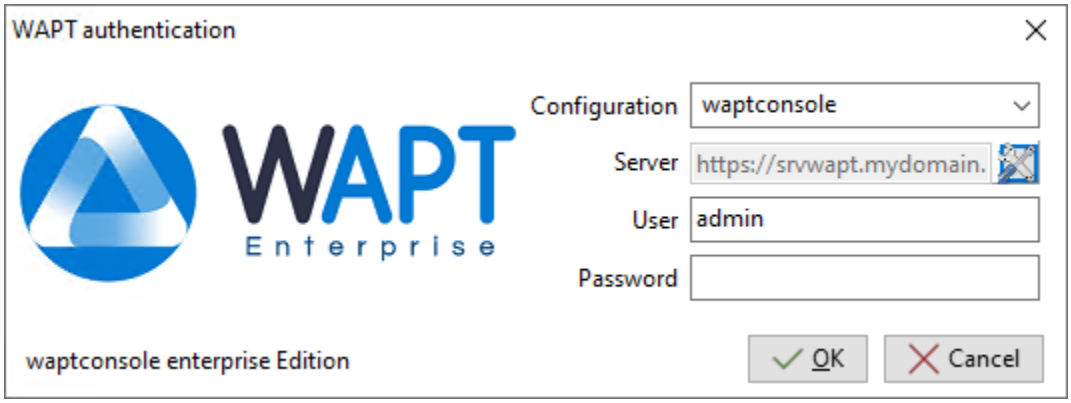


Figure10: The WAPT Console authentication window

If you have any issue logging into the WAPT Console, please refer to the FAQ: *Message d'erreur à l'ouverture de la console*.

It is recommended to launch the WAPT Console with a Local Administrator account to enable local debugging of WAPT packages.

For Enterprise version, it is possible to authenticate with *Active Directory*.

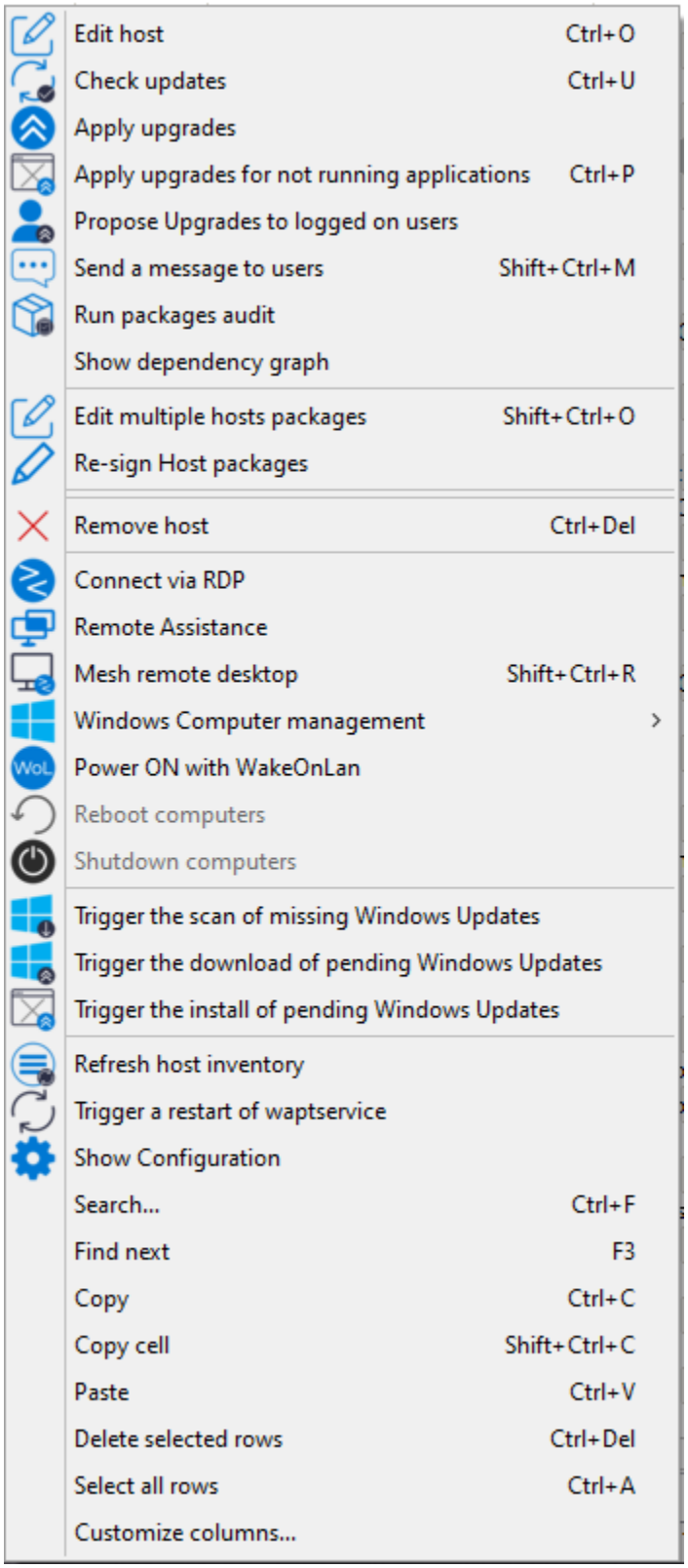
Danger: After the upgrade, please be sure your certificate (in this documentation **wapt-private.crt**) is still present at your WAPT install location : `C:\Program File (x86)\wapt\ssl`

Since you come from WAPT 1.8.2 which was in python2, you'll have to re-sign all your WAPT Packages *using the WAPT Console*, or *using the command line* (**only if you encounter package size issue with the WAPT Console way**).

23.3.4 Re-signing Host packages

Use the Administrator's certificate for re-signing packages.

- Select all host.
- Right-click on the selected hosts.



- Select *Re-sign Host packages*.

- Confirm re-signing the selected hosts.

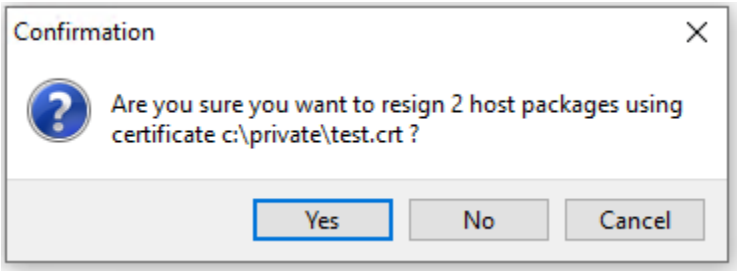


Figure11: Modal window for confirming re-signing the selection of hosts

- Then, enter you private key password.

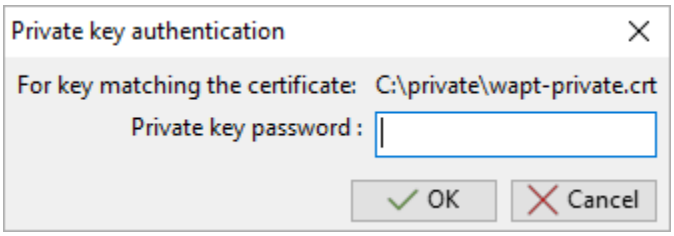


Figure12: Entering the password for unlocking the private key in the WAPT Console

- The selected WAPT *host* packages are now all re-signed using the new cryptographic method required with Python3.

23.3.5 Re-signing other types of WAPT package

- Open the repositories in your WAPT Console.
- Select all packages in the repository, then right-click on the selection.
- Select *Re-sign packages*.
- To launch the signature process, click on *Re-sign packages*.
- After processing, which may take some time, all packages will have been re-signed.

Attention: microsoft-office | 16.0.12325.20276-2 | PROD | ERROR | Access violation

If the error **Access violation** appear it may mean that the WAPT package is too big.

You can resign this packages *using the command line*.

And if it's still not working, you can still manually edit the package and visit *this procedure for signing large WAPT packages*.

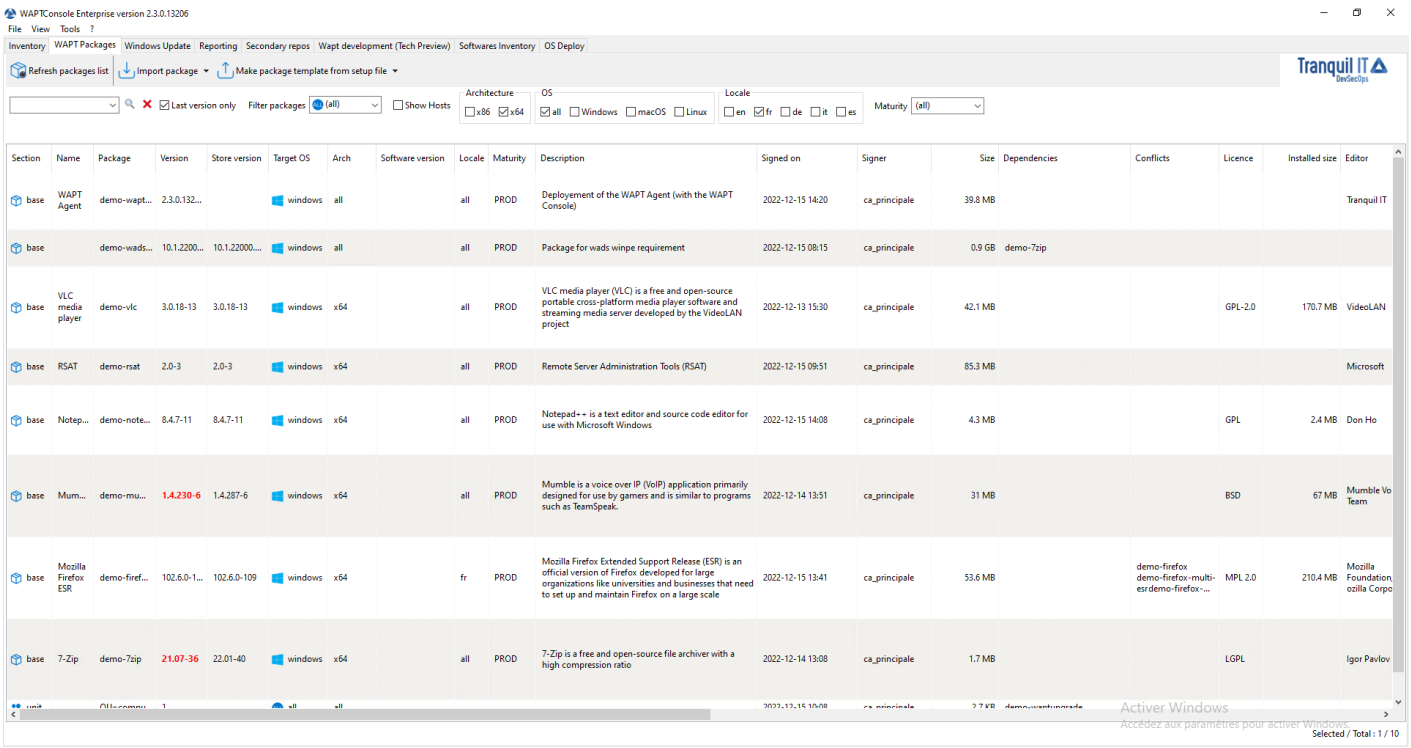


Figure13: Window showing the repositories available on the WAPT Console

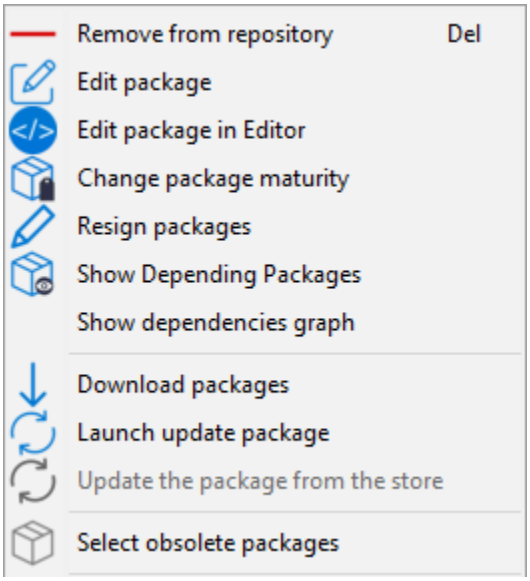


Figure14: Menu options for repositories

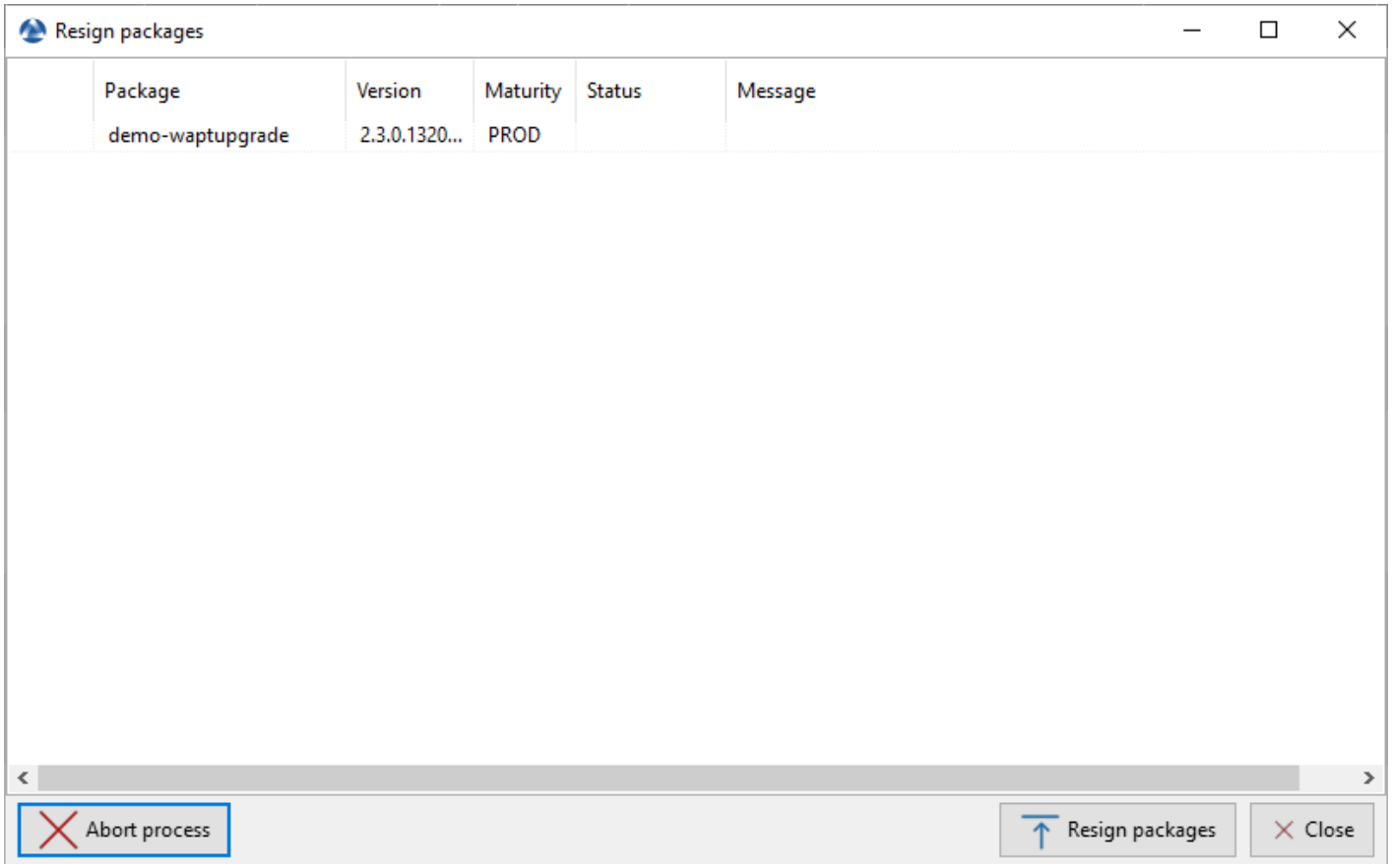


Figure15: Window for re-signing WAPT packages

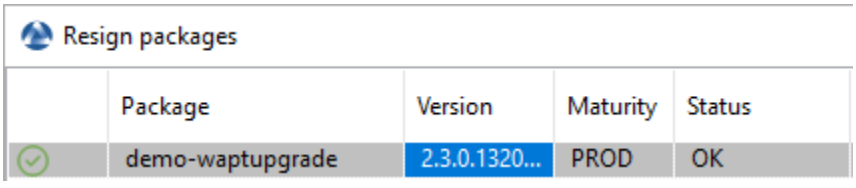


Figure16: Signature processing has ended successfully

CHAPTER 24

Sauvegarder le serveur WAPT

Pour sauvegarder votre serveur, suivez cette procédure. Des sauvegardes régulières sont recommandées.

24.1 Linux

- Arrêter les services liés à WAPT sur le serveur.

```
systemctl stop wapttasks
systemctl stop waptserver
systemctl stop nginx
```

- Sauvegarder ces répertoires en utilisant un outil de sauvegarde (ex : **rsync**, **WInSCP**, etc..).

Debian / Ubuntu

```
/var/www/wapt/
/var/www/wapt-host/
/var/www/waptwua/
/var/www/wads/
/opt/wapt/conf/
/opt/wapt/waptserver/ssl/
```

Centos / RedHat

```
/var/www/html/wapt/
/var/www/html/wapt-host/
/var/www/html/waptwua/
/var/www/html/wads/
/opt/wapt/conf/
/opt/wapt/waptserver/ssl/
```

Indication: Si vous utilisez Kerberos pour authentifier les machines et les utilisateurs, enregistrez également le fichier `keytab`. Le fichier `keytab` est situé dans le dossier `nginx`.

- Sauvegarder la base de données PostgreSQL en utilisant l'utilitaire **pg_dumpall** (adaptez le nom du fichier à vos besoins).

```
sudo -u postgres pg_dumpall > /tmp/backup_wapt.sql
```

- Redémarrer les services liés à WAPT sur le serveur.

```
systemctl start wapttasks
systemctl start waptserver
systemctl start nginx
```

24.2 Windows

- Arrêter les services liés à WAPT sur le serveur.

```
net stop wapttasks
net stop waptserver
net stop waptnginx
```

- Sauvegarder le dossier du dépôt WAPT sur une destination de sauvegarde distante.

```
C:\wapt\conf
C:\wapt\waptserver\repository\wapt
C:\wapt\waptserver\repository\wapt-host
C:\wapt\waptserver\repository\waptwua
C:\wapt\waptserver\repository\wads
C:\wapt\waptserver\nginx\ssl
```

- Sauvegarder la base de données PostgreSQL avec **pg_dump.exe**.

```
"C:\wapt\waptserver\pgsql-9.6\bin\pg_dumpall.exe" -U postgres -f C:\backup_wapt.sql
```

- Redémarrer les services liés à WAPT sur le serveur.

```
net start wapttasks
net start waptserver
net start waptnginx
```

CHAPTER 25

Restauration du serveur WAPT

En cas de panne complète, redémarrez une installation standard du serveur WAPT sur votre serveur. Puis suivez cette procédure pour restaurer vos données.

25.1 Linux

- Arrêter les services liés à WAPT sur le serveur.

```
systemctl stop nginx
systemctl stop waptserver
systemctl stop wapttasks
```

- Restaurer les répertoires suivants.

Debian / Ubuntu

```
/var/www/wapt/
/var/www/wapt-host/
/var/www/waptwua/
/var/www/wads/
/opt/wapt/conf/
/opt/wapt/waptserver/ssl/
```

Centos / RedHat

```
/var/www/html/wapt/
/var/www/html/wapt-host/
/var/www/html/waptwua/
/var/www/html/wads/
```

(suite sur la page suivante)

(suite de la page précédente)

```
/opt/wapt/conf/
/opt/wapt/waptserver/ssl/
```

- Restaurer la base de données (adaptez le nom de votre fichier). La première commande **supprime** la base de données WAPT (si elle existe). Assurez-vous que votre fichier dump est correct avant de le supprimer !

```
sudo -u postgres psql -c "drop database wapt"
sudo -u postgres psql < /tmp/backup_wapt.sql
```

- Appliquer les droits de propriété aux dossiers restaurés.

Debian / Ubuntu

```
chown -R wapt:www-data /var/www/wapt/
chown -R wapt:www-data /var/www/wapt-host/
chown -R wapt:www-data /var/www/waptwua/
chown -R wapt:www-data /var/www/wads/
chown -R wapt /opt/wapt/conf/
chown -R wapt /opt/wapt/waptserver/ssl/
```

CentOS / RedHat

```
chown -R wapt:www-data /var/www/html/wapt/
chown -R wapt:www-data /var/www/html/wapt-host/
chown -R wapt:www-data /var/www/html/waptwua/
chown -R wapt:www-data /var/www/html/wads/
chown -R wapt /opt/wapt/conf/
chown -R wapt /opt/wapt/waptserver/ssl/
```

- Analyser les dépôts de paquets.

Debian / Ubuntu

```
wapt-scanpackages /var/www/wapt/
```

CentOS / RedHat

```
wapt-scanpackages /var/www/html/wapt/
```

- Redémarrer les services liés à WAPT sur le serveur.

```
systemctl start wapttasks
systemctl start waptserver
systemctl start nginx
```

25.2 Windows

- Arrêter les services liés à WAPT sur le serveur.

```
net start wapttasks
net start waptserver
net start waptnginx
```

- Restaurer les répertoires suivants.

```
C:\wapt\waptserver\repository\wapt
C:\wapt\waptserver\repository\wapt-host
C:\wapt\waptserver\repository\waptwua
C:\wapt\waptserver\repository\wads
C:\wapt\waptserver\conf
C:\wapt\waptserver\nginx\ssl
```

- Appliquer le droit total au dossier `C:\wapt\waptserver\repository` pour le groupe « Service Réseau ».
- Restaurez la base de données PostgreSQL avec **`pg_restore.exe`**.

```
"C:\wapt\waptserver\pgsql-9.6\bin\psql.exe" -f c:\backup_wapt.sql -U postgres
```

- Analyser les dépôts de paquets.

```
wapt-scanpackages "C:\wapt\waptserver\repository\wapt"
```

- Redémarrer les services liés à WAPT sur le serveur.

```
net start wapttasks
net start waptserver
net start waptnginx
```

CHAPTER 26

Utiliser la console WAPT

Pour installer et démarrer la console WAPT, allez cette documentation pour *installer la console WAPT*.

Note: Si vous avez passé l'étape de la création de l'agent WAPT, retournez à la documentation sur *la construction de l'installeur de l'agent WAPT*.

Sur votre **poste de gestion**, les agents sont affichés dans la console WAPT.

Note: Si un hôte n'apparaît pas dans la console après avoir installé l'agent WAPT, ouvrez une invite de commande Windows **`cmd.exe`** sur l'hôte et tapez **`wapt-get register`**.

26.1 Ajouter un paquet interdit à la machine

Si vous voulez ajouter un paquet directement sur la machine, il faut éditer le paquet machine.

Pour cela, vous avez 3 façons:

1. Double-clic sur la machine.
2. Clic-droit sur la machine puis *Edit host*.
3. Sélectionner une machine et utliser le bouton *Edit host*.

Puis, il suffit de glisser déposer le ou les paquet(s) désiré(s) et de valider.

Appuyez sur *Enregistrer* fait la même chose que faire un *update*.

Appuyer sur *Enregistrer et appliquer* revient à faire un *update* immédiatement suivi par un *upgrade*.

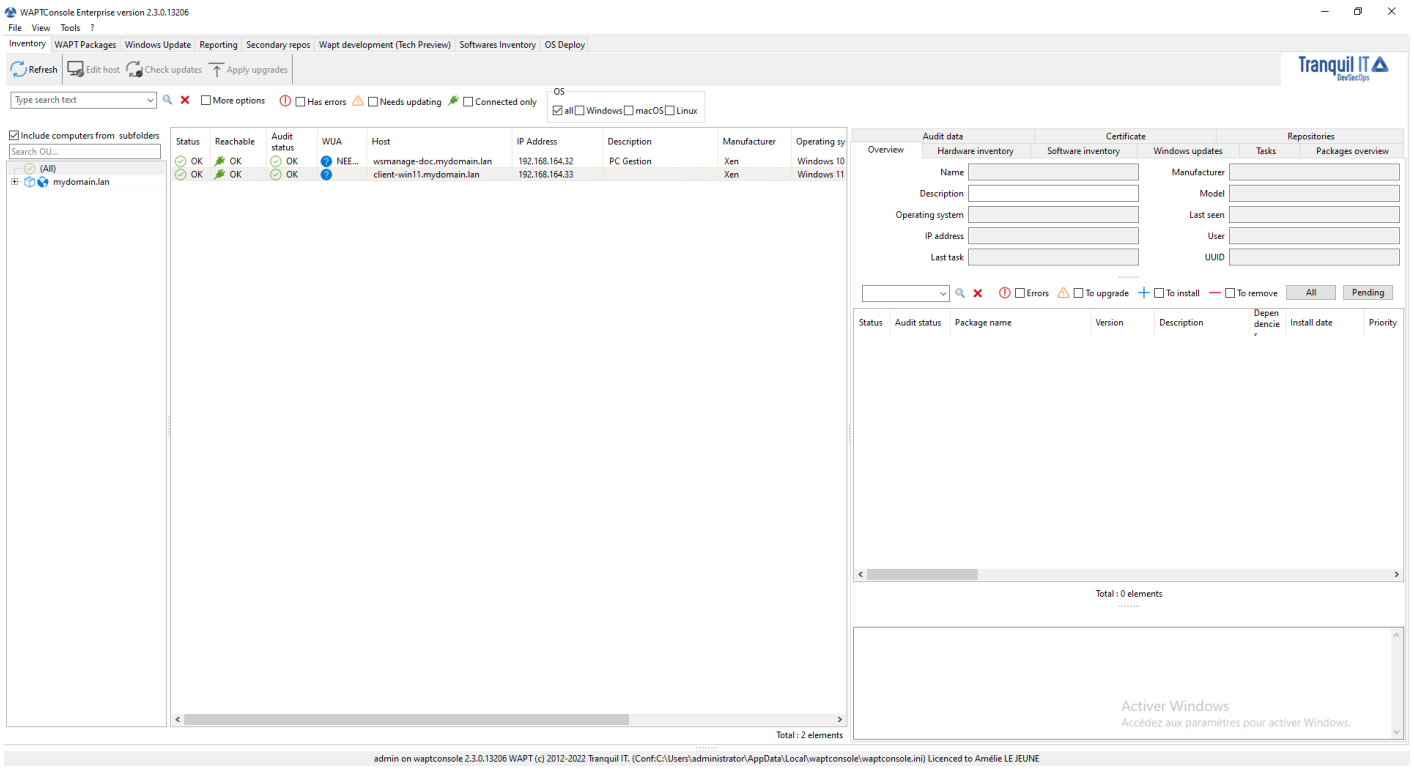


Figure1: Méthode pour ajouter un paquet WAPT à la machine

26.2 Ajouter ou supprimer un paquet WAPT sur plusieurs machines

Si vous voulez ajouter ou supprimer des paquets WAPT directement sur plusieurs machines, sélectionnez-les puis sur l'un d'entre eux *Richt-click → Edit mulitple hosts packages*. Vous pouvez ajouter des paquets en conflit et supprimer des paquets en conflit avec cette méthode également.

Une nouvelle fenêtre s'ouvre, vérifier le paquet souhaité puis cliquer sur *OK*.

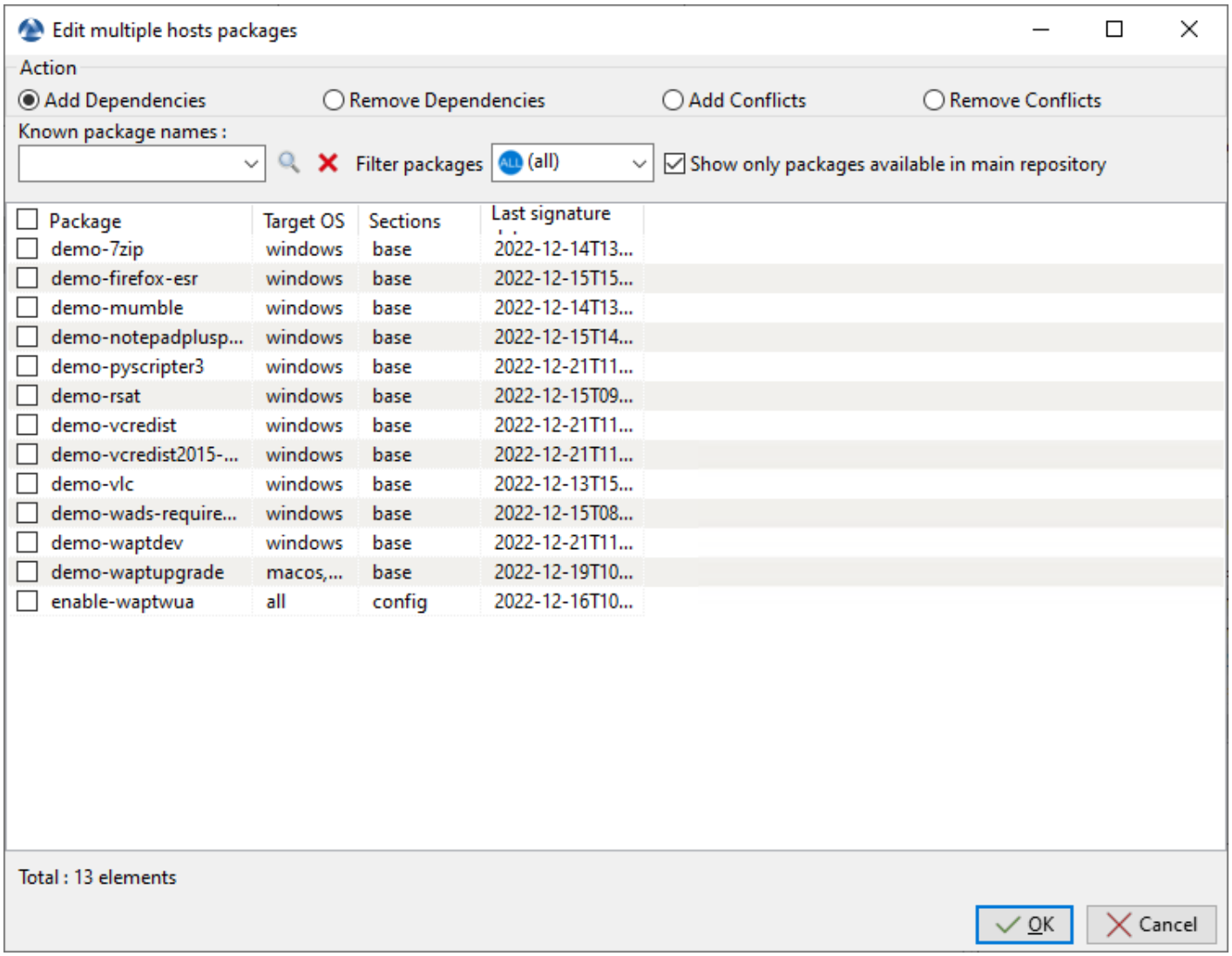
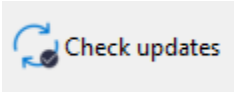


Figure2: Méthode pour ajouter ou retirer un paquet WAPT sur plusieurs machines

Après cela, faites un *update* immédiatement suivi d'un *upgrade* sur les machines.

26.3 Vérifier les mises a jour sur l'hote

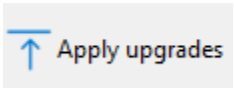


Ce bouton exécute 2 actions :

1. remonter l'état actuel de l'hôte au serveur
2. le serveur indique alors si l'hôte doit récupérer des mises à jour

Toutes modifications de configuration nécessite un *Check updates*.

26.4 Appliquer les maj sur l'hote



Ce bouton exécute les mises à jour en attente sur le poste.

> **Avertissement:** A utiliser avec précaution, cela forcera la fermeture des logiciels en cours d'utilisation.
>
> Vous pouvez utiliser a la place *Lancer les installations en attentes pour les applications non lancées* pour éviter toute perte de travail.

26.5 Envoi d'un message aux machines sélectionnées

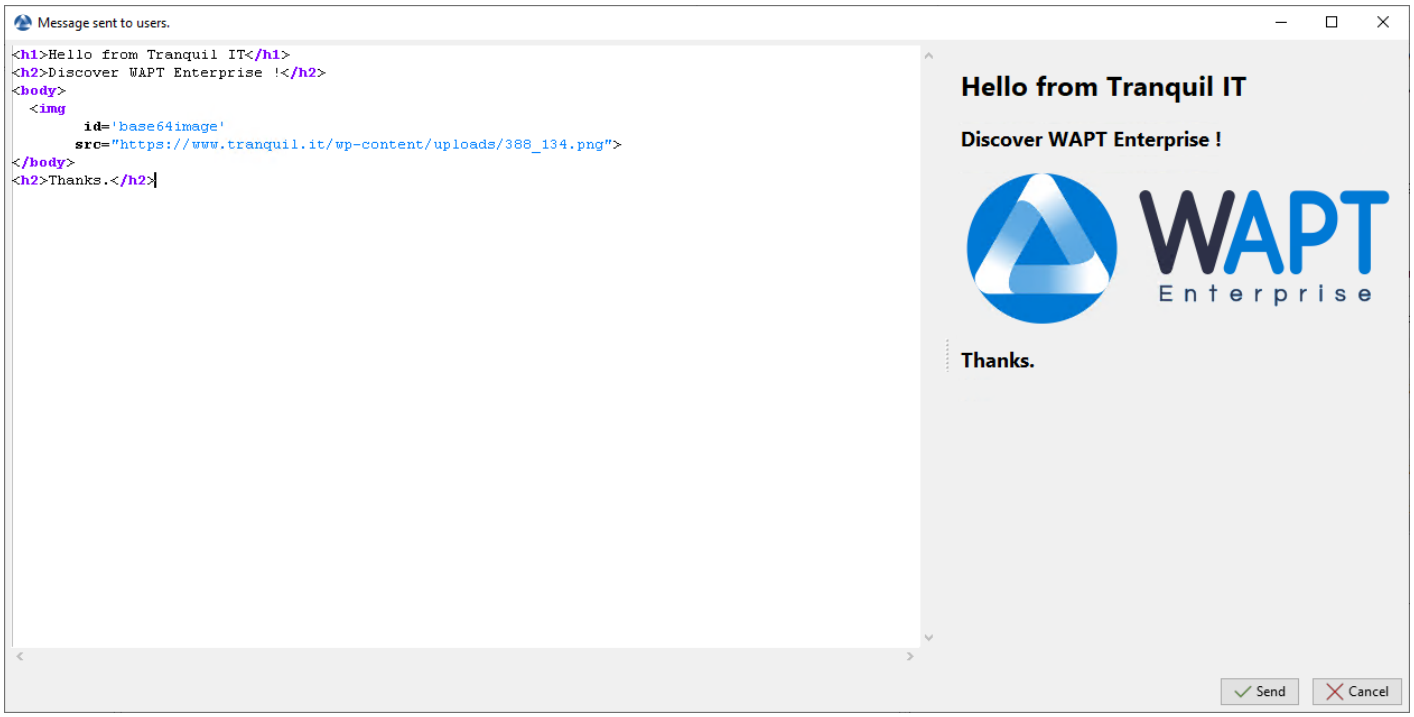
Avec l'option *Envoyer un message aux utilisateurs*, lorsque vous faites un clic droit sur les machines sélectionnées, vous pouvez envoyer un message au format html aux utilisateurs connectés.

Vous ne pouvez envoyer que du format html, pas de css ni de javascript. Il est possible d'envoyer des images, des gifs… n'importe quoi pourvu que vous puissiez l'encoder en html.

Voici un exemple de code pour envoyer un texte et une image. Les images et les gif doivent être encodés en base64, sinon vous pouvez utiliser l'identifiant *base64image*.

```
<h1>Hello from Tranquil IT</h1>
  <h2>Discover WAPT Enterprise !</h2>
   <body>
   <img
      id='base64image'
     src="https://www.tranquil.it/wp-content/uploads/388_134.png">
   </body>
  <h2>Thanks.</h2>
```

Lorsque vous créez votre message, vous avez un aperçu sur la droite.



Lorsque vous avez terminé, cliquez sur *Envoyer*, voici comment votre utilisateur le verra.

La première icône ci-dessus peut être personnalisée. Veuillez voir comment *utiliser une icône personnalisée pour votre message*.

26.6 Effectuer une recherche globale sur tous les hôtes

Effectuer des recherches globales avec tous les critères présentés ci-dessous est possible.

Choisir les filtres à cocher ou décocher.

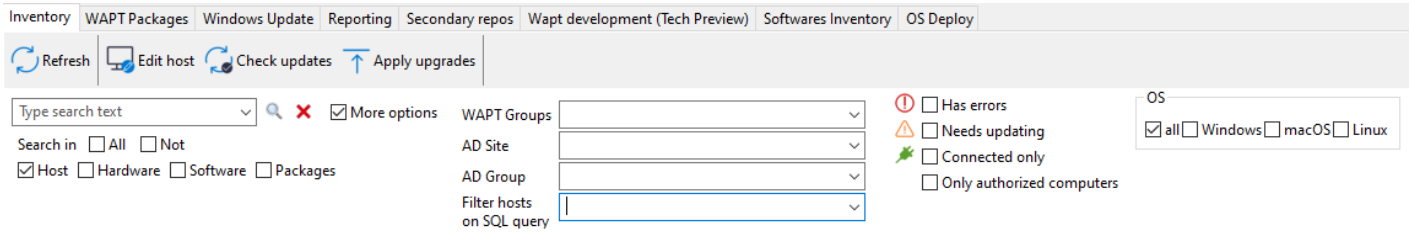


Figure3: Les fonctionnalités de recherches avancées dans la console WAPT

Table1: Choix des filtres

Options possibles	Description
Machine	La section *Host* dans l'onglet *Inventaire matériel* quand un hôte est sélectionné.
Matériel	La section *DMI* dans l'onglet *Inventaire matériel* lorsqu" un hôte est sélectionné.
Logiciel	La section *Inventaire logiciel* lorsqu'un hôte est sélectionné.
Paquets	Liste des paquets installés sur les hôtes sélectionnés.
Has errors	Rechercher uniquement les hôtes dont les tâches ne se sont pas correctement terminées.
Needs updating	Rechercher uniquement les hôte nécessitant une mise à jour.
Connectés seulement	Rechercher uniquement les hôtes connectés
Seuls les ordinateurs autorisés	Rechercher uniquement les hôtes autorisés par le certificat de l'utilisateur en cours de la Console WAPT.
WAPT Group	Filtrer les machines sur la base de leur appartenance / dépendance à un paquet WAPT de type *group*.
AD Site	Filtrer les machines sur la base de leur appartenance / dépendance à Site et Services Active Directory.
AD Group	Filtrer les machines sur la base de leur appartenance / dépendance à un groupe Active Directory.
OS	Filtrer les machines en fonction de l"OS (Operating System).

Indication: Les filtres fonctionnent avec des expressions régulières.

26.7 Afficher l'inventaire

Lorsque les agents WAPT s'enregistrent avec un **`register`**, ils envoient des informations au serveur WAPT.

Les informations affichées dans la console ne sont pas mises à jour en temps réel, vous devez rafraîchir l'affichage pour voir les nouveaux statuts et informations.

Cliquez sur le bouton *Refresh* ou appuyez sur `F5` sur le clavier.

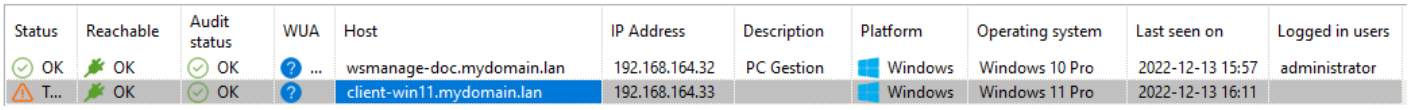
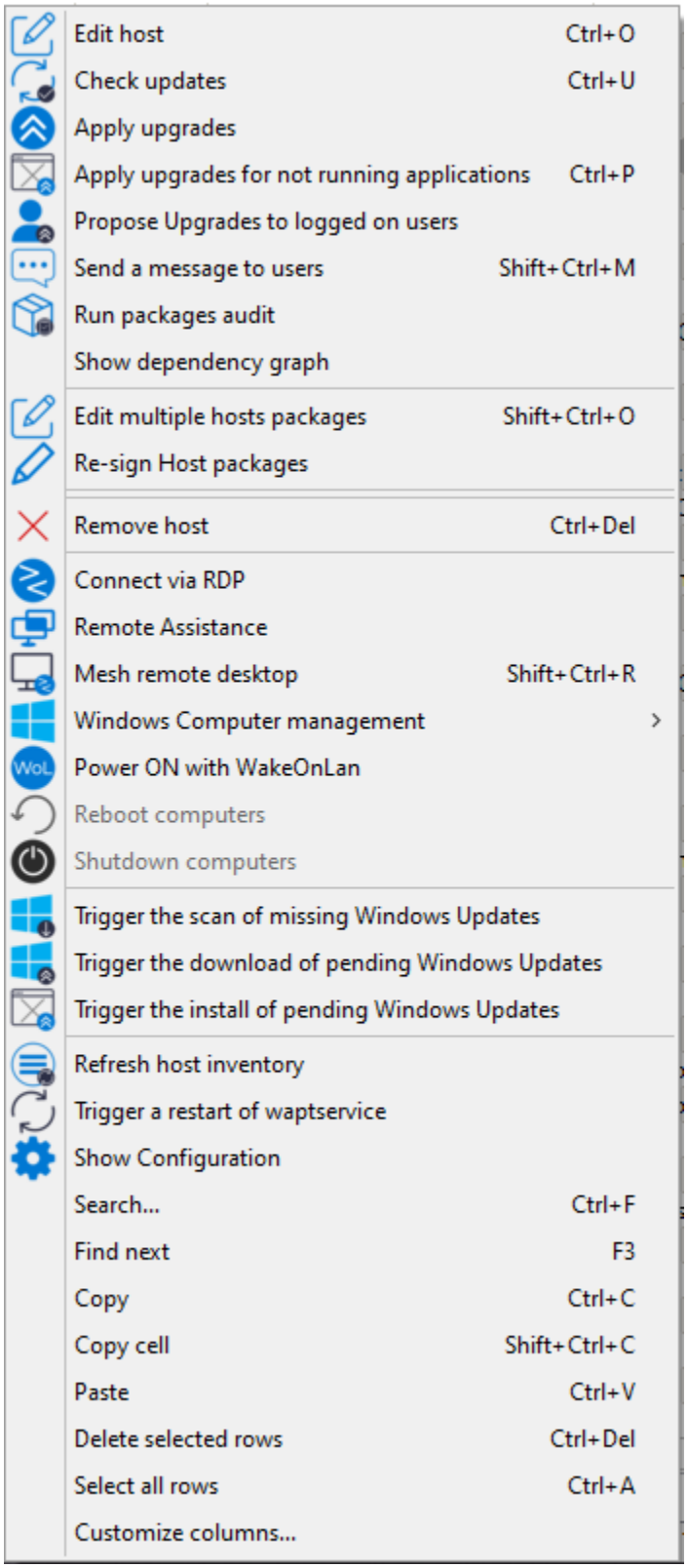


Figure4: Affichage de l'inventaire depuis La console

La console WAPT liste les hôtes qui sont enregistrés sur le serveur WAPT ainsi que des informations utiles pour gérer les hôtes.

Sélectionner un hôte affiche ses informations sur le panneau de droite de la console WAPT (*Hardware inventory* et *Software inventory*).

26.8 Comment effectuer des actions sur les hôtes ?



Certaines actions ne sont pas disponibles lorsque vous sélectionnez plusieurs hôtes.

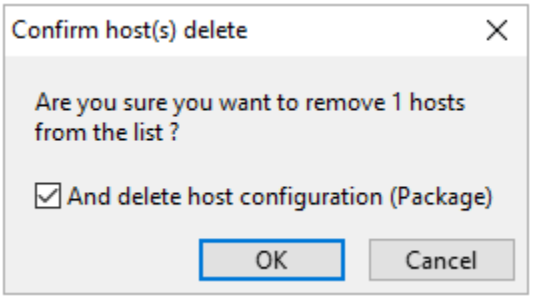
Table2: Liste des actions disponibles qui peuvent être faites sur les hôtes dans la console WAPT

Nom	Multi-sélection
Edite hôte	⊗
Vérifier les mises a jour	ok
Appliquer les mises à jour	ok
Appliquer les mises a jours d'applications qui ne sont pas lancées	ok
Proposer la mise a jours a l'utilisateur	ok
Envoyer un message aux utilisateurs	ok
Lance les audits des paquets	ok
Ajouter un paquet aux dépendances de la machine	ok
Retire un paquets des dépendances de l'hôte	ok
Re-signe les paquets *hôtes*	ok
Ajouter un paquet dans les conflits de la machine	ok
Retire un paquet des conflits de l'hôte	ok
Supprime le poste	ok
Connexion en RDP	⊗
Assistance a distance	⊗
Mesh remote desktop	ok
Gestion de l'ordinateur windows	⊗
Mettre a jour les GPO sur l'hôte	ok
Lance CleanMgr on host	⊗
Gestion de l'ordinateur	⊗
Gérer les utilisateurs et groupes	⊗
Gestion des services	⊗
Allumer avec WakeOnLan	ok
Redémarre les ordinateurs	⊗
Eteints les ordinateurs	⊗
Déclenche le scan des mises a jours Windows manquantes	ok
Déclenche le téléchargement des mises a jours Windows manquantes	ok
Déclenche l'installation des mises a jours Windows manquantes	ok
Rafraichir l'inventaire	ok
Lance un redémarrage de waptservice	ok

26.9 Supprimer l'Agent WAPT de la Console WAPT

Si vous voulez supprimer un agent WAPT depuis la console, faites un clic droit sur l'agent cible puis un clic droit et sélectionnez *Supprimer la machine* (voir ci-dessus). Vous aurez la possibilité de cocher *Supprimer la configuration*, si vous le faites, cela supprimera le *paquet machine* ciblé et supprimera toutes les informations concernant la machine sélectionnée.

Note: Supprimer une machine de la console WAPT **ne désinstalle pas** l'Agent WAPT de la machine. Veuillez vous référer à cette *documentation pour désinstaller l'Agent WAPT*. Si l'Agent WAPT n'est pas correctement supprimé, l'ordinateur s'enregistrera à nouveau auprès du Serveur WAPT.



26.10 Importe des paquets depuis un dépôt externe

Importer un paquet WAPT consiste à:

- Importer un paquet WAPT existant depuis un dépôt externe.
- Changer ses préfixes (par exemple de *tis* à *test*).
- Re-signer le paquet WAPT avec la clé privé *Administrateur* pour permettre de déployer des paquets dupliqués sur vos postes équipés de l'agent WAPT.
- Enfin, le téléverser sur le dépôt WAPT principal.

Attention: En important un paquet dans votre dépôt et en le signant, **VOUS DEVENEZ ALORS RESPONSABLE** de ce paquet et de ce qu'il fait. **Il a été signé avec votre propre clé privée**.

Tranquil IT décline toute responsabilité si vous choisissez d'utiliser des packages WAPT récupérés dans ses référentiels.

Tranquil IT déclare se dégager de toutes responsabilités si vous choisissez d'utiliser des paquets WAPT venant de ses dépôts. Sans un contrat de support, Tranquil IT ne garantie pas la pertinence du paquet pour vos propres cas d'usage, et ne garantie pas non plus la capacité du paquet à convenir à la politique de sécurité interne à votre *Organisation*.

Note: Tranquil IT utilise une ferme de construction de paquets WAPT pour maintenir son dépôt à jour, qui est surnommée LUTI (du mot français *l'outil*). Le statut de LUTI est maintenant disponible publiquement à l'adresse https://luti.tranquil.it.

LUTI surveille, dans la mesure du possible, le site Web du fournisseur du logiciel pour déclencher une mise à jour du packaging. Il vérifie l'état du fichier d'installation du logiciel sur VirusTotal, puis teste l'installation, la désinstallation et la mise à jour du paquet. Les résultats de la construction sont disponibles dans le dépôt https://wapt.tranquil.it/wapt-testing.

Après 5 jours, si l'état virustotal du packaging n'a pas changé, le nouveau packaging sera téléchargé vers le dépôt principal de WAPT. Il y a une exception à cette règle pour les navigateurs web, qui sont téléchargés de wapt-testing vers le dépôt wapt après 1/2 heure.

- Allez dans l'onglet *Private repository*.

Chaque version du paquet logiciel disponible est affichée dans le dépôt.

Si aucun paquet n'a été importé, la liste est vide. Seul le paquet « *test-waptupgrade* » sera présent si l'agent WAPT a été généré précédemment. Visitez la documentation sur *la création d'un agent WAPT*.

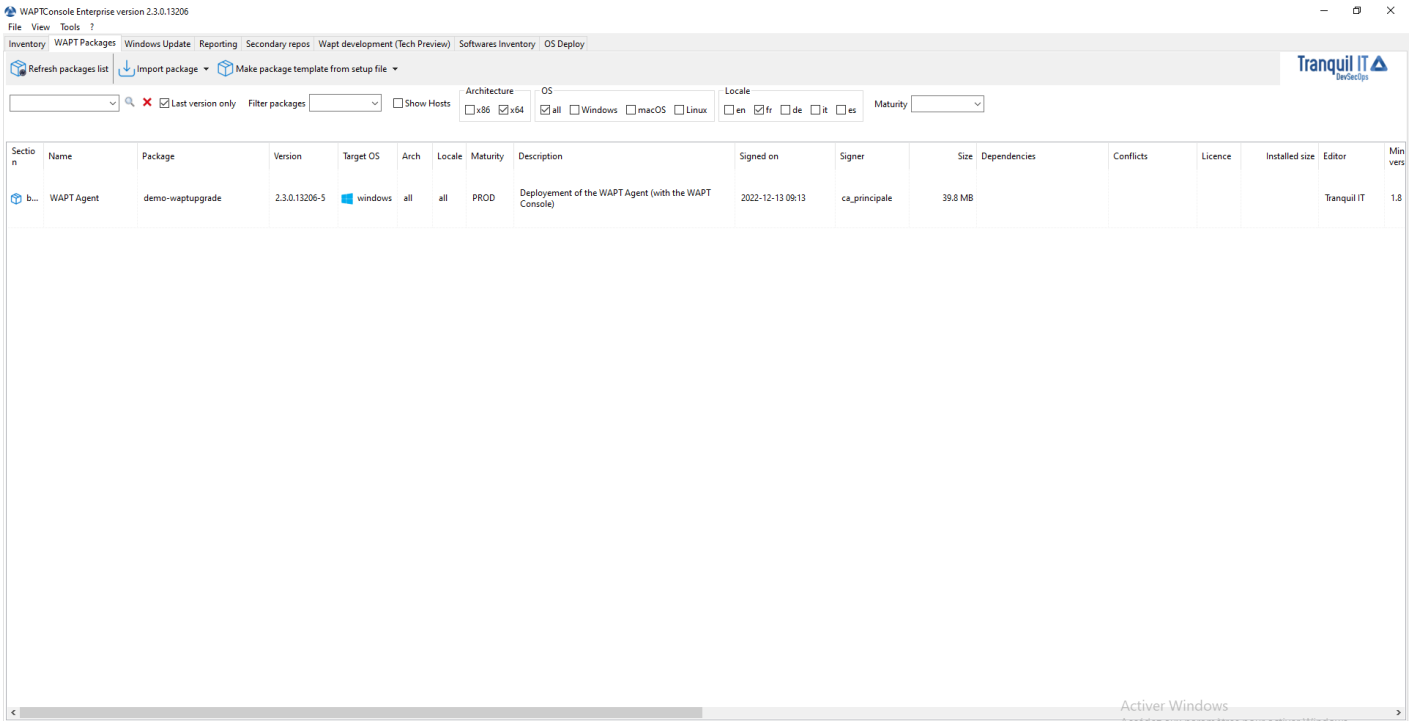


Figure5: Les logiciels disponibles sont affichés dans la console WAPT

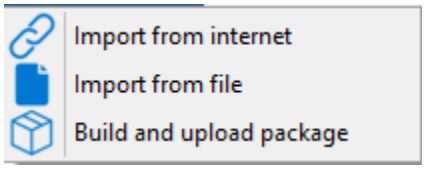
26.10.1 Importer un paquet depuis un dépôt externe sur Internet

Cette première méthode vous permet de télécharger des paquets directement depuis un dépôt externe WAPT dans votre *Organisation*.

Par défaut le dépôt Tranquil IT est configuré, pour ajouter un autre dépôt, voir la documentation sur les paramètres de configuration des dépôts externes.

Par défaut, les certificats TLS et SSL des dépôts externes sont vérifiés.

- Cliquez sur *Importer un paquet* et *Importer depuis Internet*.



Note: La grille d'affichage montre la liste de paquets disponibles sur le dépôt distant. Il est possible de choisir la plateforme, l'OS et la langue.

- Il y a 2 méthodes pour importer un paquet WAPT:
  - pour importer un paquet, sélectionner un paquet puis *Clic-droit* → *Importer*;

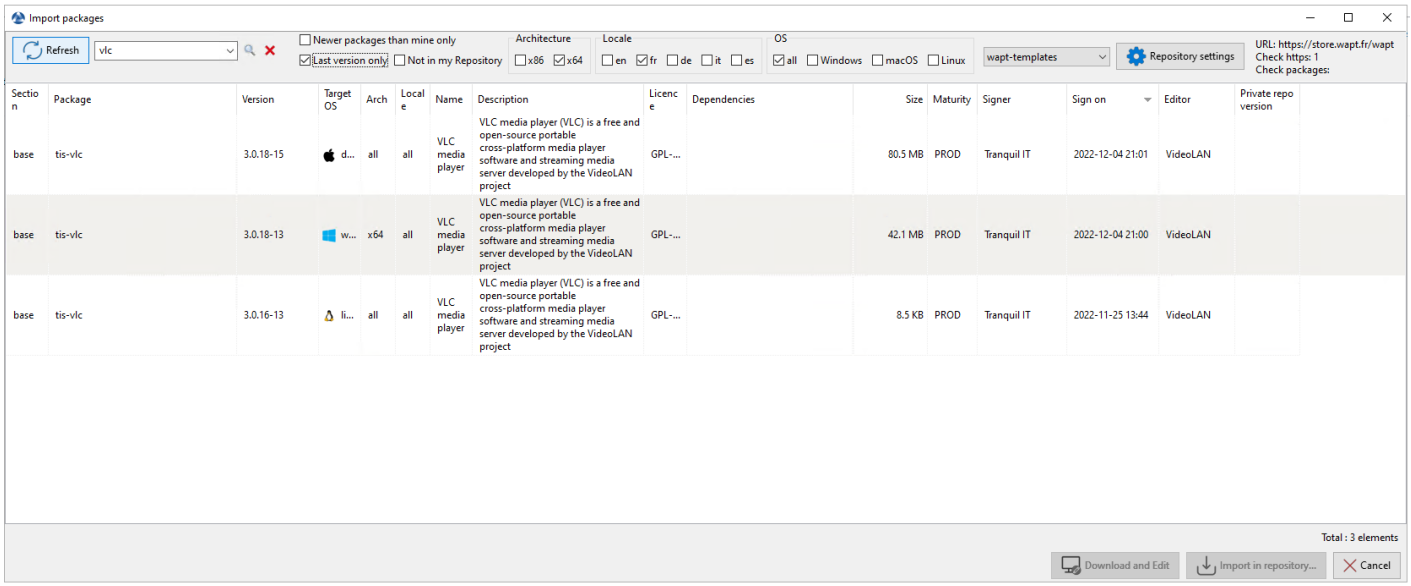
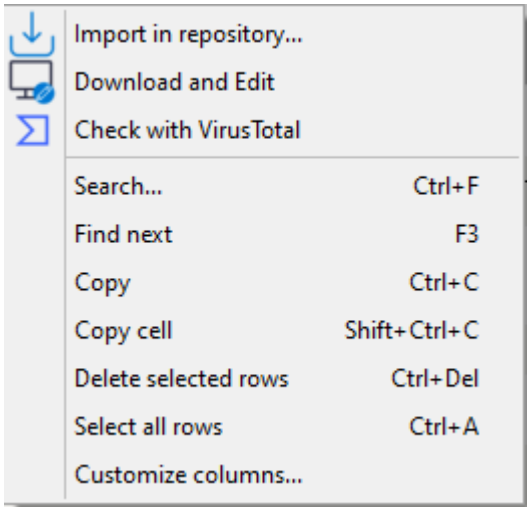


Figure6: Le paquet WAPT importé s'affichera dans votre dépôt WAPT local



– Ou en bas a droit de la console *Importer dans le dépôt*.

Indication: Vous pouvez vérifier le paquet avec Virus Total afin de savoir si le paquet n'est pas signalé dans la liste des antivirus.

- Valider l'importation dans le dépôt local. Il est possible de *modifier la maturité* d'un paquet WAPT avant d'importer le paquet dans le dépôt privé.

Indication: Vous pouvez mettre à jour tous vos paquets depuis votre dépôt en cochant *Paquets plus récents que les miens uniquement* et *Dernière version uniquement*, puis utilisez `Ctrl+A` pour sélectionner tous les paquets plus récents qui sont disponibles sur le dépôt Tranquil IT et que vous avez sur votre dépôt dans une version plus ancienne, puis cliquez sur *Importer dans le dépôt*.

Vous pouvez rechercher les paquets WAPT qui ne sont pas dans votre dépôt local si vous cochez *Pas présent dans mon dépôt*.

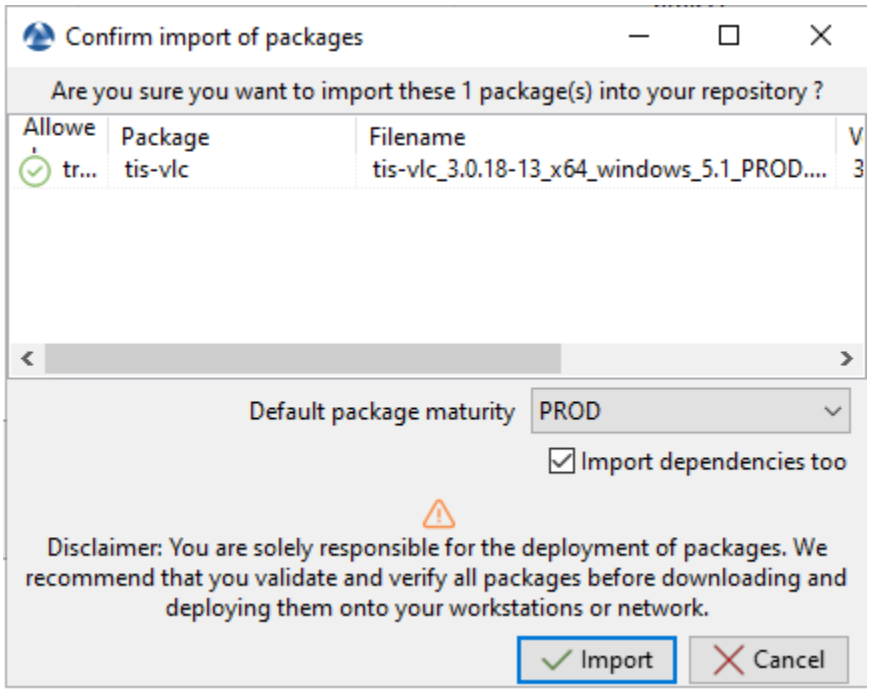


Figure7: Boîte de dialogue pour préparer et confirmer l'importation d'un packaging WAPT dans un référentiel WAPT

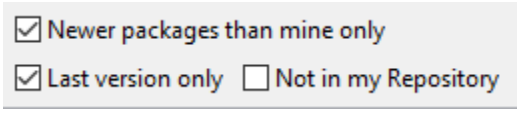


Figure8: Case à cocher pour sélectionner la version la plus récente et la dernière version d'un paquet WAPT sur Internet

---

- Le téléchargement du paquet commence.

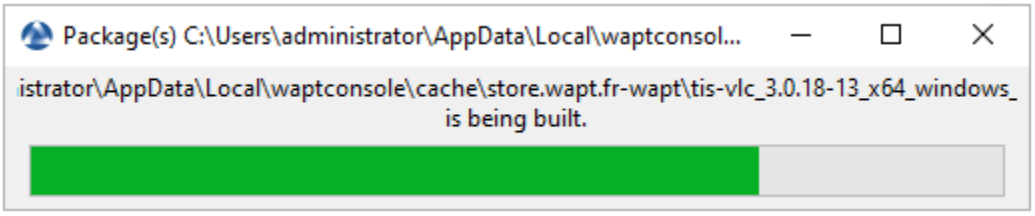


Figure9: Progression du processus d'importation des paquets

- Puis, entrer le mot de passe de votre clé privé.

La console WAPT confirme que le paquet a bien été dupliqué sur votre dépôt WAPT local.

La paquet apparaît alors sur votre dépôt local WAPT avec le préfixe de votre Organisation.

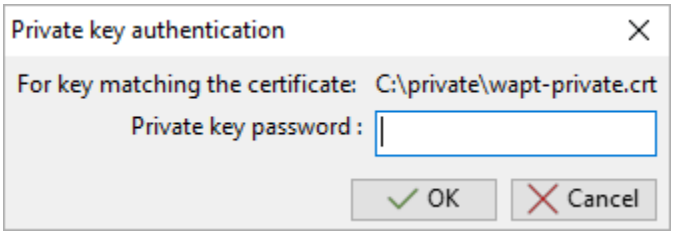


Figure10: Entrez le mot de passe pour déchiffrer la clé privée

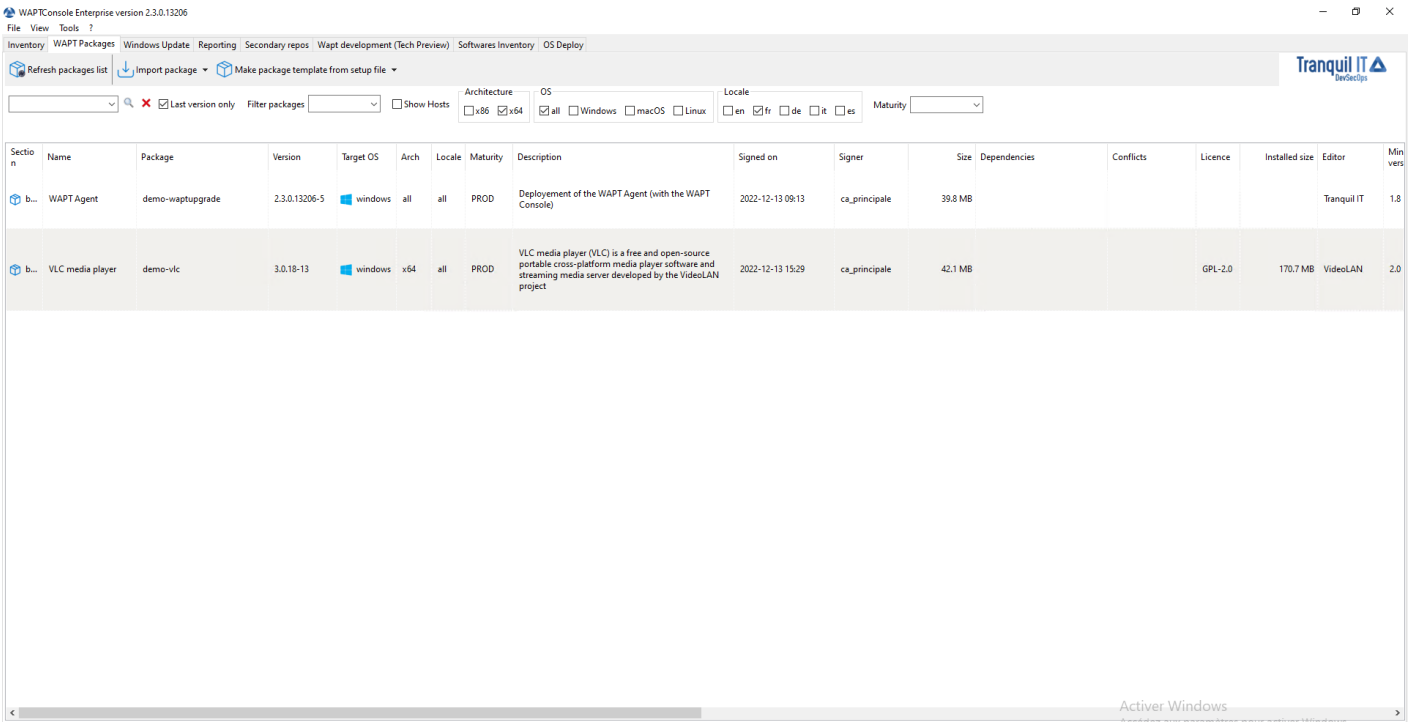
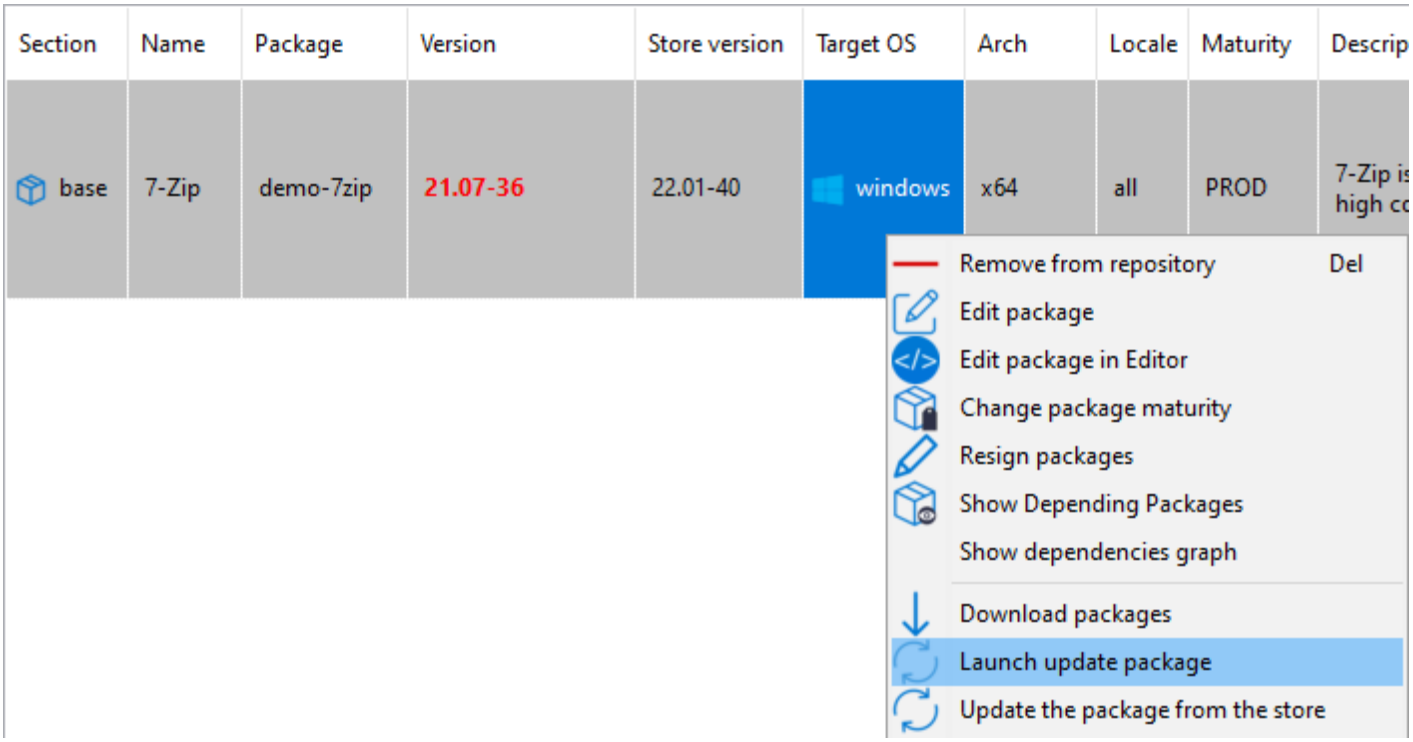


Figure11: La console WAPT affiche le paquet importé

Télécharger une version plus récente d'un paquet depuis votre dépôt

Si vous voyez dans votre dépôt une version de paquet en rouge et en **gras**, cela signifie qu'une version plus récente existe sur le dépôt public (par défaut le dépôt est celui de Tranquil IT). Dans l'onglet *Paquets WAPT*, *Version* est la version du paquet dans le dépôt local ; *Version du dépôt* est la version du magasin public et *Version du logiciel* est la version de Luti lorsque Luti connaît la version du logiciel de l'éditeur.



Vous avez 2 options pour mettre à jour le paquet WAPT dans le dépôt local :

- Lancer un **`update_package`** en faisant un clic droit sur le paquet à mettre à jour puis *Lancer la mise à jour*. Elle exécutera l'**`update_package`** défini dans le paquet. S'il n'y a pas de code pour la mise à jour du logiciel, la commande ne fera rien. Cette méthode a un inconvénient : si le `setup.py` a été amélioré dans le dépôt Tranquil IT, vous ne bénéficierez pas de cette amélioration. Pour en savoir plus, veuillez consulter *la documentation sur la mise à jour d'un paquet WAPT*.
- Cliquer sur *Mettre à jour le paquet à partir du dépôt*. Cette méthode va aller chercher la dernière version du paquet dans les dépôts publics (par défaut Tranquil IT). Le bénéfice de cette méthode est que vous bénéficierez d'une amélioration du code dans le `setup.py`. C'est donc la méthode recommandée. Vous pouvez consulter cette documentation sur *comment mettre à jour un paquet WAPT depuis un dépôt public*.

Changer la maturité d'un paquet WAPT avant de l'importer dans le dépôt

Il est possible de changer la maturité d'un paquet WAPT avant de le charger dans votre dépôt privé en choisissant **DEV**, **PREPROD** ou **PROD** dans *Maturité des paquets par défaut*.

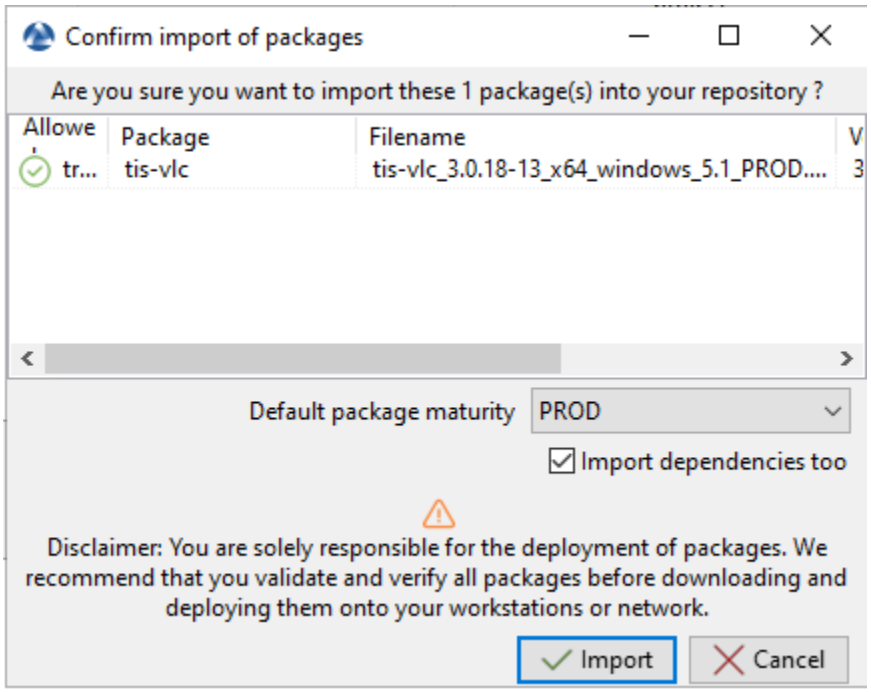
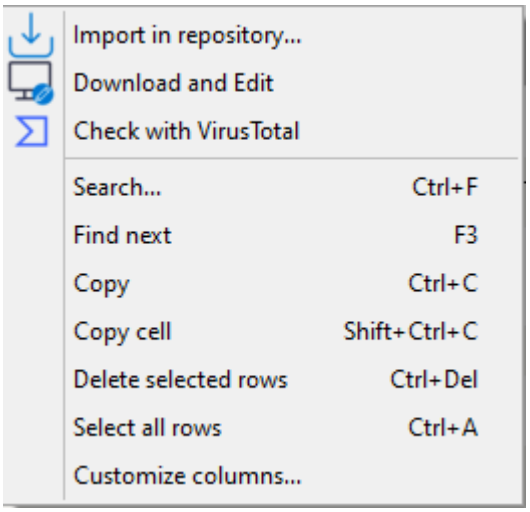


Figure12: Boîte de dialogue pour préparer et confirmer l'importation d'un packaging WAPT dans un référentiel WAPT

Editez un paquet avant de l'importer

Il est possible d'éditer un paquet téléchargé depuis un dépôt externe avant de l'importer dans dépôt WAPT principal.

- Plusieurs choix sont possibles:
  - pour importer un paquet, sélectionner un paquet puis *Clic-droit* → *Importer*;



  - ou en cliquant sur *Télécharger et éditer* en bas à droite de la fenêtre;

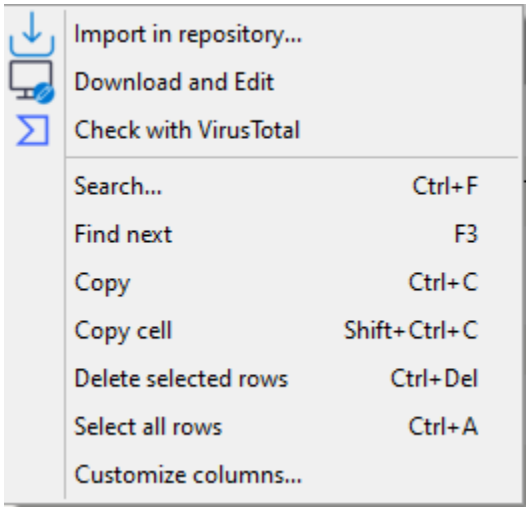
PyScripter,s'il est installé, va ouvrir les fichiers `control` et `setup.py` du paquet WAPT.

Pour plus d'information, visitez la documentation sur *créer un paquet WAPT*.

26.11 Dupliquer des paquets depuis un dépôt externe

Vous pouvez importer un fichier `.wapt` depuis n'importe quel stockage.

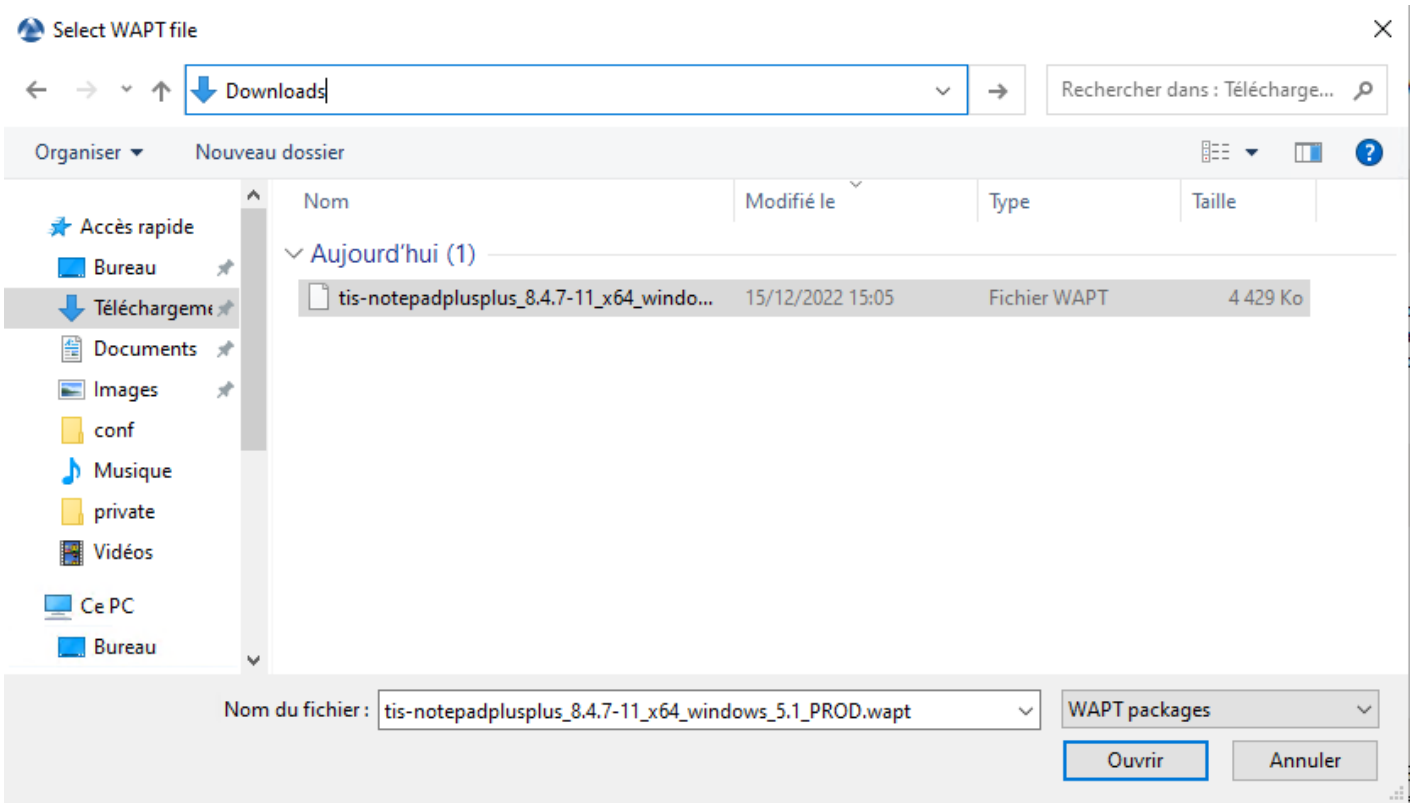
- Cliquer sur *Importer un pacquet* puis sur *Importer depuis un fichier*.



- Sélectionner le fichier à importer.
- Cliquez sur *Ouvrir* pour importer le fichier.

La console WAPT confirme que le paquet a bien été dupliqué sur votre dépôt WAPT local.

La paquet apparaît alors sur votre dépôt local WAPT avec le préfixe de votre Organisation.



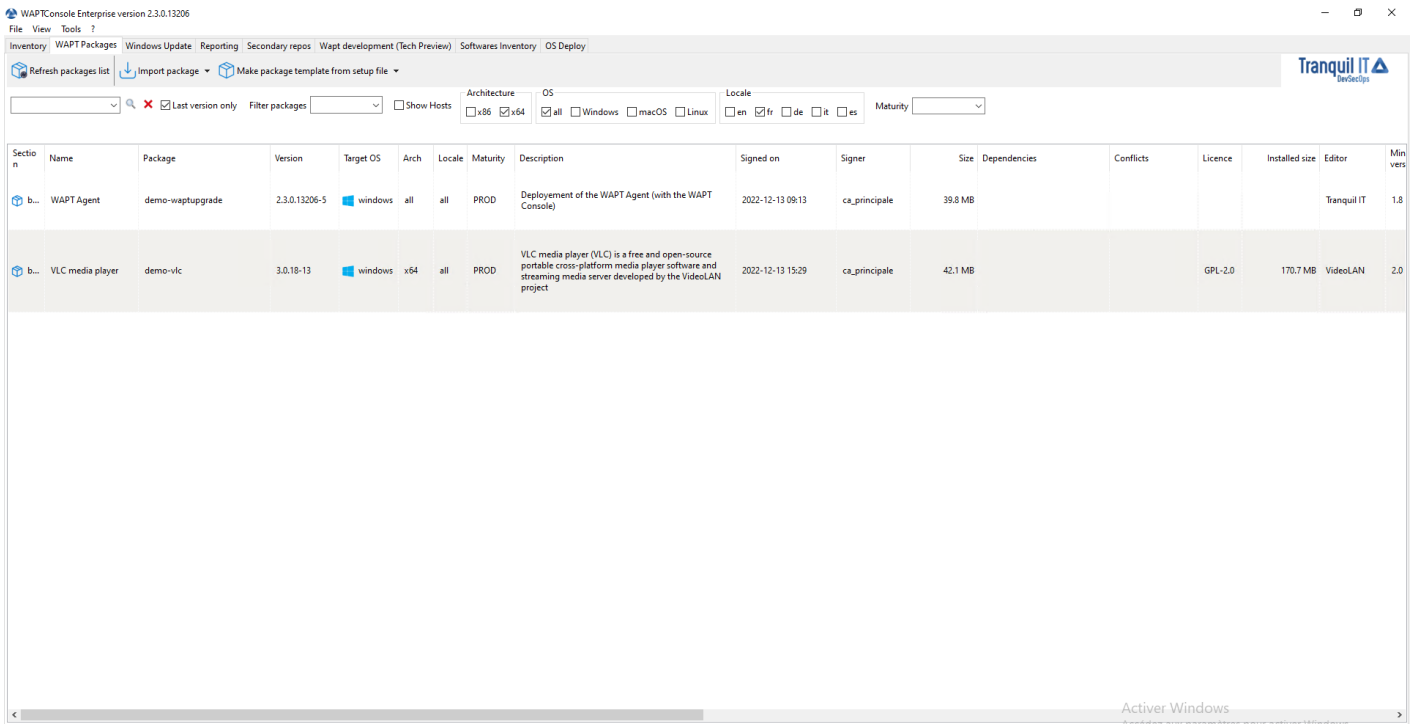


Figure13: La console WAPT affiche le paquet importé

Note: Il n'est pas possible de changer la maturité avant d'importer dans ce cas.

Lors de l'import de votre nouveau paquet WAPT dans votre dépôt privé, le changement du préfixe et la re-signature du paquet WAPT sont transparents et automatique.

26.12 Gérer les paquets sur le dépôt

Dans l'onglet *WAPT Packages*, la liste des paquets actuellement disponibles sur le dépôt WAPT apparaît. Par défaut, la console va uniquement montrer la dernière version des paquets.

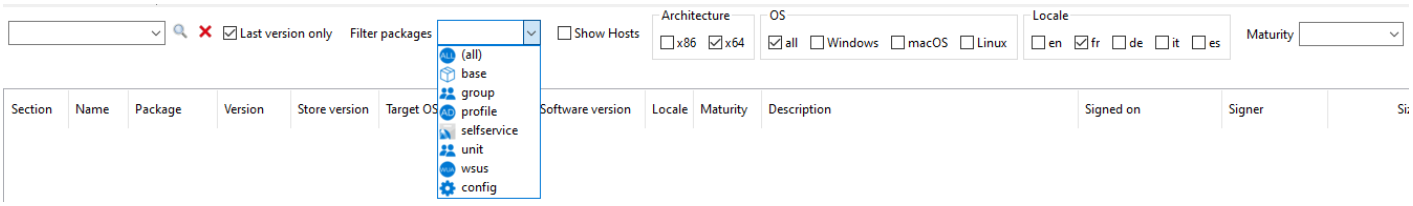


Table3: Liste des éléments de la fenêtre

Label	Description
La barre de recherche *Nom du type ou description*	Permet de rechercher par nom de packaging WAPT ou par description.
Package'Increment the package version"	Permet d'afficher toutes les versions des packages WAPT dans la console WAPT.
Prefix du nouveau paquet	Permet de filtrer les packages WAPT par type (*all*, *base*, *group*, *profile*, *selfservice*, *unit*, *waptwua*).
Machine	Affiche les hôtes sur lesquels le packaging WAPT sélectionné est installé.
La case à cocher *Architecture x86*	Permet de filtrer sur les hôtes ayant une architecture de processeur basée sur x86.
La case à cocher *Architecture x64*	Permet de filtrer sur des hôtes ayant une architecture de processeur basée sur x64.
La case *OS all* à cocher	Permet de filtrer les hôtes en fonction de tout OS.
La case *OS Windows* à cocher	Permet de filtrer les hôtes en fonction du :abbr:`OS (Operating System)`de Windows.
La case *OS macOS* à cocher	Permet de filtrer les hôtes en fonction du macOS OS.
La case *OS Linux* à cocher	Permet de filtrer les hôtes en fonction du :abbr:`OS (Operating System)`de Linux.
La case à cocher *Locale en*	Permet de filtrer les hôtes localisés en anglais.
La case *Locale fr* à cocher	Permet de filtrer les hôtes localisés en français.
La case à cocher *Locale de*	Permet de filtrer les hôtes localisés en allemand.
La boîte à cocher *Locale it*	Permet de filtrer les hôtes localisés en italien.
La case à cocher *Locale es*	Permet de filtrer les hôtes localisés en espagnol.
La liste déroulante *Maturité*	Permet de filtrer sur le niveau de maturité configuré sur les hôtes.

26.12.1 Faire une recherche basé sur un paquet WAPT

Dans l'onglet le dépôt, sélectionnez le paquet puis cliquez sur *Afficher les clients*.

La grille va afficher les hôtes sur lesquels le paquet est installé. Notez que le filtre n'est actif que sur l'attribut *Package* du paquet sélectionné.

Les différentes colonnes affichent des informations à propos des paquets installés sur la machine (e.g *package version*, *package status*, *audit status*, *installation date*, *architecture*).

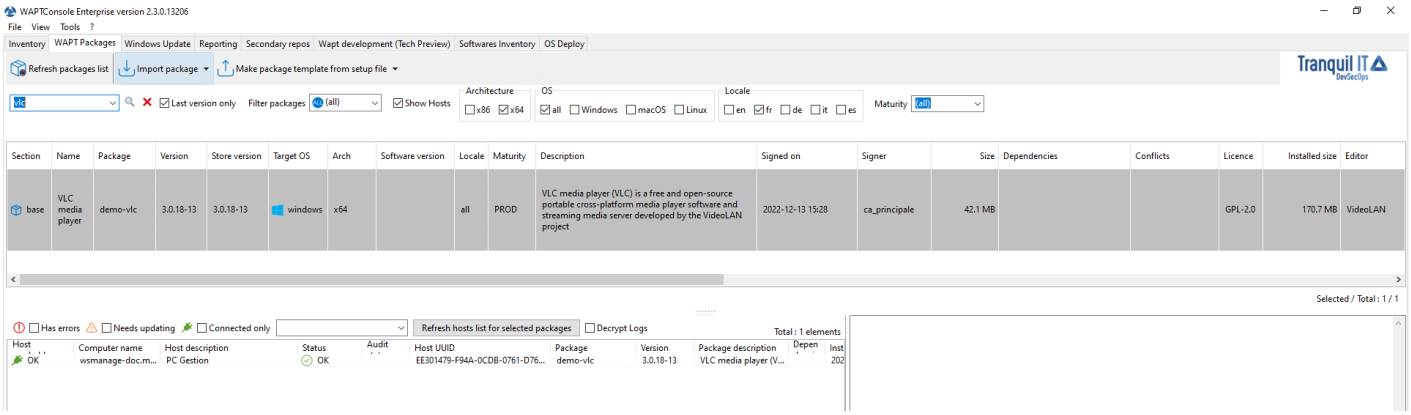


Figure14: Faire une recherche basé sur un paquet WAPT

Vous pouvez aussi ajouter les colonnes *Log install* et *Last Audit Output* pour jeter un coup d'oeil aux journaux d'installation et aux journaux d'audit.

26.12.2 Afficher les dépendances du paquet

Dans l'onglet *Dépôt*, sélectionner le paquet WAPT et cliquer ensuite sur *Montrer les paquets en dépendance*. Cela montrera si le paquet WAPT a une dépendance.

Dans cet exemple, le paquet *unit Computers* a **`demo-vlc`** dans ses dépendances.

26.12.3 Afficher les dépendances du paquet

Dans le dépôt, sélectionnez le paquet WAPT et cliquez ensuite sur *Montrer le graphe de dépendance* pour voir toutes les dépendances et sous-dépendances.

Dans cet exemple, on peut voir toutes les dépendances et sous-dépendances du paquet `demo-waptdev`.

Indication: Vous pouvez faire la même chose avec une machine dans l'onglet *Inventaire*.

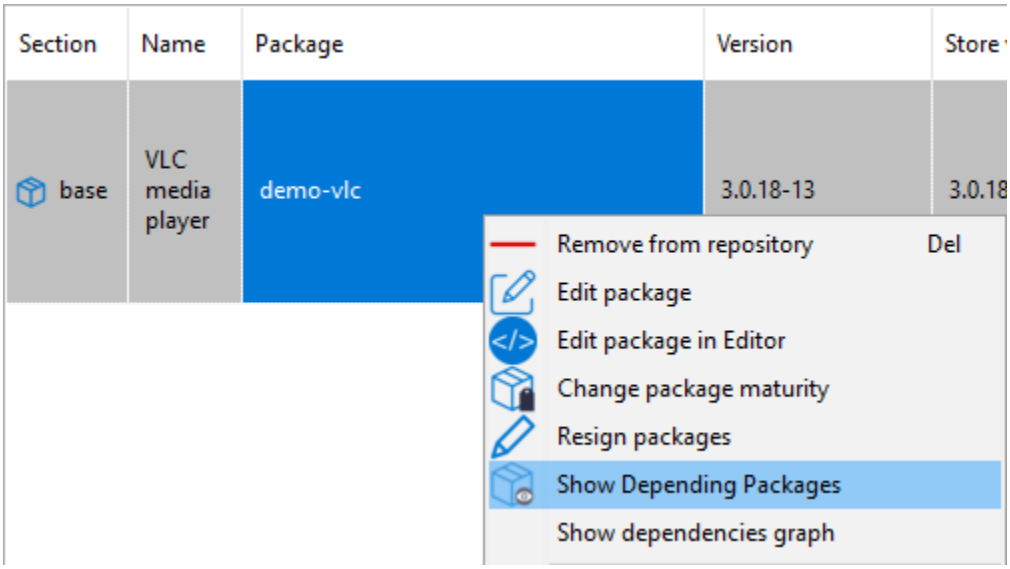


Figure15: Cliquer avec le bouton droit de la souris sur le paquet sélectionné pour voir si d'autres logiciels en dépendent

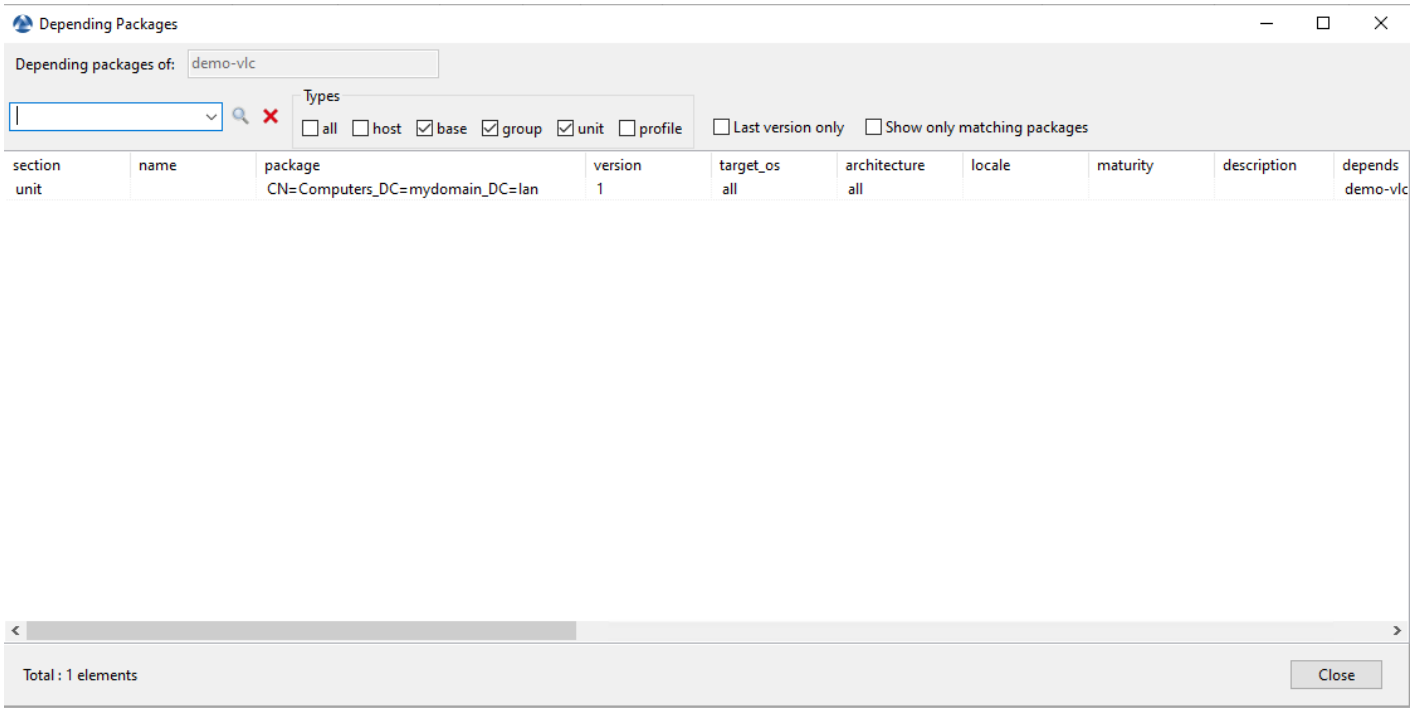


Figure16: Exemple d'un paquet dont le paquet sélectionné a une dépendance

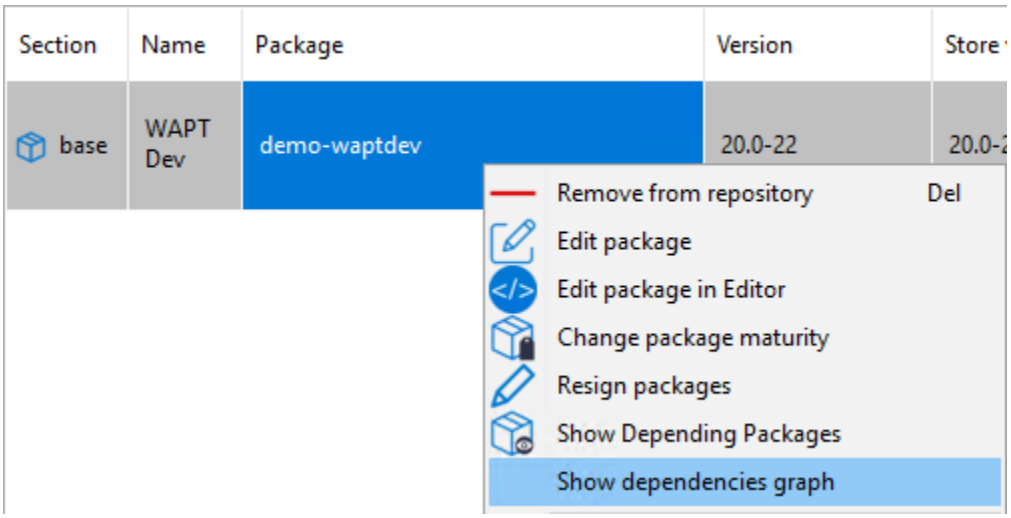


Figure17: Cliquer avec le bouton droit de la souris sur un paquet pour voir son graphe de dépendances

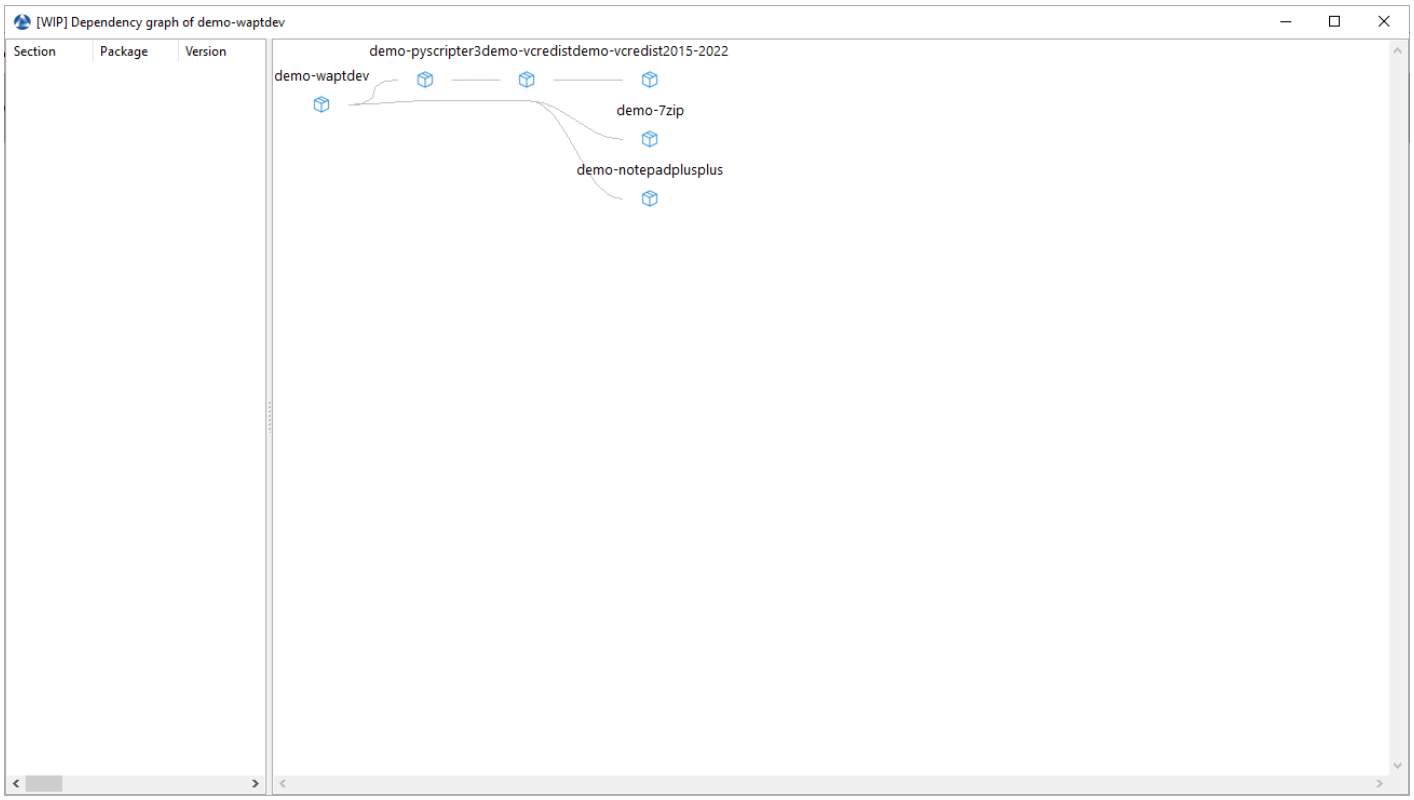
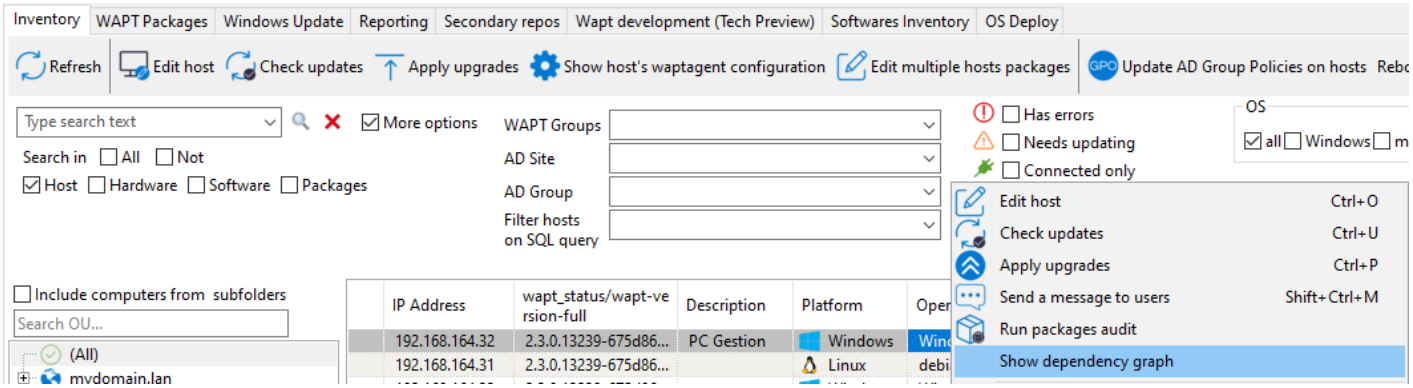


Figure18: Exemple d'un paquet avec son graphe de dépendances



Dans cet exemple, ont peut voir toutes les dépendances et sous-dépendances de la machine sélectionnée.

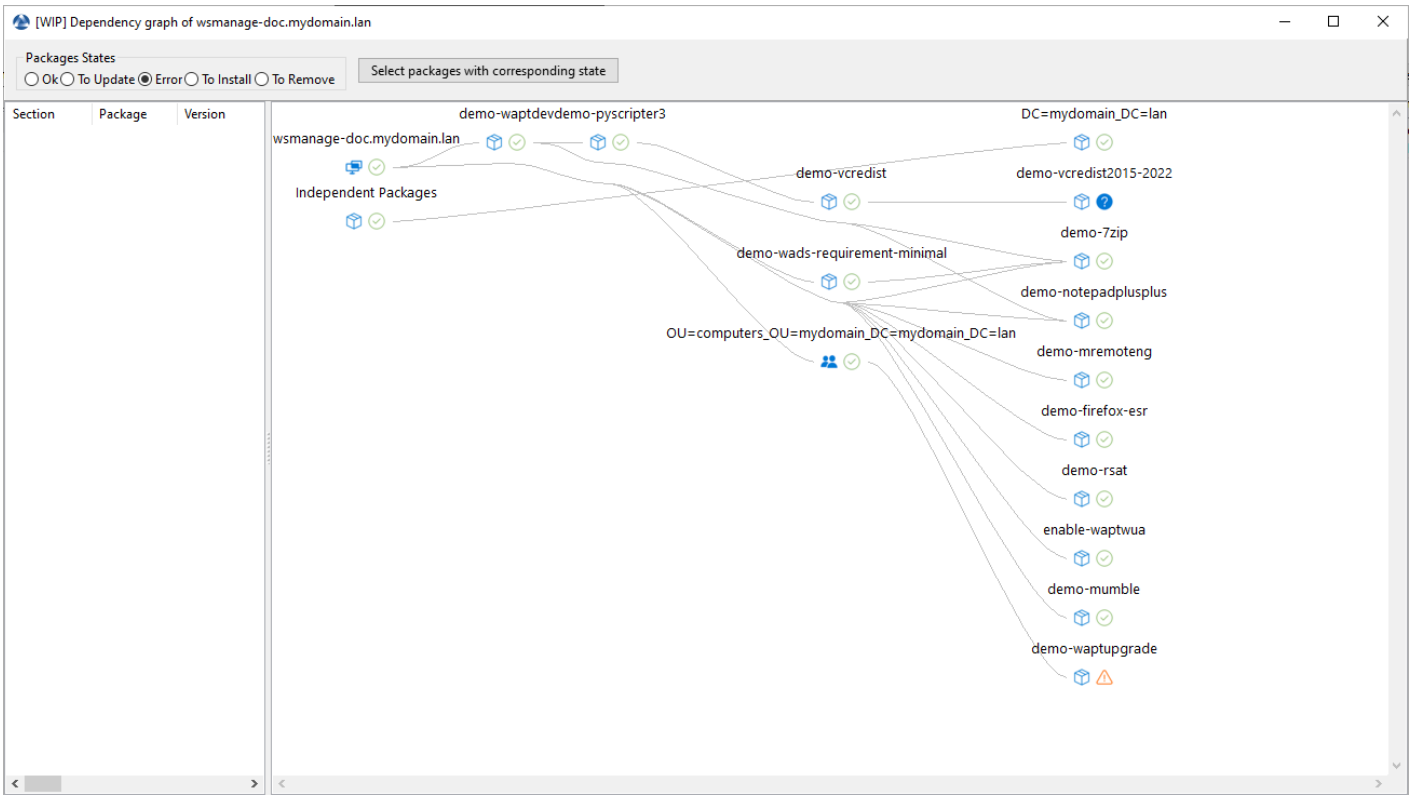


Figure19: Exemple d'une machine avec son graphe de dépendances

26.12.4 Changer la maturité d'un paquet WAPT après l'import sur le dépôt

Lorsqu'un paquet est importé sur le dépôt WAPT il est possible de changer la maturité en faisant un clic-droit sur le paquet WAPT. Sélectionnez votre maturité sur le menu *Change packages maturity*.

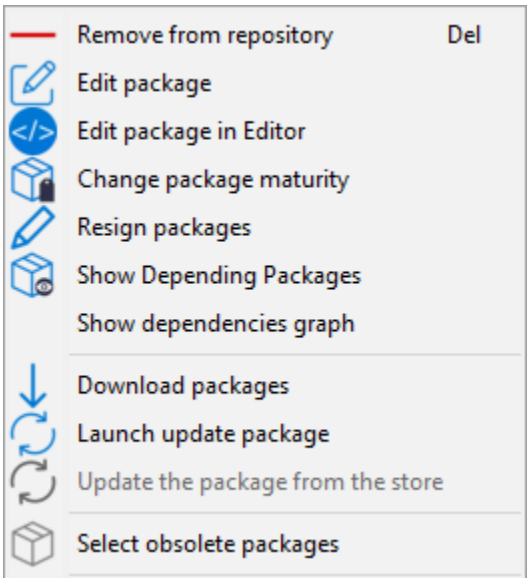


Table4: Changer la maturité d'un paquet WAPT

Label	Description
Package'Increment the package version"	Incrémenter la version du paquet (numéro de verison après -).
Delete old packages after successful process	Supprimer les vieux paquets WAPT après avoir changé la maturité.
Change la maturité du paquet	Configurer la nouvelle maturité du paquet WAPT.
Prefix du nouveau paquet	Configurer un nouveau préfixe pour le paquet WAPT. Le préfixe est sensible à la casse, nous recommandons d'utiliser des minuscules.

Vous pouvez arrêter le processus en cliquant le bouton *Arrêter le processus*.

Vous pouvez arrêter le processus en cliquant le bouton *Arrêter le processus*.

Une fois fini, le statut passe en ✅.

Avertissement: Vous pouvez changer la maturité de multiples paquets en une fois

Le changement de maturité du paquet va modifier le hash du paquet.

Si le paquet est utilisé dans une GPO, comme **`waptupgrade`**, vous devrez changer le hash de votre GPO.

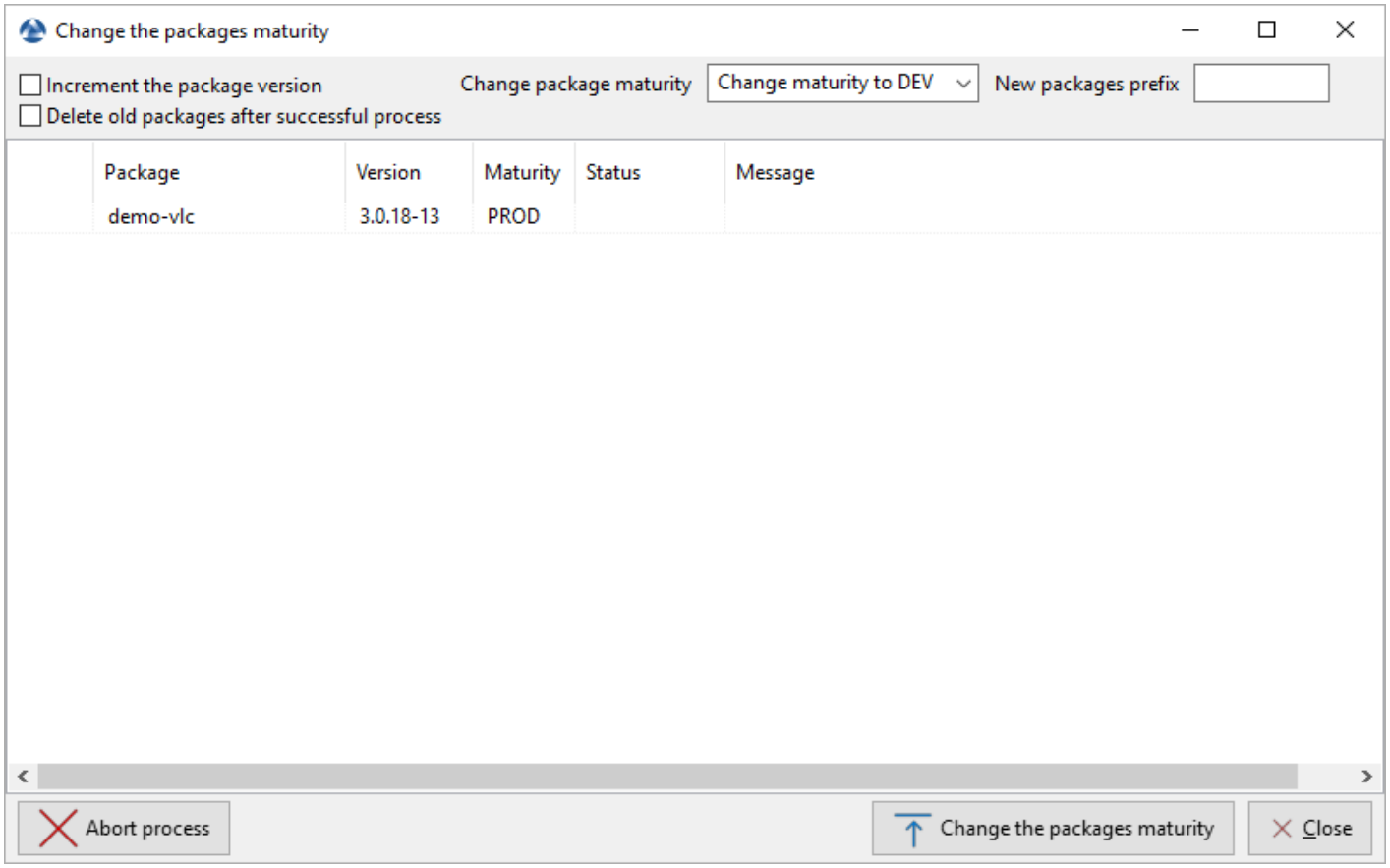
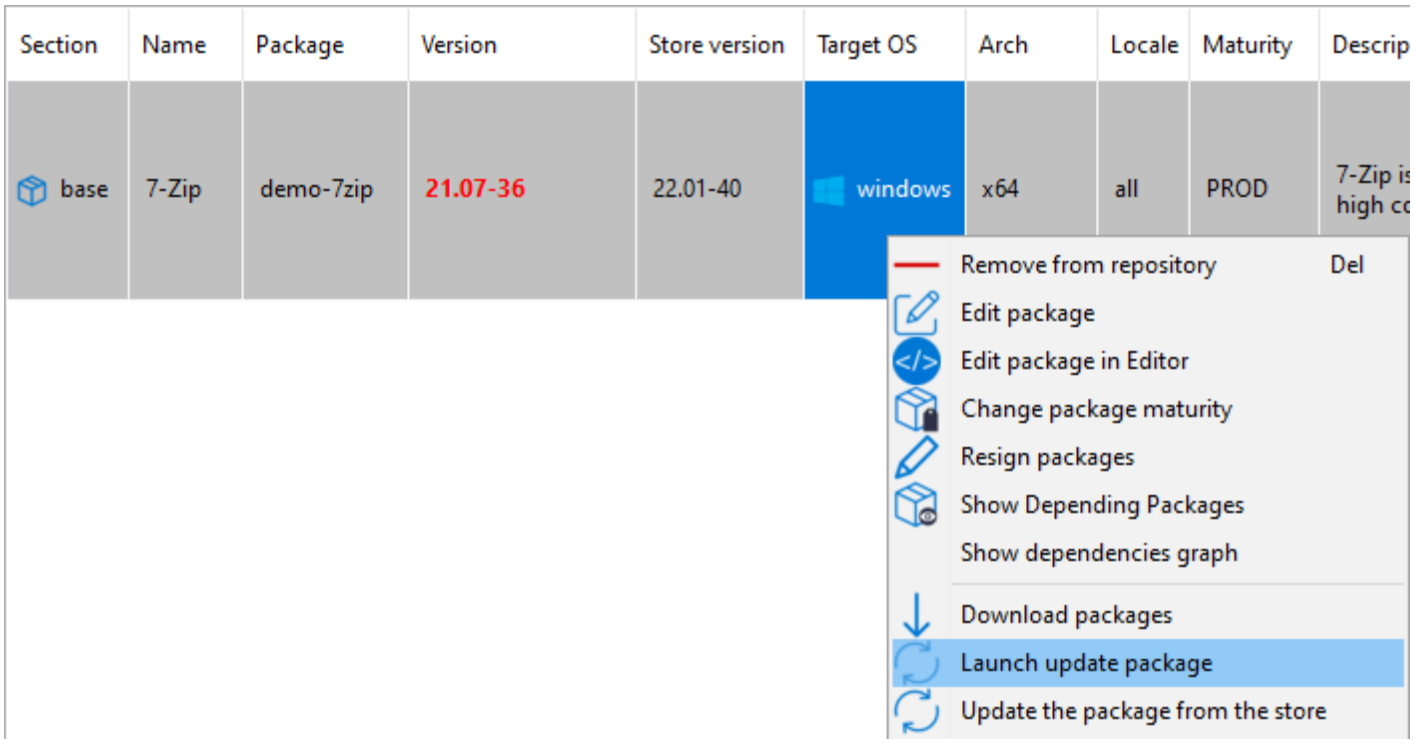


Figure20: Changer la maturité d'un paquet WAPT

26.12.5 Mettre à jour le paquet à partir du dépôt de Tranquil IT

Lorsque vous voulez mettre à niveau un ou plusieurs paquets WAPT avec l'option *Lancer la mise à jour des paquets*, voici comment cela fonctionne. Tout d'abord, sélectionner un ou plusieurs paquets que vous souhaitez mettre à niveau.



Ensuite, une fenêtre avec les paquets sélectionnés s'ouvrira, vous pouvez cliquer sur *Lancer la mise jour des paquets sélectionnés*. Après cela, si vous n'avez pas coché *Transférer directement*, vous devrez cliquer sur *Transférer les paquets sélectionnés* afin de signer, construire et télécharger le paquet sur le Serveur WAPT.

26.12.6 Mettre à jour le paquet à partir du dépôt de Tranquil IT

Lorsque vous voulez mettre à jour un ou plusieurs paquets WAPT avec l'option *Mettre à jour le paquet depuis le dépôt public*, voici comment cela fonctionne. Tout d'abord, sélectionner un ou plusieurs paquets que vous souhaitez mettre à niveau.

Ensuite, une fenêtre avec les paquets sélectionnés s'ouvre, vous pouvez cliquer sur *Mettre à jour le paquet à partir du dépôt public*. Après cela, WAPT ira chercher la nouvelle version dans le dépôt de Tranquil IT avec les améliorations du code. Vous verrez la même boîte de dialogue lorsque vous cliquez sur *Importer depuis Internet*.

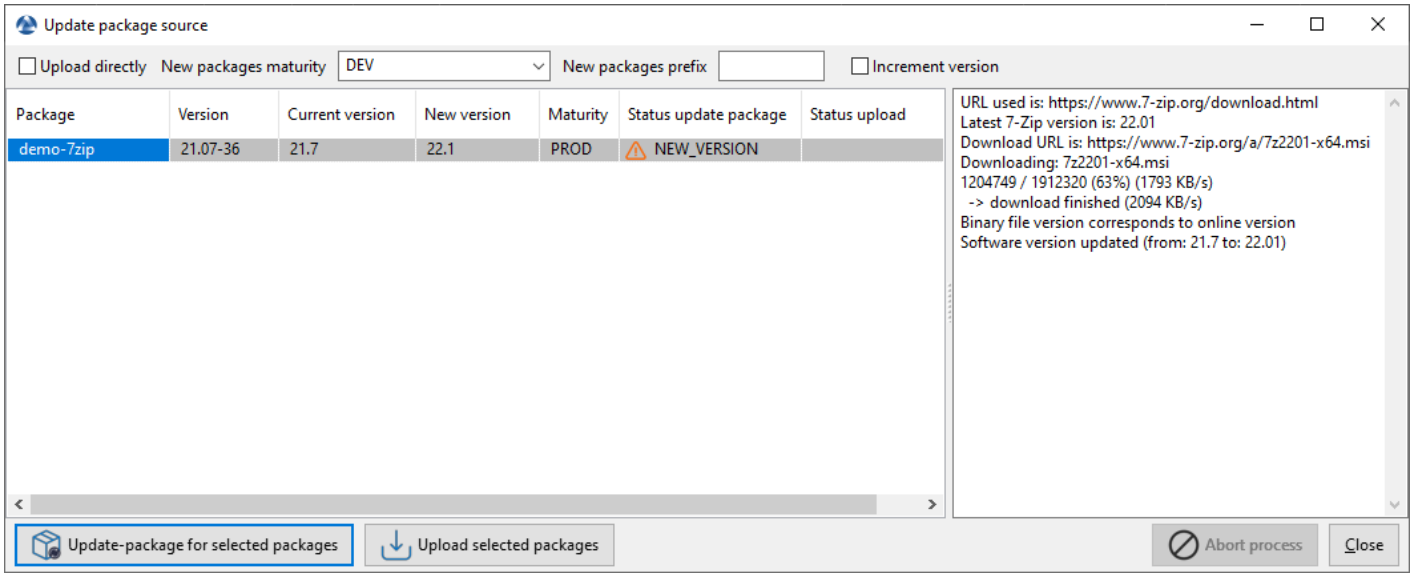


Figure21: Fenêtres de la Console WAPT affichant des versions plus récentes de paquets WAPT

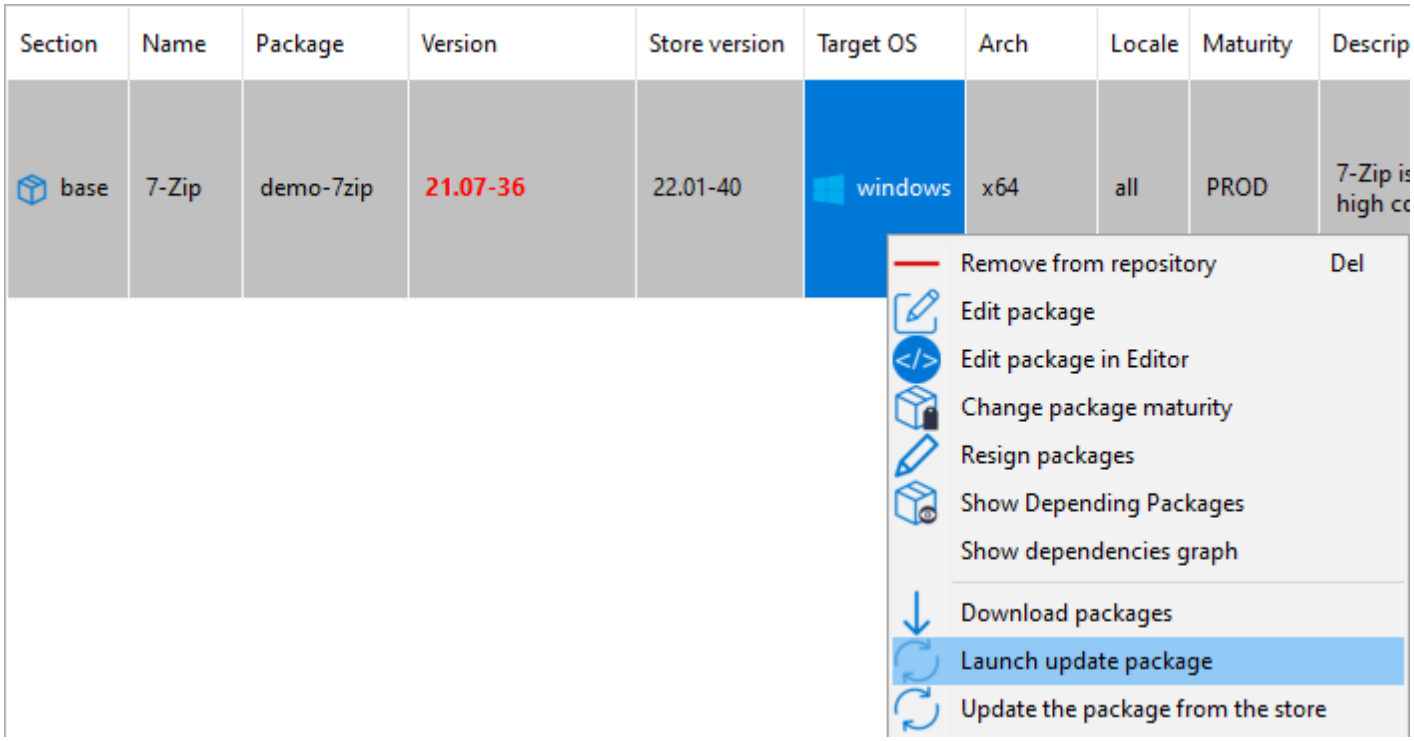


Figure22: Console WAPT affichant une version plus récente du paquet

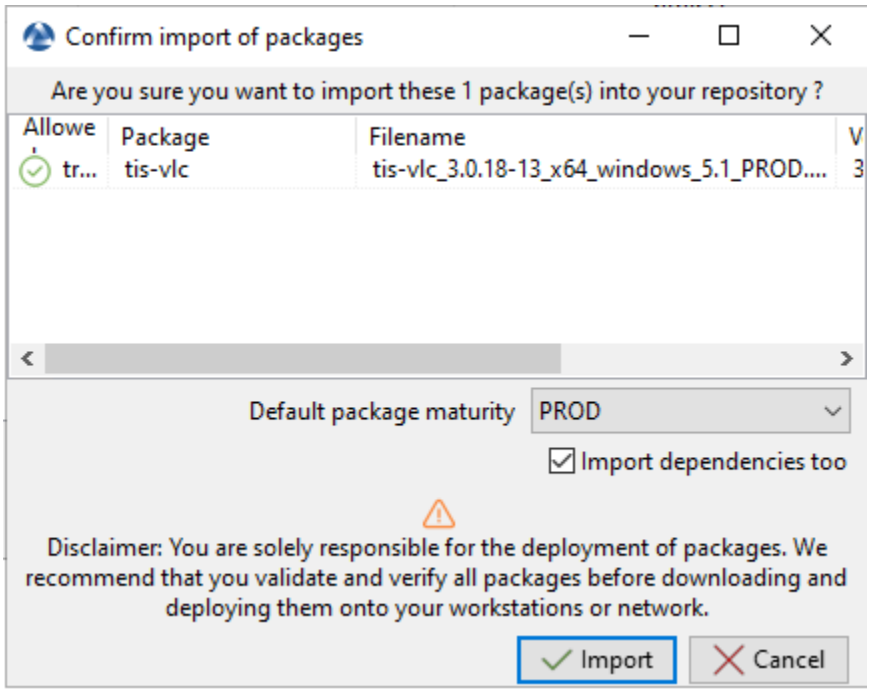
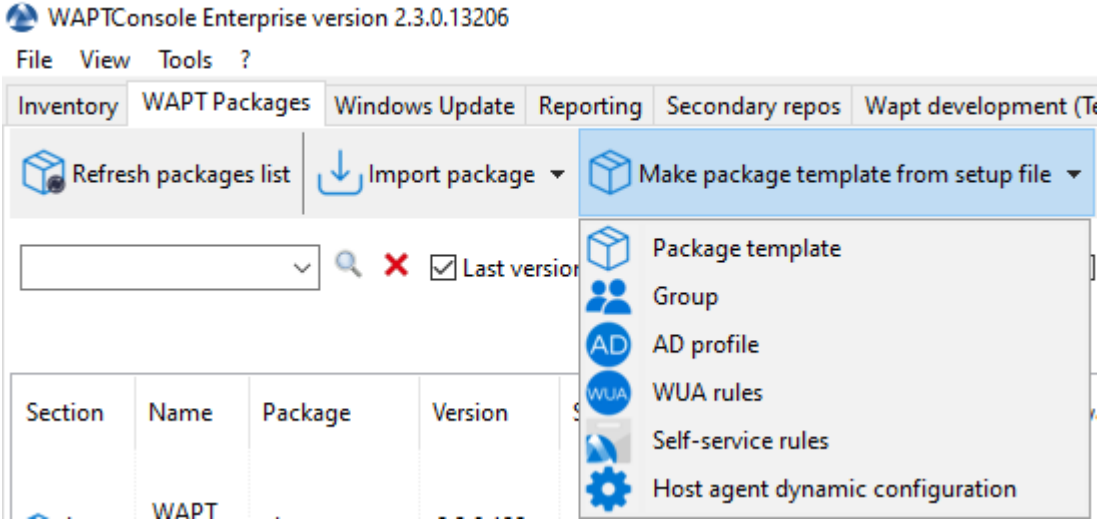


Figure23: Fenêtres de la Console WAPT affichant des versions plus récentes de paquets WAPT

26.12.7 Créer un groupe de paquet

Les groupes de paquet vous permettent de créer un paquet contenant d'autres paquets qui seront affectés en tant que dépendance à un hôte.

Pour créer un paquet de packages WAPT, allez sur l'onglet *WAPT Packages* dans la console WAPT, puis cliquez sur le bouton *Make package template from setup file* et enfin choisissez l'élément de menu *Group*.



- Change le nom dans *nom du paquet*.
- Remplissez la description.

- Ajoutez des paquets WAPT au groupe en les glissant et déposant ou bien en faisant un clic-droit sur le nom du paquet, et ajoutez-le au pack de paquets.

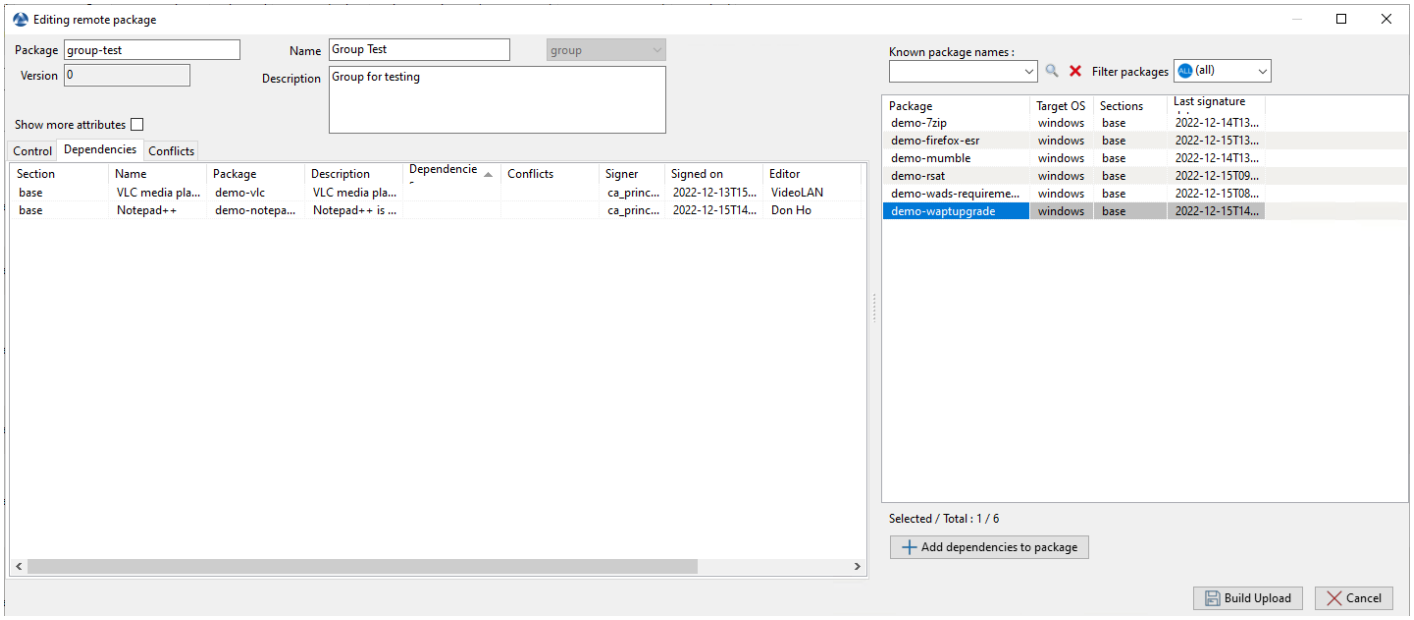


Figure24: Créer un groupe de paquet

- Cliquer sur *Enregistrer* pour enregistrer le paquet groupe.

Indication: Pour désinstaller un paquet, il est possible d'ajouter le paquet comme paquet interdit à un paquet *group*. Le paquet WAPT interdit, s'il est installé, sera supprimé avant que les autres paquets WAPT ne soient installés.

26.12.8 Supprimer un paquet WAPT

Pour supprimer un paquet depuis le dépôt, *Clic-droit → Supprimer du dépôt*.

Vous pouvez sélectionner plusieurs paquets a la fois pour les supprimer.

Supprimer un paquet de cette manière ne supprimera **uniquement** que ce paquet et non les versions antérieures. Pour ce faire, vous devrez décocher *Dernière version seulement* et sélectionner toutes les versions du paquet à supprimer. Vous pouvez utiliser l'option *Sélectionner les paquets obsolètes* pour vous aider puis *Supprimer du dépôt* pour aller plus vite.

Si vous supprimez un paquet utilisé par au moins une machine, vous aurez une fenêtre d'alerte. Si vous cochez *Appliquer en cascade*, cela supprimera les dépendances sur toutes les autres machines concernées.

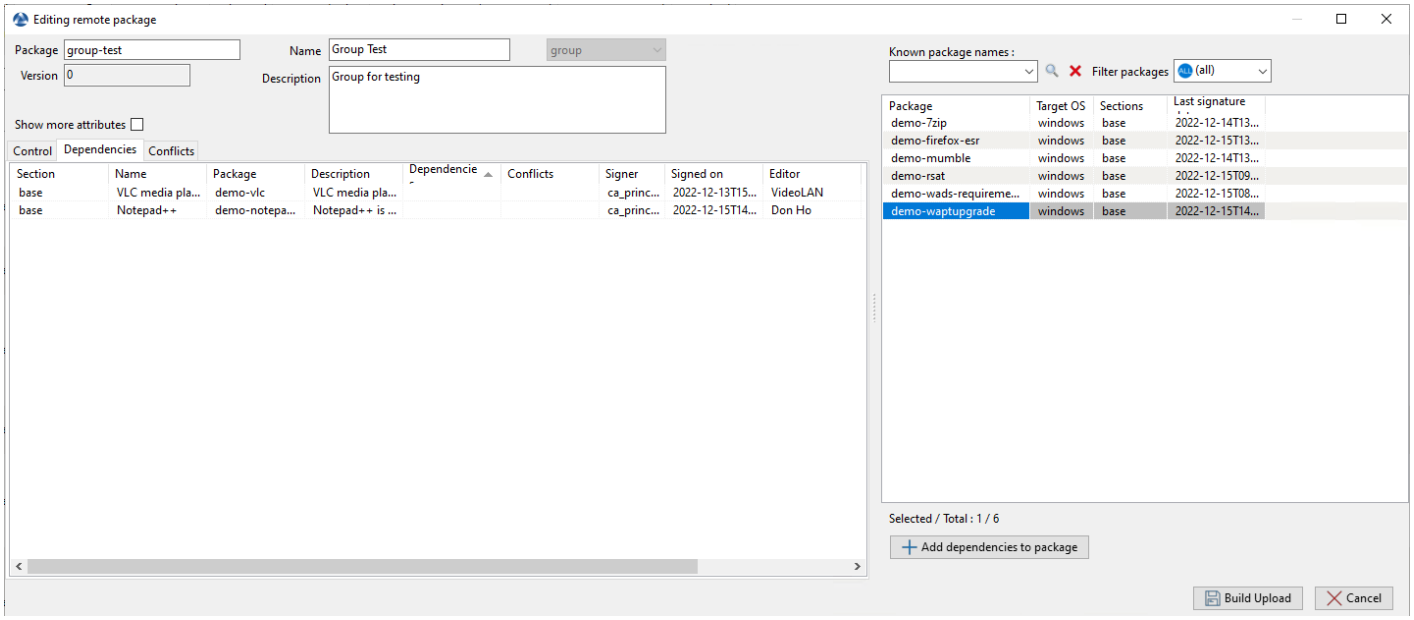
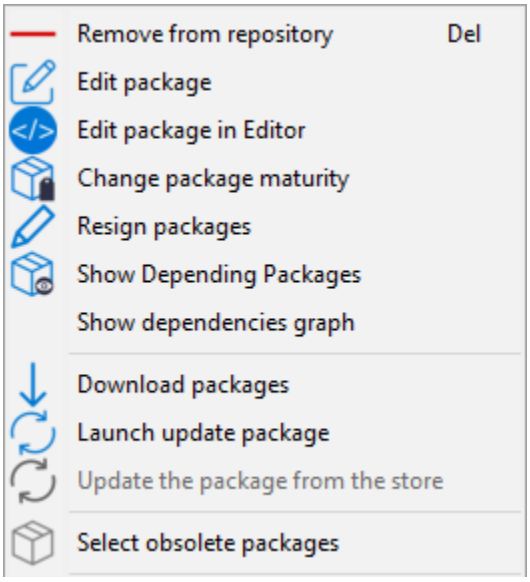


Figure25: Ajouter un paquet interdit à la machine



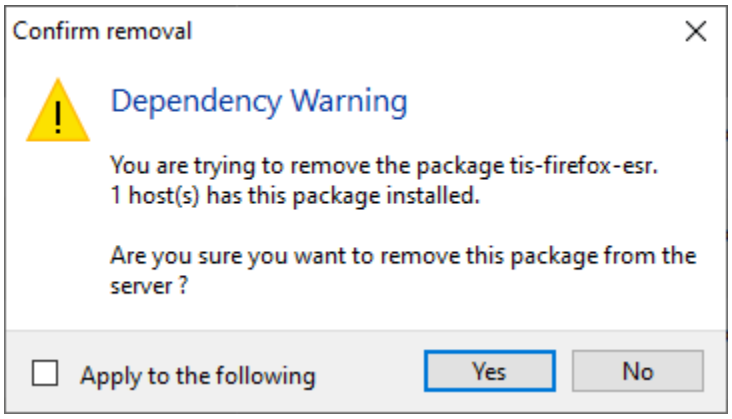
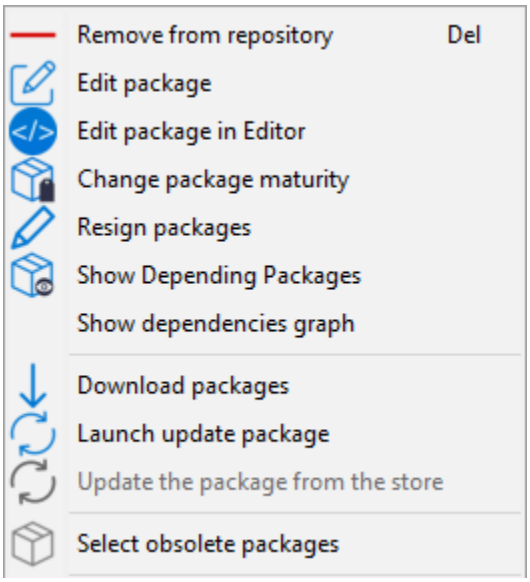


Figure26: Fenêtre signalant qu'un paquet est utilisé par au moins une machine lors de la tentative de suppression d'un paquet du dépôt local

26.12.9 Editer un paquet WAPT



Pour éditer un paquet, *clique-droit → Editer le paquet*.

Le paquet WAPT sera téléchargé localement dans **le dossier de développement des paquets de base**, défini dans les paramètres de la console WAPT.

Si **`PyScripter`** est installé, il ouvrira automatiquement les fichiers `control` et `setup.py`.

Une fois édité vous pouvez téléverser *depuis la console wapt*.

26.12.10 Déployer des paquets WAPT puis la console WAPT

Vous pouvez déployer des paquets de multiples façons:

- Directement en *ajoutant un paquet a le ou les hôte(s) sélectionné(s).*
- En *ajoutant un paquet WAPT à une Unité Organisationnelle* dont l'hôte est membre.
- En *ajoutant un paquet à un profile d'hôte* qui est appliqué à l'hôte.
- En *ajoutant le paquet à un paquet groupe* dont l'hôte est membre.

26.13 Créer un paquet de configuration

Les paquets WAPT **configuration** permettent de créer plusieurs configurations WAPT sans avoir à créer plusieurs agents WAPT.

Pour créer un paquet de configuration, aller dans l'onglet *Paquets WAPT* de la Console WAPT, puis cliquer sur le bouton *Faire un modèle de paquet à partir d'un fichier de configuration* et enfin choisir l'élément de menu *Configuration dynamique de l'agent WAPT*.

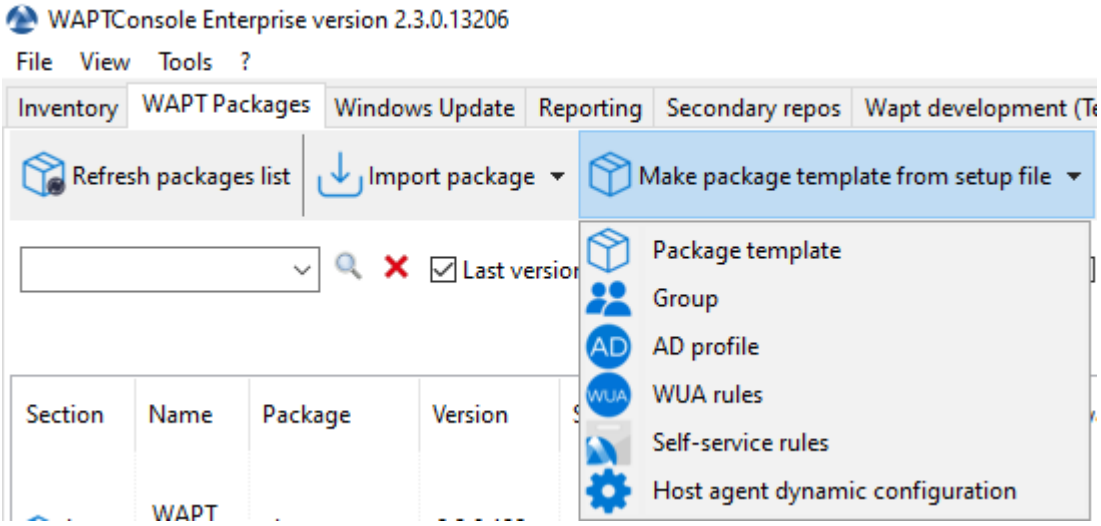


Figure27: Grille des groupes de paquets

Une nouvelle fenêtre s'ouvrira afin de créer le nouveau paquet de configuration.

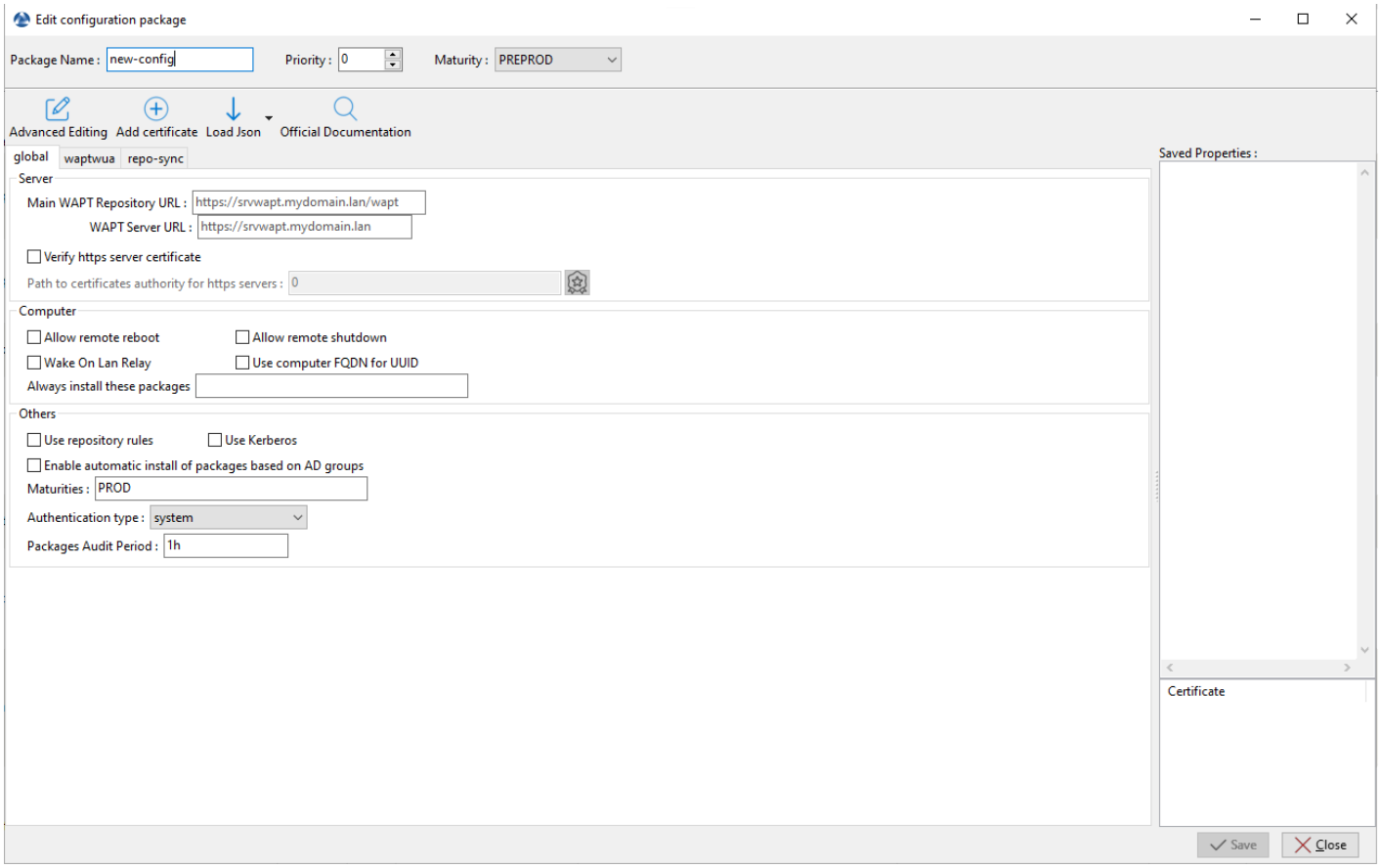
Changer le nom dans le champ *Nom du paquet*. Maintenant, il y a plusieurs choses qui peuvent être faites.

26.13.1 La barre d'outils de configuration

L'option *Configuration avancée* transformera la vue simplifiée en une liste complète d'options. Veuillez consulter cette documentation pour *les options avancées de l'Agent WAPT*.

Vous pouvez utiliser un filtre pour rechercher le nom d'un attribut.

- L'option *Créer un nouveau dépôt* vous permettra de créer un nouveau dépôt WAPT si vous avez un autre dépôt que le vôtre. Si vous le faites, un nouvel onglet nommé avec le nom du dépôt apparaîtra dans le paquet de configuration.


Edit configuration package
Package Name : new-config
Priority : 0
Maturity : PREPROD
Advanced Editing
Add certificate
Load Json
Official Documentation
global
waptwua
repo-sync
Server
Main WAPT Repository URL : https://srvwapt.mydomain.lan/wapt
WAPT Server URL : https://srvwapt.mydomain.lan
Verify https server certificate
Path to certificates authority for https servers : 0
Computer
Allow remote reboot
Allow remote shutdown
Wake On Lan Relay
Use computer FQDN for UUID
Always install these packages
Others
Use repository rules
Use Kerberos
Enable automatic install of packages based on AD groups
Maturities : PROD
Authentication type : system
Packages Audit Period : 1h
Saved Properties :
Certificate
Save
Close

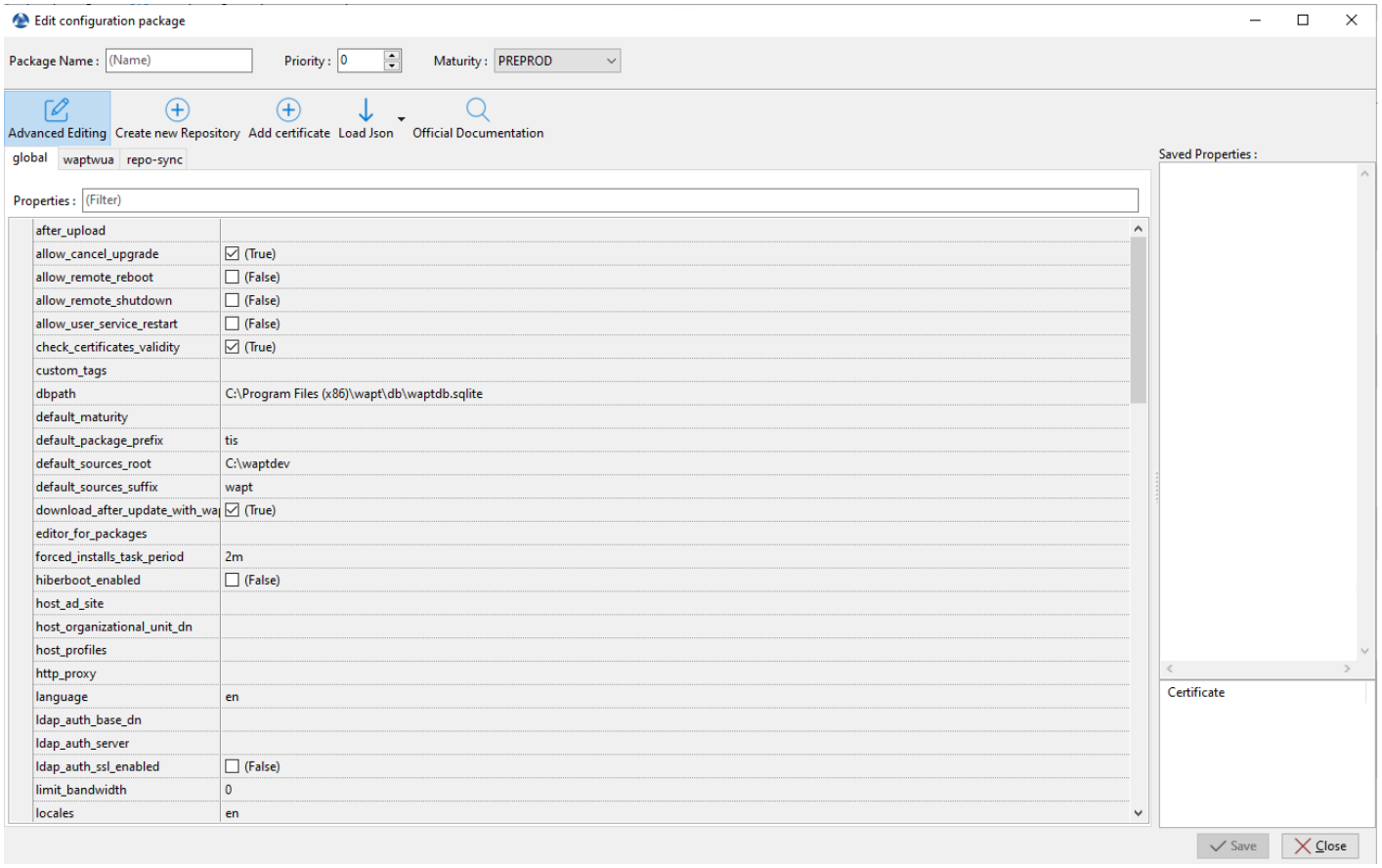
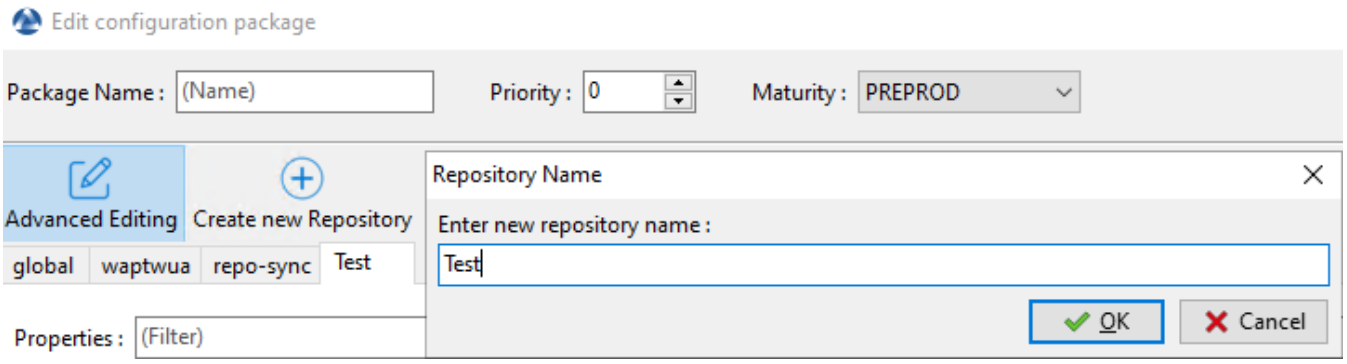


Figure28: Formulaire de configuration avancée d'un Agent WAPT



- Le bouton *Ajouter un certificat* ouvrira une nouvelle fenêtre qui permettra de sélectionner un certificat. Le certificat sera inclus dans le paquet de certificats autorisés à effectuer des actions sur les machines.

Figure29: Ajouter un certificat dans un paquet de configuration WAPT

Vous pouvez supprimer le certificat avec *Clic droit* → *Supprimer le certificat*.

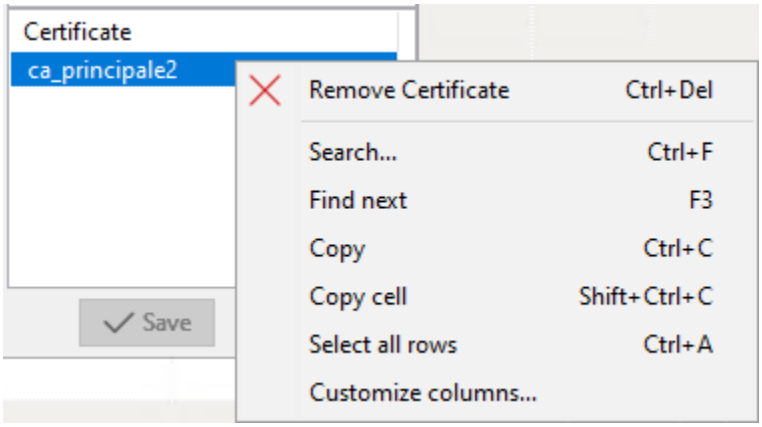


Figure30: Supprimer un certificat dans un paquet de configuration WAPT

- Le bouton *Charger un fichier json* importera un fichier `json` qui peut être un fichier de configuration.

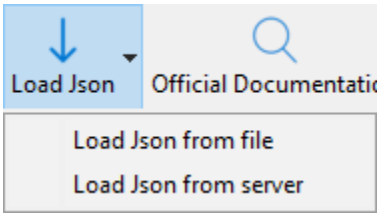


Figure31: Chargement d'un fichier json depuis un fichier ou depuis le serveur WAPT

Si vous choisissez de télécharger un fichier de configuration `json` depuis le serveur WAPT, vous pouvez choisir parmi les configurations existantes.

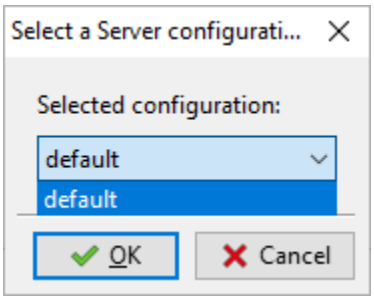


Figure32: Sélectionner un fichier json sur le Serveur WAPT

- Le bouton *Documentation officielle* ouvrira un lien vers la documentation de Tranquil IT avec toutes les options du fichier `wapt-get.ini`.

26.13.2 Les onglets de configuration

Par défaut, il y a 3 onglets : *global*, *waptwua* et *repo-sync*.

- L'onglet *global* aura les mêmes options lorsque vous créerez un *Agent WAPT*. Le volet de droite résume les options qui seront définies.

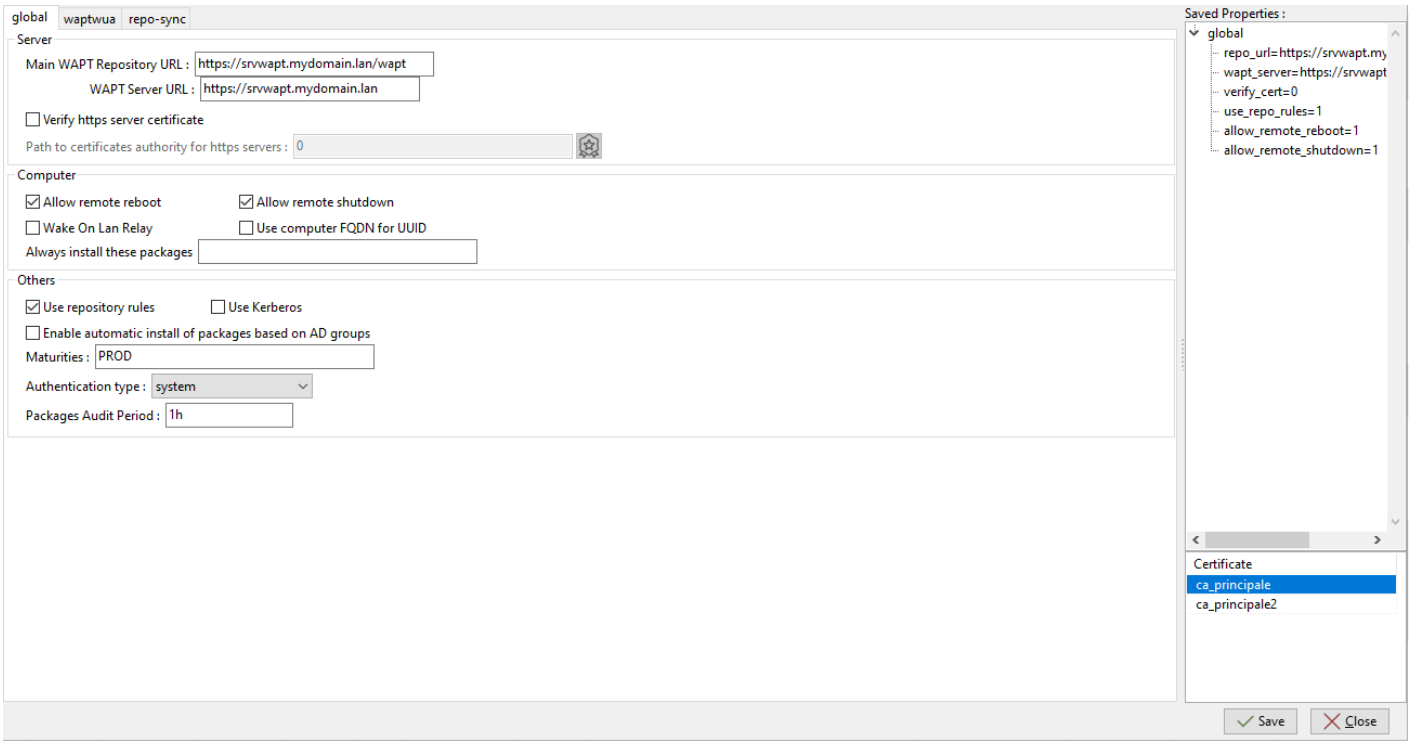


Figure33: Onglet global pour configurer un paquet WAPT de configuration

- L'onglet *waptwua* vous aidera à choisir les options à utiliser avec *WAPT Windows Updates*.
- L'onglet *repo-sync* aidera à configurer la machine distante pour qu'elle devienne un *dépôt distant*.

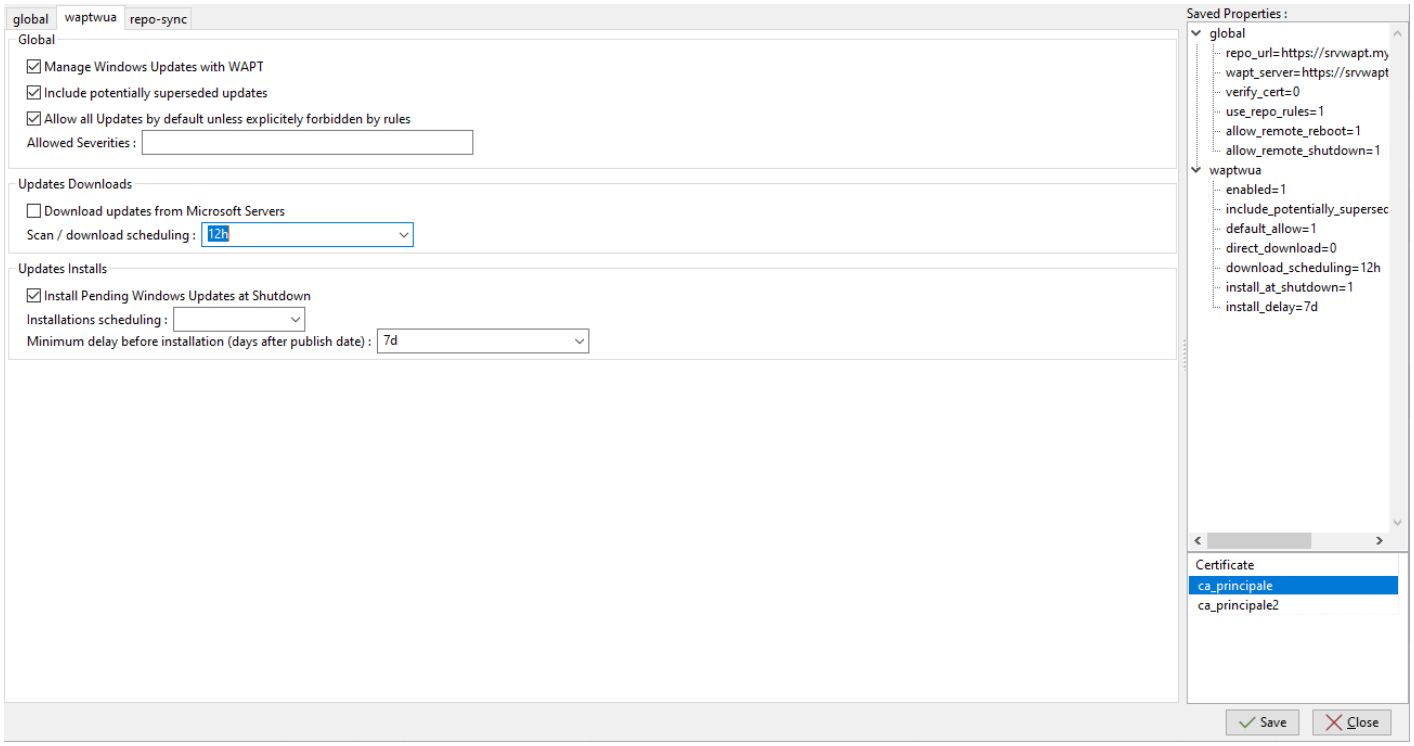


Figure34: Onglet pour permettre à une machine distante d'utiliser les mises à jour Windows avec WAPT

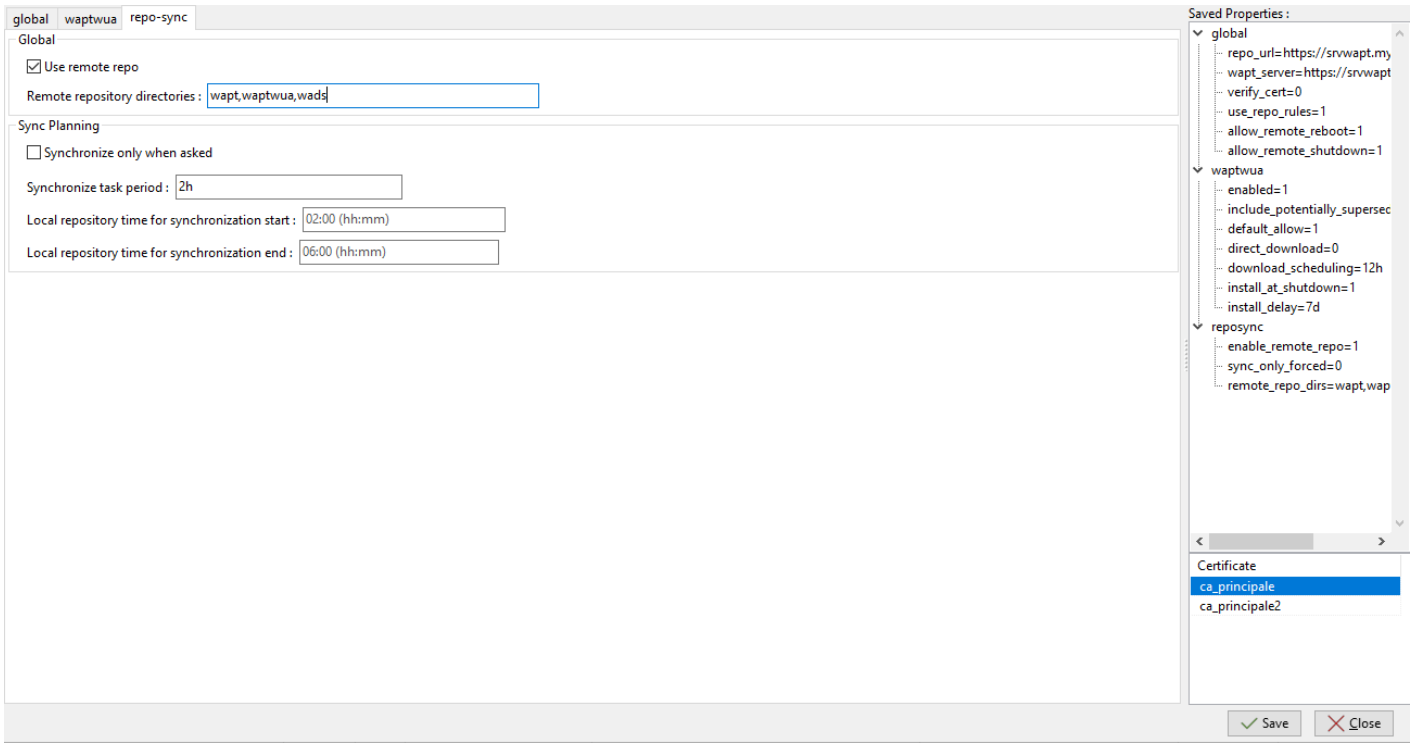


Figure35: Onglet pour configurer un paquet WAPT pour que la machine distante devienne un dépôt

26.13.3 Supprimer une section de configuration

Pour supprimer une option ou une section entière, utiliser *Clic droit → Supprimer la propriété sélectionnée*.

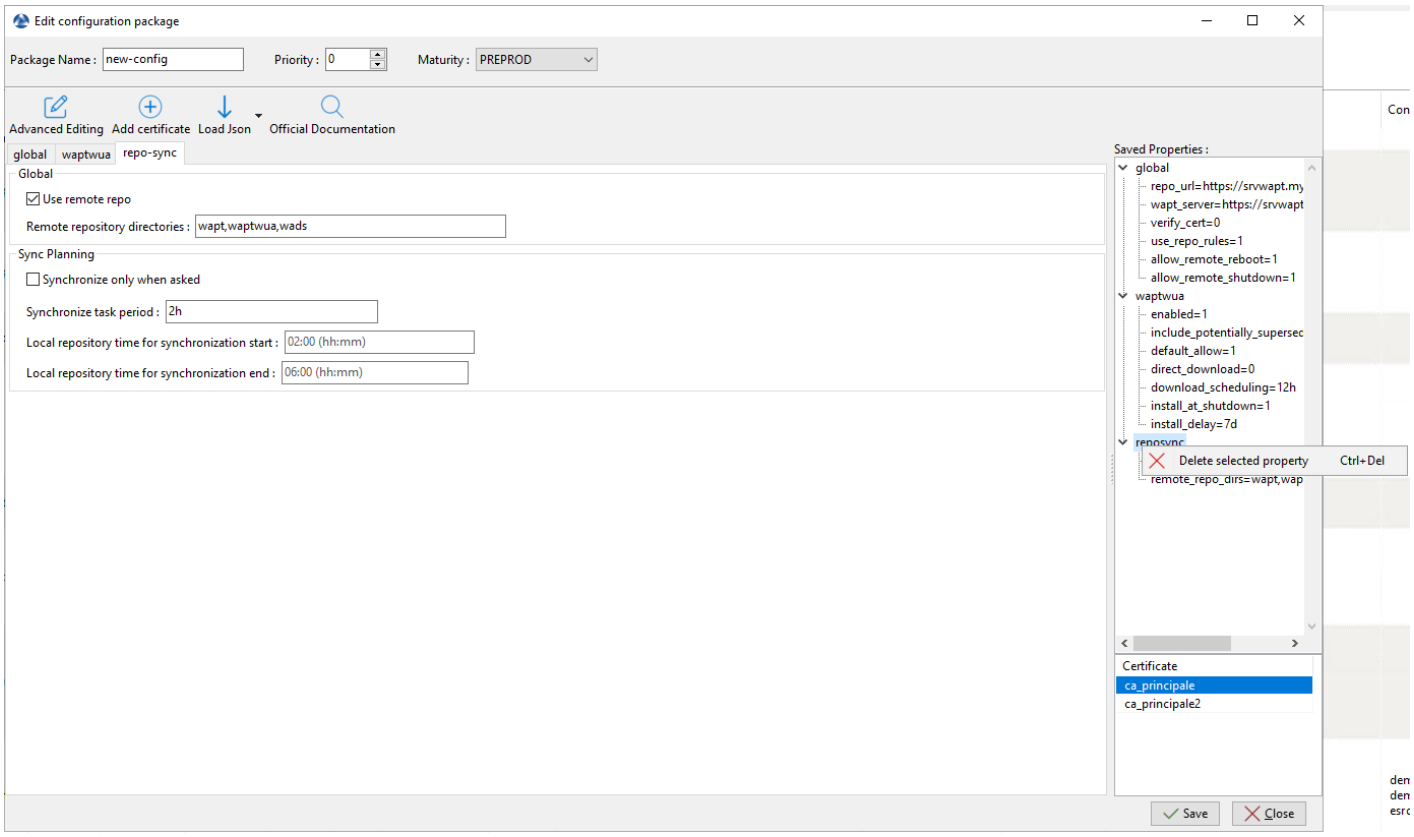


Figure36: Supprimer un attribut, une option ou une section

26.13.4 Enregistrer le paquet de configuration et appliquer la nouvelle configuration à une machine distante

Cliquer sur le bouton *Enregistrer* pour sauvegarder le nouveau paquet de configuration. Le fichier de configuration sera téléchargé sur le Serveur WAPT au format `.json`. Sélectionner les machines et appliquer le paquet de configuration comme n'importe quel autre paquet WAPT.

Sur la machine, le paquet de configuration sera situé dans le répertoire d'installation WAPT `conf.d` dans un format `json` qui peut ressembler à ceci :

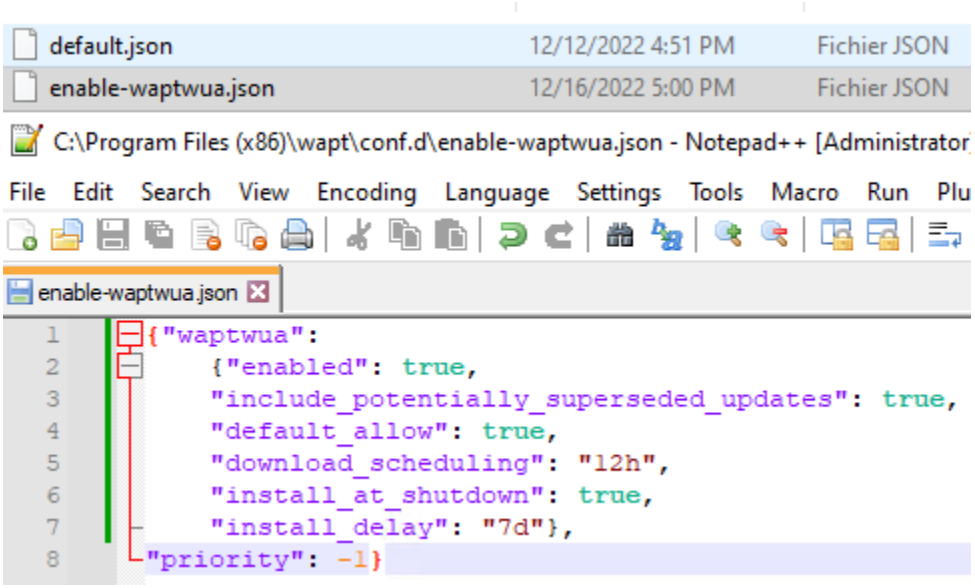


Figure37: Exemple d'un fichier de configuration réalisé par un paquet de configuration WAPT

CHAPTER 27

Utilisation des fonctions avancées de la console WAPT

Cette page détaille l'utilisation avancée de la Console WAPT.

27.1 Utiliser des paquets profile dans WAPT

27.1.1 Principe de fonctionnement

WAPT Enterprise propose une fonctionnalité de paquet *profile* Active Directory.

Le paquet *profile* automatise l'installation de paquets WAPT et des paquets de configuration sur les machines en fonction de leur appartenance aux groupes de sécurité informatique Active Directory.

L'Agent WAPT signalera au Serveur WAPT les groupes Active Directory auxquels la machine appartient.

Si un paquet *profile* a le même nom qu'un groupe Active Directory, l'agent WAPT installera automatiquement le paquet *profile* pour le groupe Active Directory dont l'hôte est membre.

Si la machine n'est plus membre de son groupe Active Directory, le paquet *profile* correspondant sera désinstallé.

Les paquets *profile* sont stockés dans le répertoire web https://srvwapt.mydomain.lan/wapt/.

Les paquets *profile* ne sont pas explicitement affectés à une machine (c'est-à-dire en tant que dépendances dans le paquet *host*) mais sont implicitement pris en compte par le moteur de dépendance de l'Agent WAPT lors des mises à niveau WAPT.

Note: Pour des raisons de performances, cette fonctionnalité n'est activée que si l'option `use_ad_groups` est activée dans le fichier de configuration `wapt-get.ini` de l'Agent WAPT.

Important: **Les groupes de Securité Ordinateur dans Active Directory contiennent des Ordinateurs et non des Utilisateurs**.

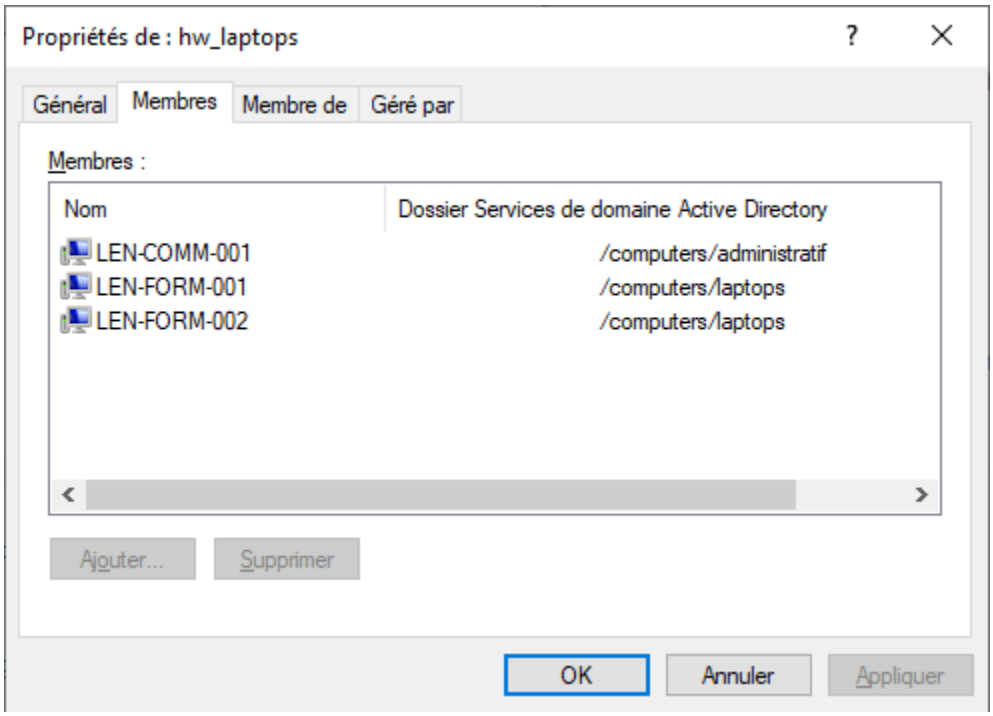


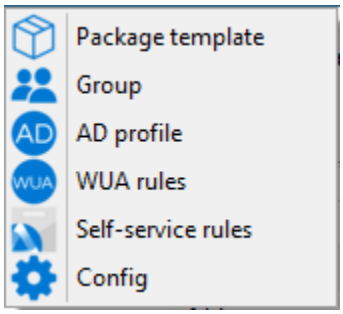
Figure1: Fenêtre montrant le groupe Computers dans Active Directory

Avertissement: L'installation automatique de logiciels et de configurations en fonction de l'utilisateur et de l'appartenance à un groupe d'utilisateurs n'est pas implémentée avec WAPT et une telle implémentation n'est pas souhaitable. Le cas d'utilisation de l'installation de logiciels en fonction du profil de l'utilisateur est mieux servi par la fonction différenciée *self-service* qui est également disponible avec WAPT Enterprise.

Le nom du groupe **DOIT** être en minuscules dans Active Directory et dans la Console WAPT.

27.1.2 Créer des paquets WAPT de type *profile* dans la Console WAPT

Vous pouvez créer des paquets groupés *profils* en cliquant sur *Faire un modèle de paquet à partir du fichier d'installation → profil AD*.



Important: Pré-requis:

- Le nom du groupe AD *profile* et le paquet WAPT de type *profile* **DOIVENT** être tout en minuscules.

Exemple:

- Groupe de Sécurité AD: **hw_laptops** ;
- Paquet WAPT de type *profile* : **HW_laptops**.

Une fenêtre s'ouvre et on vous demande de choisir quels paquets doivent être contenus dans le paquet **profile** tout juste créé.

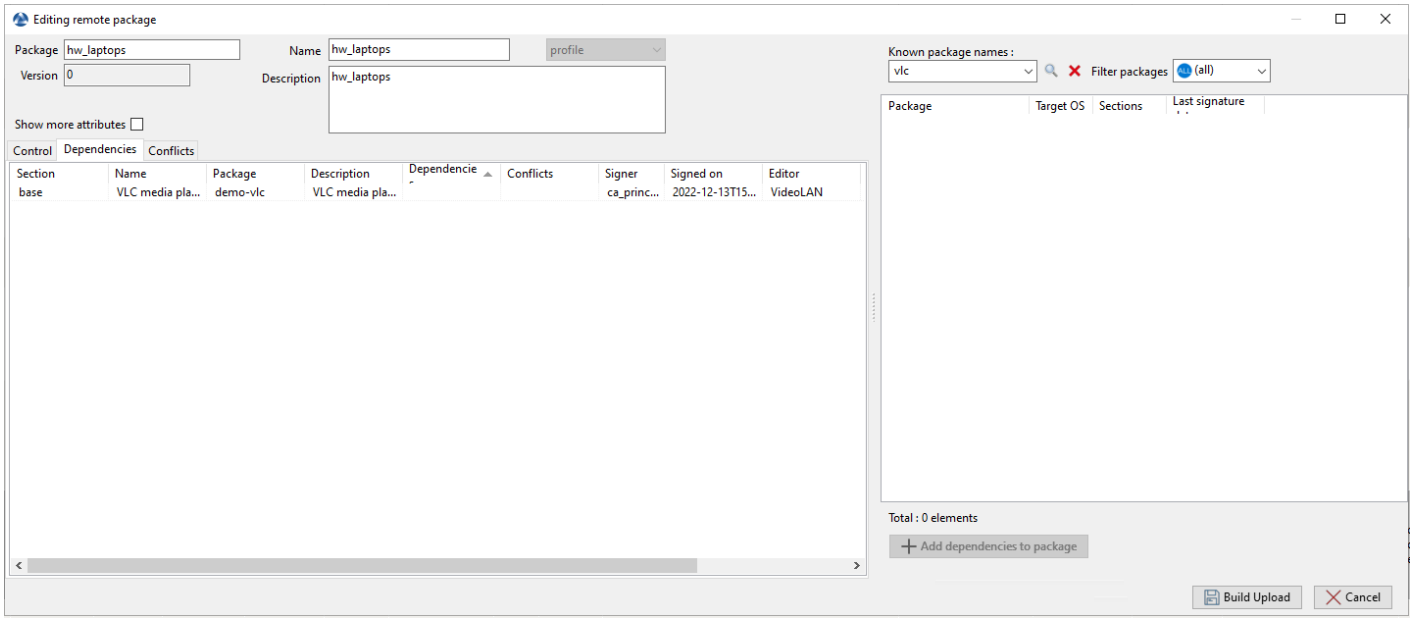


Figure2: Ajout de paquets WAPT à un paquet *profile*

Enregistrez le paquet *profil* et il sera téléchargé sur le serveur WAPT.

27.2 Utiliser des Unités Organisationnelles dans WAPT

27.2.1 Principe de fonctionnement

WAPT Enterprise propose la fonctionnalité de paquets d'Unité Organisationnelle.

Les paquets de type *unit* automatisent les installations de logiciels et de configurations en se basant sur l'arborescence d'Active Directory.

Les paquets de type *unit* ne sont pas explicitement affectés à la machine (c'est-à-dire en tant que dépendances dans le paquet hôte) mais sont implicitement pris en compte par le moteur de dépendance de l'agent WAPT lors de la mise à niveau WAPT.

Note: Si l'ordinateur est retiré d'une Unité Organisationnelle, les paquets de type *unit* obsolètes sont supprimés.

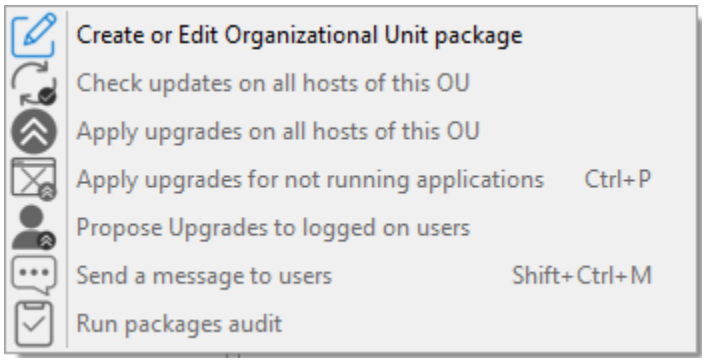
L'agent WAPT connait son emplacement dans l'arborescence Active Directory, il connaît donc pour cette raison la hiérarchie des Unités Organisationnelles qui le concerne, par exemple:

```
DC=ad,DC=mydomain,DC=lan
OU=Paris,DC=ad,DC=mydomain,DC=lan
OU=computers,OU=Paris,DC=ad,DC=mydomain,DC=lan
OU=service1,OU=computers,OU=Paris,DC=ad,DC=mydomain,DC=lan
```

Si un paquet de type *unit* est défini au niveau de chaque Unité Organisationnelle, l'Agent WAPT téléchargera automatiquement les paquets WAPT et les configurations qui sont attachés à chaque niveau. En utilisant l'héritage, WAPT appliquera les paquets WAPT et les dépendances qui sont attachés à chaque Unité Organisationnelle.

27.2.2 Créer des paquets d'Unité Organisationnelles dans la console WAPT

Vous pouvez créer des paquet *unit* en faisant *clic-droit sur une OU → Créer ou éditer le paquet de l'Unité Organisationnelle*.



Une fenêtre s'ouvre et vous êtes invité à choisir les paquets à inclure dans le paquet de type *unit*.

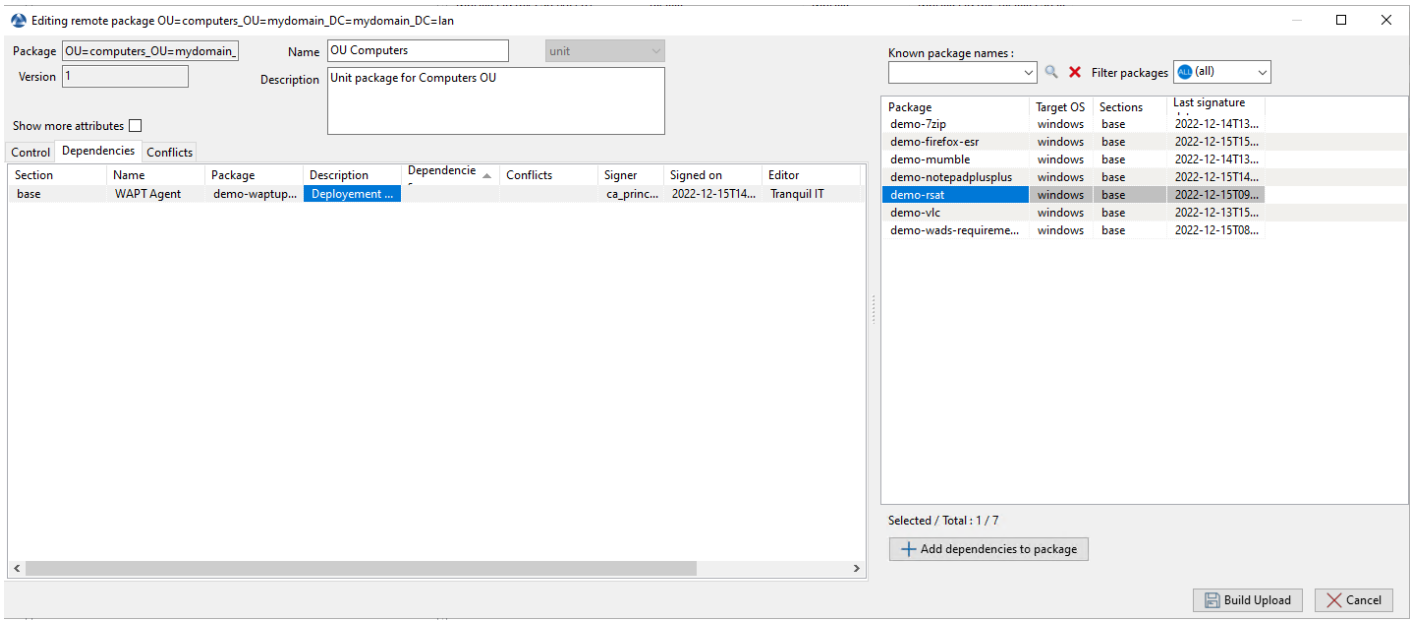
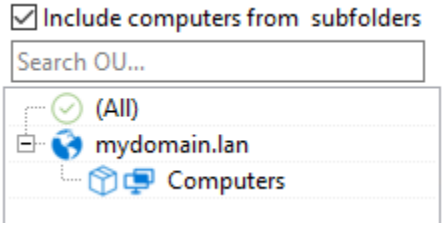


Figure3: Ajouter des paquets au paquet unit

Sauvegarder le paquet et il sera déployé sur tous les hôtes appartenant à cette OU.

Lorsque vous avez un paquet de type **unit**, vous verrez un cube devant le nom de l'UO dans la Console WAPT.



27.2.3 Les actions possibles avec les Unités Organisationnelles

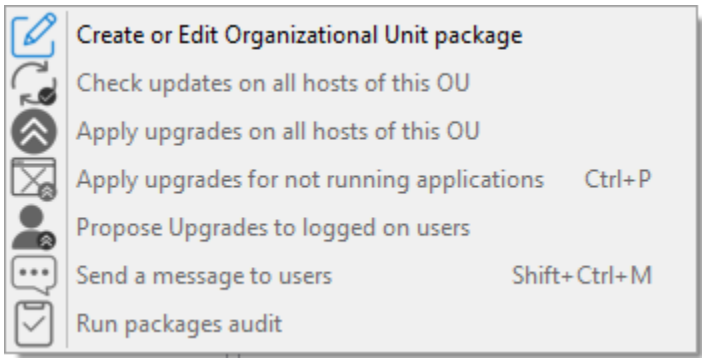
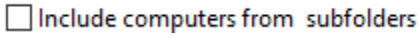


Table1: Créer ou éditer des paquets d'Unité Organisationnelles

Option de menu	Description
Créer ou éditer des paquets d'Unité Organisationnelles	*Visitez cette documentation pour plus de details sur la Création ou l'Edition de paquet OU*.
Vérifier les mises à jours des machines de cette OU	Permet de télécharger l'état actuel de l'hôte vers le serveur WAPT et de forcer le serveur WAPT à afficher si les hôtes de l"OU sélectionnée ont des mises à jour en attente.
Lancer l'installation des paquets pour les machines de cette OU	Vous pouvez voir dans l'image que les actions *update* et *upgrade* peuvent être effectuées via ce menu, sélectionnant ainsi les hôtes par leur Unité d'Organisation.

Indication: Vous pouvez filtrer la manière dont les hôtes sont affichés basé sur leur appartenance a des OU de l'Active Directory.



La case *Inclure les postes des sous-dossiers* vous permet d'afficher les hôtes des sous-dossiers.

27.2.4 Simuler des Unités Organisationnelles pour des hôtes en WORKGROUP

Il arrive que des hôtes spécifiques ne peuvent être joints à un domaine Active Directory.

Avec cette spécifité, de tels hôtes ne peuvent s'afficher dans les Unités Organisationnelles depuis votre console WAPT.

Pour que toutes les machines apparaissent dans la Console WAPT sous la bonne Unité Organisationnelle, qu'elles soient jointes à un domaine AD ou non, WAPT permet de spécifier une Unité Organisationnelle usurpée dans le fichier de configuration de l'Agent WAPT.

Les bénéfices de cette astuce sont:

- Vous pouvez gérer les hôtes avec WAPT comme s'ils étaient joints à l'AD.
- Les hôtes hors-du-domaine et en WORKGROUP s'affiche désormais dans l'arborescence AD.
- Les paquet *Unit* sont utilisables sur ces hôtes.

Pour configurer une *fausse* Unité Organisationnelle sur les hôtes, créez un *paquet WAPT vide*, puis utilisez le code suivant:

```
# -*- coding: utf-8 -*-
from setuphelpers import *

uninstallkey = []

def install():

  print('Setting Fake Organizational Unit')
  fake_ou = "OU=REAL_AD_SUB_OU,OU=REAL_AD_OU,DC=MYDOMAIN,DC=LAN"
  inifile_writestring(WAPT.config_filename,'global','host_organizational_unit_dn',fake_ou)

  print('Reload WAPT configuration')
  WAPT.reload_config_if_updated()

def update_package():
  pass
```

Le `host_organizational_unit_dn` sera comme ci-dessous dans `wapt-get.ini` :

```
[global]
host_organizational_unit_dn=OU=REAL_AD_SUB_OU,OU=REAL_AD_OU,DC=MYDOMAIN,DC=LAN
```

Note:

- Tenez-vous-en à un cas spécifique avec votre `host_organizational_unit_dn` (ne mélangez pas les « dc « s et les « DC « s, les « ou « s et les « OU « s . . .).
- Respectez la casse utilisée dans les champs *DN/computer_ad_dn* dans la grille d'inventaire machines.

27.3 Ajouter des plugins dans la Console

Pour ajouter des plugins, aller dans *Outils*, `Préférences` et onglet `Outils externes`.

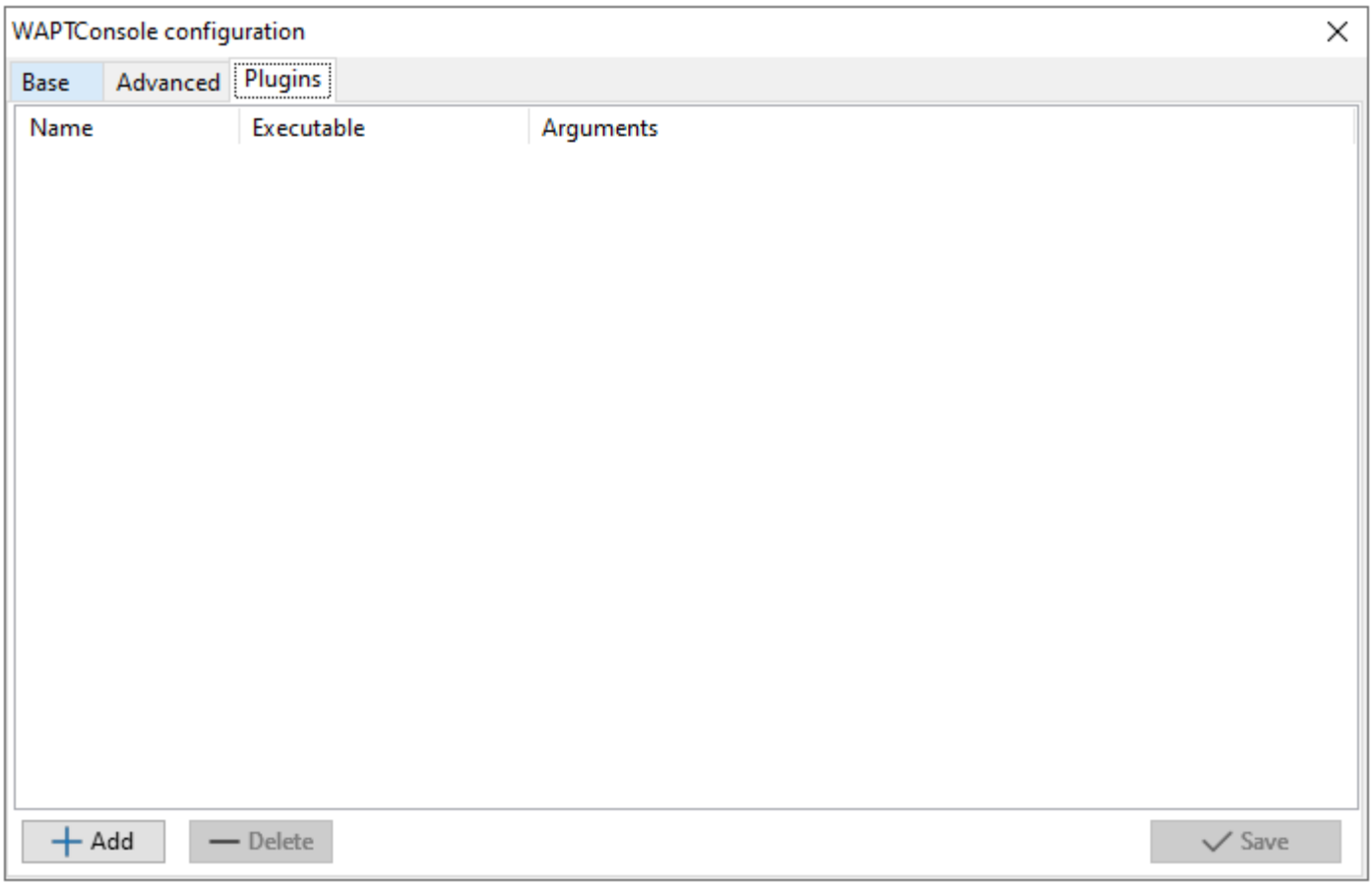


Figure4: Créer un outil externe personnalisé pour WAPT

Cliquez sur *Ajouter* pour ajouter un plugin, puis éditez les colonnes correspondantes.

Colonne	Description
Nom	Le nom qui apparaîtra dans le menu.
Exécutable	Chemin de l'exécutable qui sera exécuté après le clic.
Arguments	Arguments passés à l'exécutable. Tous les paramètres qui sont affichés dans la grille peuvent être utilisés, comme {ip}, {uuid} ou {computer_fqdn}. Pour obtenir le nom du paramètre, vous pouvez faire un clic droit sur l'en-tête de la colonne, et le nom sera affiché en paranthèses à côté du nom de la colonne.

Les plugins vont alors aparraître dans le menu:

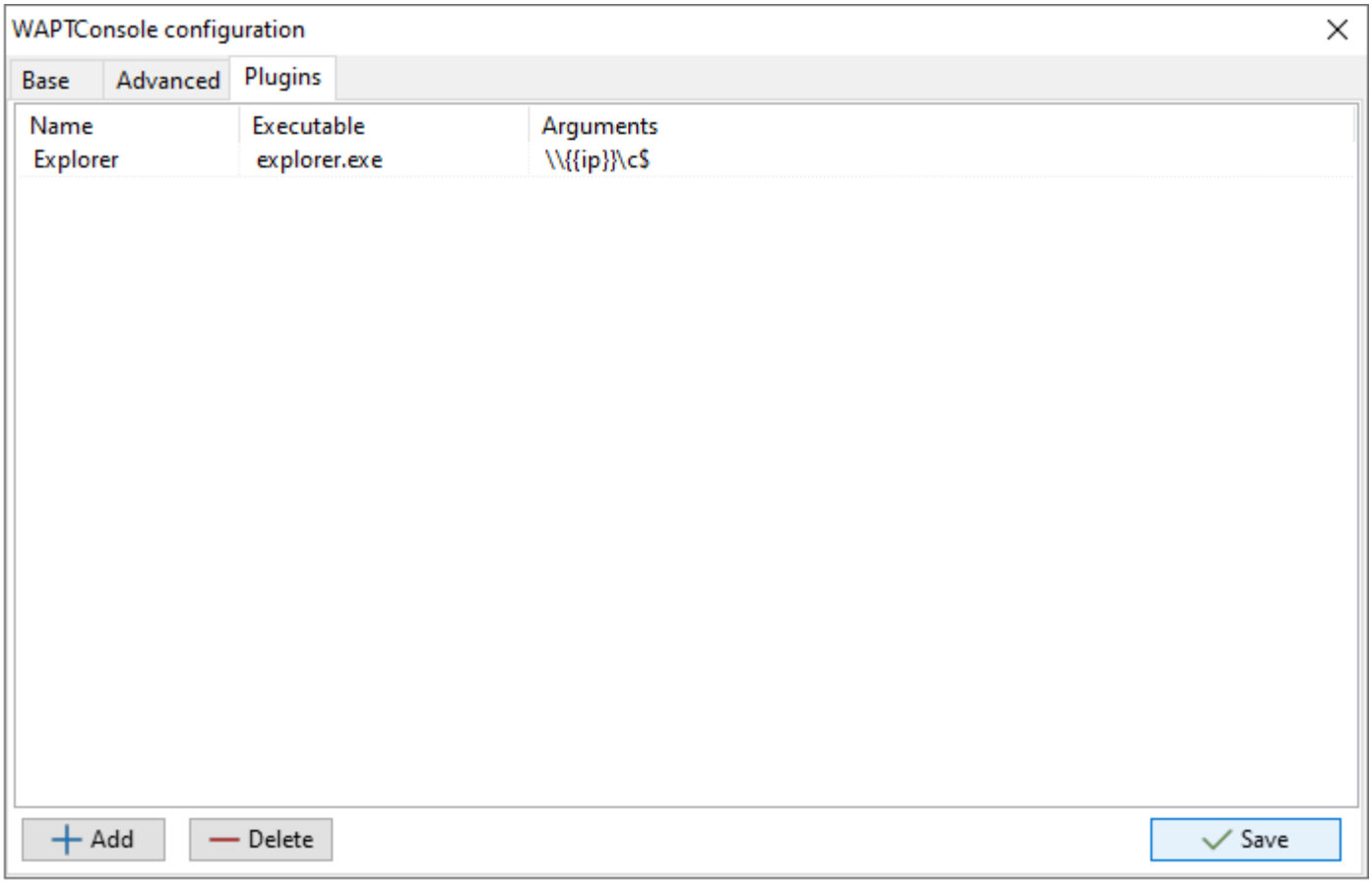


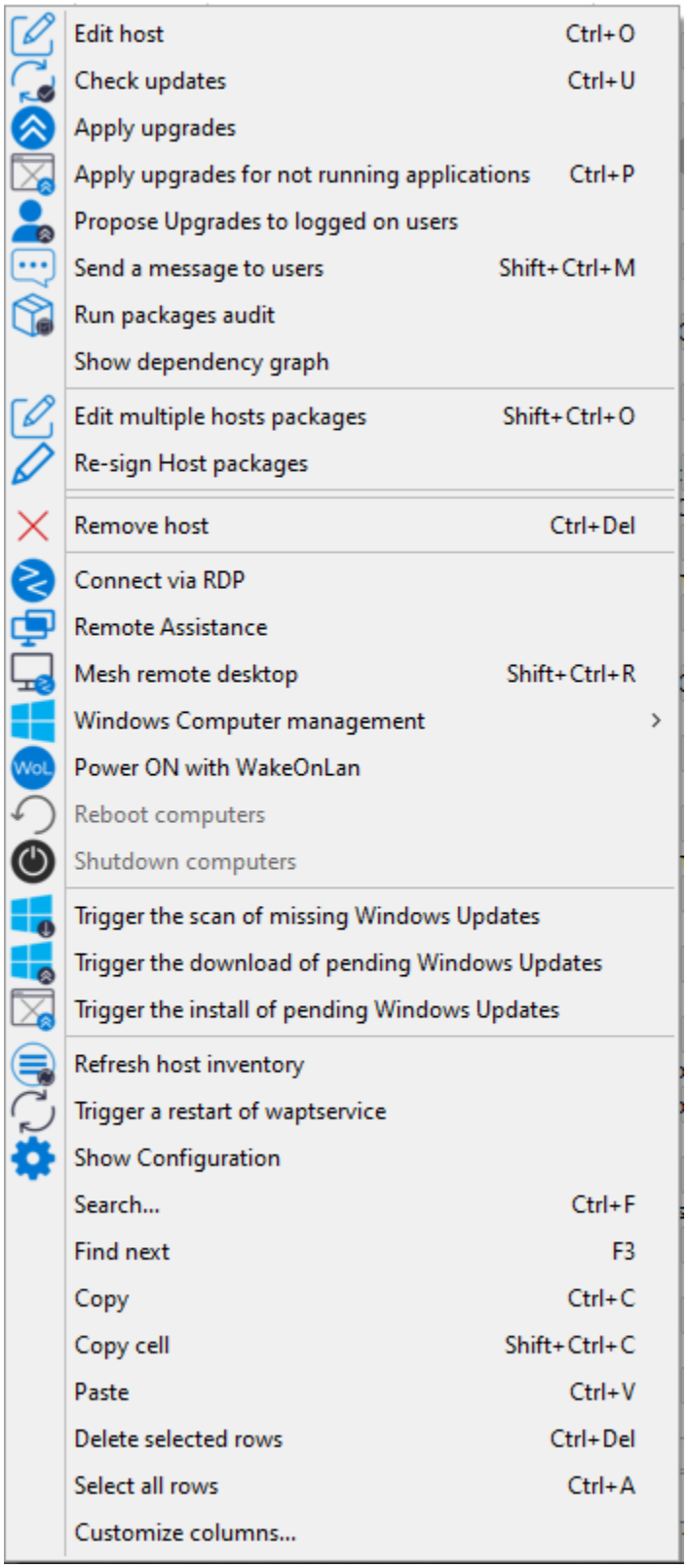
Figure5: Créer un outil externe personnalisé pour WAPT

27.4 Re-Signer tous les paquets avec la console WAPT

Cette méthode pour re-signer tous les paquets hôtes est utile lorsque la méthode cryptographique sous-jacente ou que la librairie change, comme ce cas-ci lors de la mise à jour de WAPT 1.8.2 (basé sur Python 2.7) et WAPT >= 2.0 (basé sur Python 3.x).

Utilisez le certificat Administrateur pour re-signer les paquets.

- Sélectionnez tous les hôtes.
- Clic-droit sur les hôtes sélectionnés.



- Sélectionnez *Resigner les paquets de configuration machine*.

- Confirmez la re-signature sur les hôtes sléectionnés.

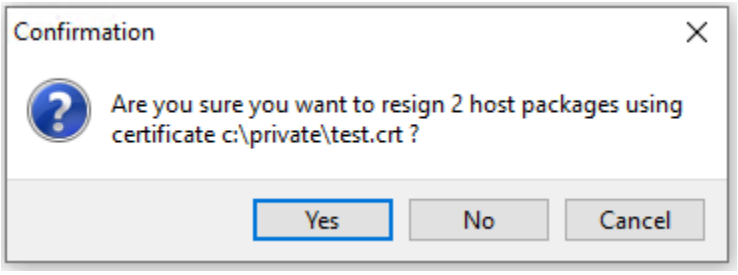


Figure6: Fenêtre modale pour confirmer la re-signature sur les machines sélectionnées

- Puis, entrez votre clé privée.

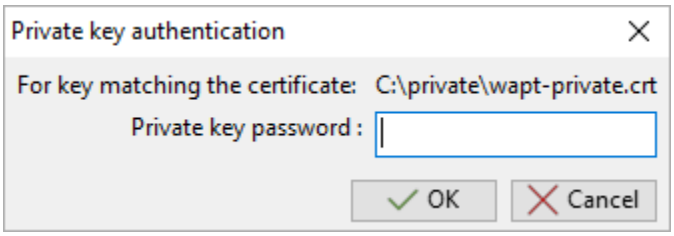


Figure7: Entrer le mot de passe pour déchiffrer la clé privée

- Les paquets *hôtes* sélectionnés sont désormais tous re-signé avec la nouvelle méthode cryptographique exigée par Python3.
- Accédez au dépôt depuis votre console WAPT.
- Sélectionnez tous les paquets dans le dépôt, puis clic-droit sur la sélection.
- Sélectionnez *Resigner les paquets*.
- Pour lancer le processus de signature, cliquez sur *Resigner les paquets*.
- Après un traitement qui peut prendre un certain temps, tous les paquets seront re-signés.

Attention: ⓘ microsoft-office 16.0.12325.20276-2 PROD ERROR Access violation

Si l'erreur **Access violation** apparaît, c'est parce que le paquet est trop gros.

You can resign this packages *using the command line*.

And if it's still not working, you can still manually edit the package and visit *this procedure for signing large WAPT packages*.

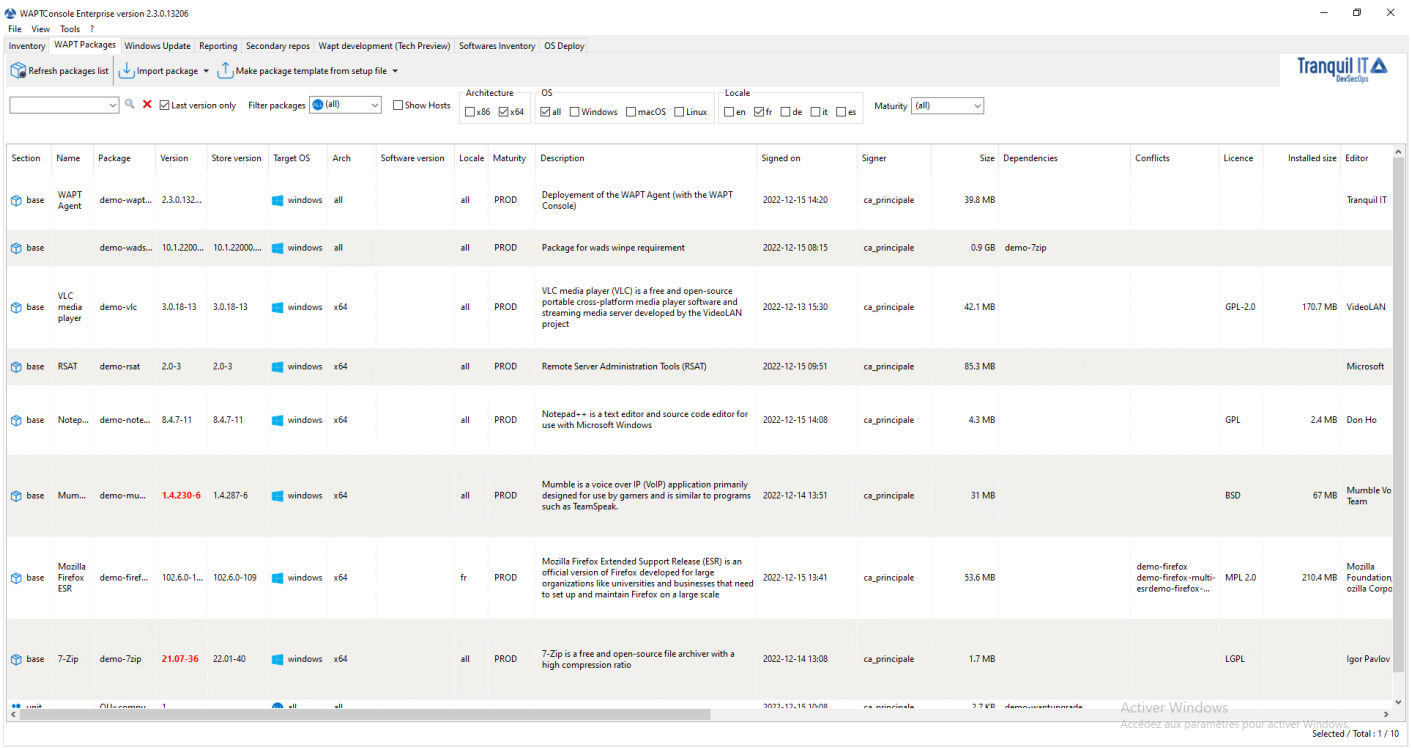


Figure8: Les dépôts disponibles sur la console WAPT

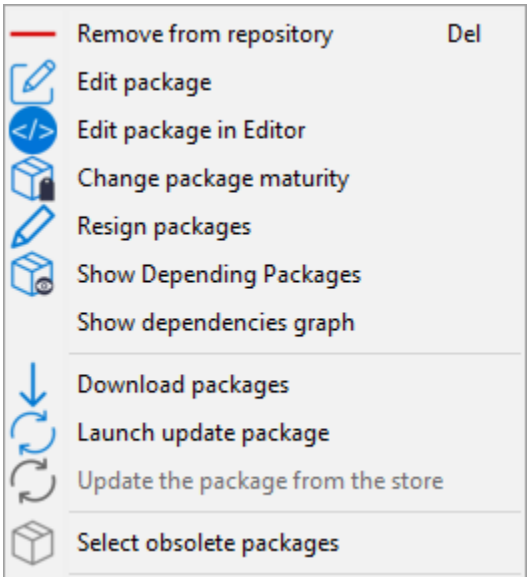


Figure9: Options de menu pour les référentiels

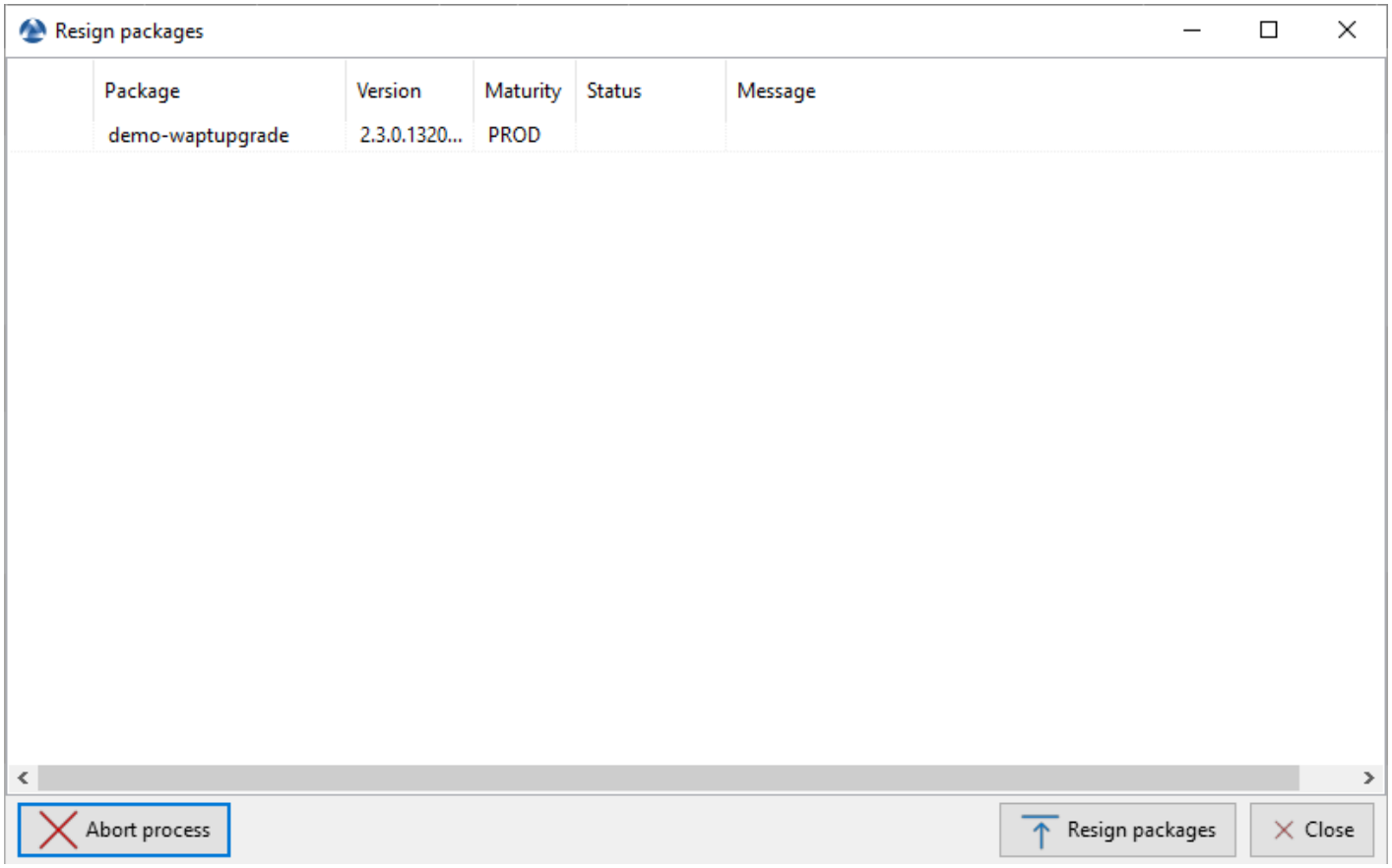


Figure10: Fenêtre de re-signature des paquets WAPT

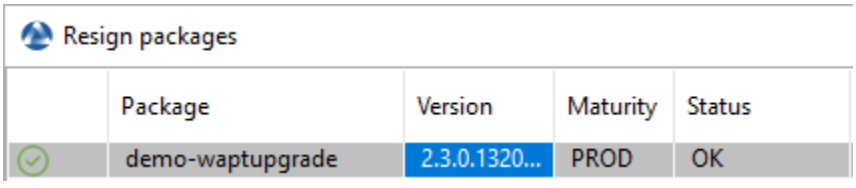


Figure11: Le processus de signature s'est correctement terminé

27.5 Afficher l'onglet des certificats dans la Console WAPT

Dans cet onglet, vous pouvez voir les certificats auxquels la machine accepte de faire confiance.

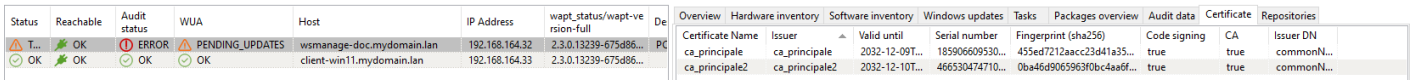


Figure12: Fenêtre montrant les certificats approuvés par la machine sélectionnée.

27.6 Afficher les données d'audit de la machine dans la Console WAPT

Vous pouvez gérer la sortie d'audit et afficher le résultat de l'audit si vous activez l'option dans l'onglet *Visualisation* → *Préférences d'affichage*. Cochez l'option *Montrer l'onglet des données d'audit de la machine* pour voir l'onglet *Données d'audit* sur chaque client.

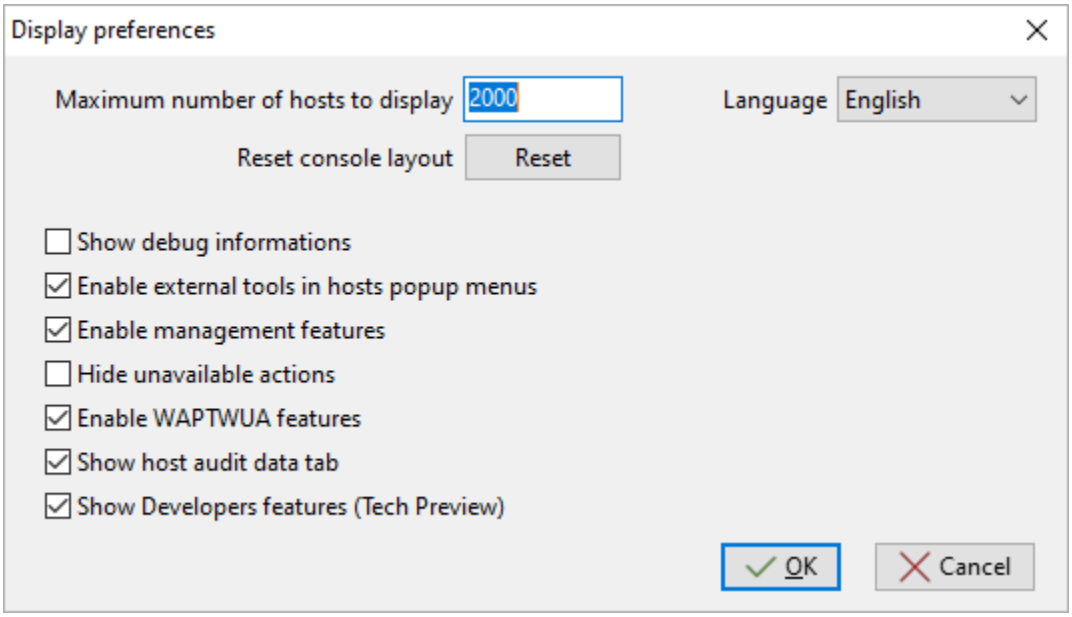


Figure13: Fenêtre affichant les préférences avancées

Pour utiliser les audits dans les paquets WAPT, visitez *cette page pour gérer les données d'audit*.

27.6.1 Afficher les données chiffrées avec un certificat dans l'onglet des données d'audit

Avec la fonction d'audit, il est possible de chiffrer les données sensibles provenant de machines distantes ; il sera possible de lire les données sensibles chiffrées avec un certificat installé sur la machine de l'administrateur WAPT. De cette façon, le Serveur WAPT peut déposer des données sensibles d'inventaire sans que le serveur devienne un bien sensible.

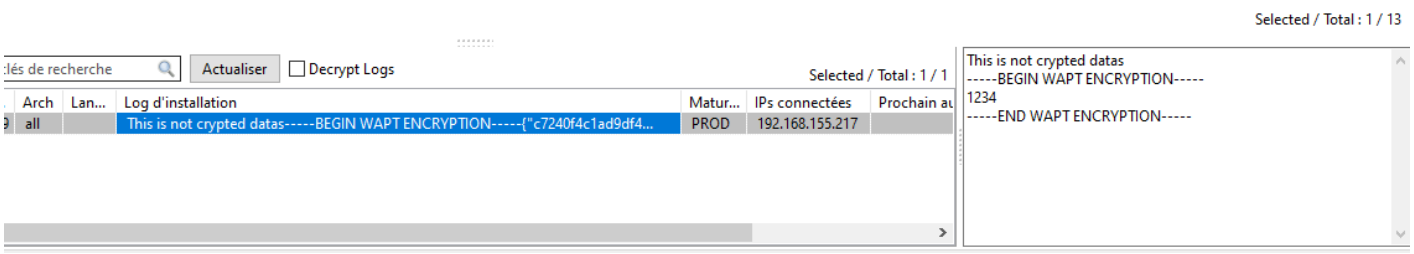
Cette méthode est particulièrement utile par exemple pour gérer en toute sécurité les mots de passe aléatoires LAPS (Local Administrator Password Service) dans WAPT.

Si vous disposez de la clé privée correspondant au certificat qui a été utilisé pour chiffrer les données, celles-ci seront déchiffrées et apparaîtront en clair.

Voici un exemple de code :

```
# -*- coding: utf-8 -*-
from setuphelpers import *
from waptcrypto import print_encrypted_data

def audit():
  randompassword = '1234'
  print_encrypted_data(randompassword, glob.glob('*.crt'))
```

Ce code va chiffrer le mot de passe *1234* avec tous les certificats présents sur la machine qui est utilisée pour gérer WAPT. Depuis la Console WAPT, vous verrez dans l'onglet *Données d'audit* la version chiffrée et vous pourrez déchiffrer les données avec la clé privée associée au certificat public qui a été utilisé pour chiffrer les données.



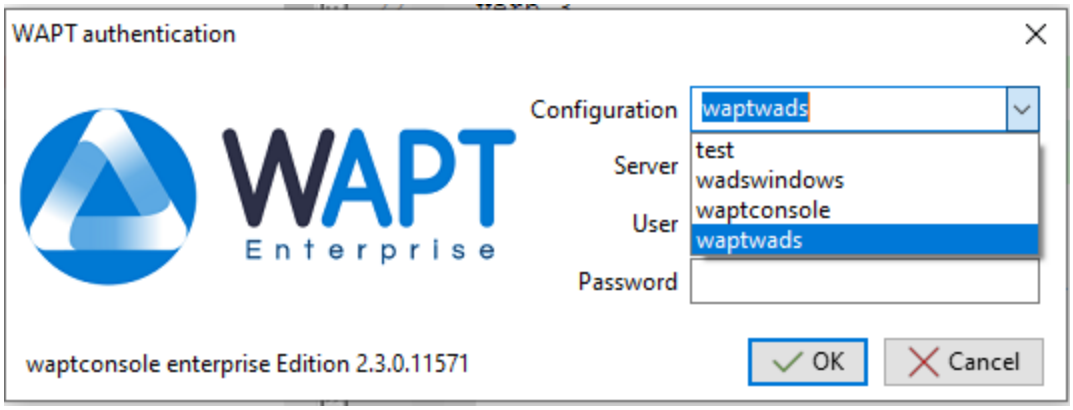
27.7 Gérer plusieurs profils de Serveur WAPT dans la Console WAPT

Vous pouvez connecter la Console WAPT à plusieurs Serveurs WAPT.

Pour ce faire, aller dans `%localappdata%waptconsole`, copier le fichier `waptconsole.ini` et renommez-le, par exemple `waptconsole2.ini`. Modifiez le nouveau fichier avec les paramètres du deuxième Serveur WAPT (ex : IP / DNS, préfixe, etc).

Ensuite, lorsque vous rouvrez la Console WAPT, vous pouvez sélectionner un Serveur WAPT ou l'autre.

Indication: Vous pouvez avoir plusieurs profils de connexion au Serveur WAPT mais les Serveurs WAPT ne communiquent pas entre eux.


WAPT authentication
Configuration
waptwads
Server
User
Password
test
wadswindows
waptconsole
waptwads
WAPT
Enterprise
waptconsole enterprise Edition 2.3.0.11571
OK
Cancel

CHAPTER 28

Utiliser les WAPT Windows Update Agent (WAPTWUA)

Indication: WAPT peut gérer les mises à jours Windows sur vos équipements et remplacer les mises à jour automatiques ou un serveur WSUS.

Note: WAPTWUA fonctionne avec l'API WUA (Windows Update Agent) de Windows.

Les fonctions internes de WAPTWUA sont basées sur l'API WUA. Pour plus d'informations : https://docs.microsoft.com/en-us/windows/win32/wua_sdk/using-the-windows-update-agent-api.

Attention: Le WAPTWUA ne peut pas fonctionner en même temps que le store Windows.

28.1 Principe de fonctionnement

Vidéo de démonstration :

https://youtu.be/x36gAaT31Ko

Chaque Patch Tuesda (le Patch Tuesda est un terme non officiel utilisé pour désigner le deuxième mardi de chaque mois où Microsoft publie des correctifs pour ses produits logiciels.), le serveur WAPT télécharge un fichier mis à jour `wsusscn2.cab` depuis les serveurs de Microsoft.

Par défaut, le téléchargement se fait une fois par jour et auncun téléchargement ne se déclenche si le fichier `wsusscn2.cab` n'a pas changé depuis le dernier téléchargement.

Indication: Pour que WAPTWUA fonctionne, le Serveur WAPT doit avoir accès à :

- windowsupdate.microsoft.com
- ..windowsupdate.microsoft.com
- ..update.microsoft.com
- windowsupdate.com
- download.windowsupdate.com
- download.microsoft.com
- download.windowsupdate.com
- wustat.windows.com
- ntservicepack.microsoft.com
- go.microsoft.com
- dl.delivery.mp.microsoft.com

Even though you may choose other sources for Windows updates, ports 443 and 80 need to accept incoming trafic on the WAPT Server.

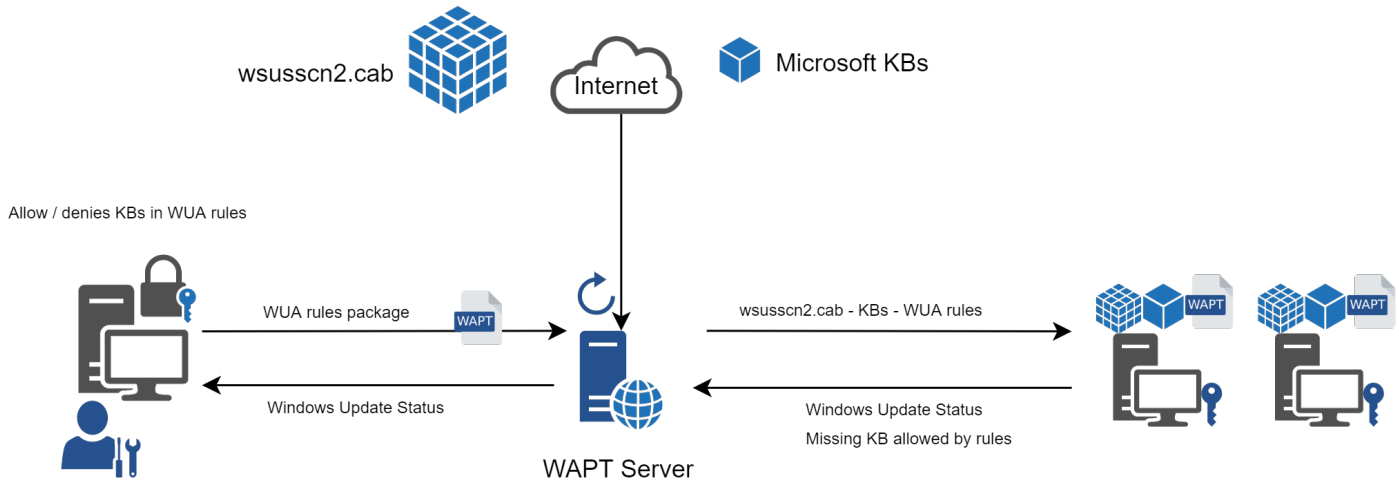


Figure1: Diagramme de flux des mises à jour Windows WAPT

Le fichier `wsusscn2.cab` est ensuite téléchargé par l'agent WAPT à partir du dépôt du serveur WAPT, puis transmis à WUA l'utilitaire Windows pour décortiquer l'arbre de mise à jour pour l'hôte.

Régulièrement, l'hôte analysera les mises à jour disponibles en utilisant le fichier `wsusscn2.cab`. L'hôte enverra sa liste de mises à jour nécessaires au serveur WAPT.

Si une mise à jour est en attente sur l'hôte et si cette mise à jour n'est pas présente sur le serveur WAPT, le serveur WAPT téléchargera la mise à jour nécessaire à partir des serveurs officiels de Microsoft.

Indication: Ce mode opératoire permet à WAPT de ne télécharger que les mises à jour nécessaires aux postes, gagnant ainsi de la bande passante, du temps de téléchargement et de l'espace disque.

Note: Sut le serveur WAPT, les mises à jour téléchargées sont sotckées:

- sur les hôtes Linux dans `/var/www/waptwua` ;
- sur les hôtes Windows dans `C:\wapt\waptserver\repository\waptwua`.

L'URL de téléchargement du dépôt de l'agent de mise à jour Windows WAPT est basée sur le paramètre `repo_url` de `wapt-get.ini` :

Note:

- En cas de dépôt répliqué, il est totalement possible de l'utiliser avec WAPT Windows Update afin de réduire l'utilisation de la bande passante.

Si un proxy est nécessaire pour accéder à Internet, veillez à *définir le serveur proxy dans le fichier waptserver.ini*.

28.2 Différences entre les mises à jour Windows WAPT et WSUS

WSUS va télécharger par défaut les mises à jours des catégories sélectionnées. Cela peut mener à une base de données vraiment conséquente et une forte utilisation de l'espace de stockage.

WAPT Windows Update ne va que télécharger ce dont il a besoin pour au moins un poste client. Cela aide à garder une base de données locale de petite taille (environ une 10aine de Gigaoctets) et il peut facilement se nettoyer si vous avez besoin de récupérer de l'espace.

28.3 Les mises à jours majeures d'OS

Les mises à jour majeures d'OS permettent de passer d'une version d'OS à une autre. Cela inclus par exemple, des mises à jours de Windows 7 vers Windows 10, ou bien de Windows 10 1903 vers Windows 10 20H2.

Les mises à jour de version majeures sont gérées de la même manière que les mises à jour d'OS mineures. Les mises à jour majeures sont gérées via le téléchargement du contenu de l'ISO de la nouvelle version (même contenu qu'une installation de base) puis lance le **`setup.exe`** avec les bons paramètres. Ce Processus est le même que pour WSUS, SCCM et les mises à jour WAPT Windows Updates.

Dans le cas de WAPT Windows Updates, vous aurez besoin de créer un paquet de mise à jour d'OS en utilisant un modèle de paquet fournit sur https://store.wapt.fr.

28.4 Les mises à jours de pilotes

Les mises à jour de pilote via WSUS ne sont pas recommandés puisqu »il est difficile de gérer correctement les effets de bord. Dans le cas de WAPT WindowsUpdates, **LES PILOTES NE SONT PAS TELECHARGES** puisqu'ils ne sont pas référencés dans les fichiers `wsusscn2.cab` fournis par Microsoft.

Il est recommandé de pousser les mises à jour des pilotes via un paquet WAPT personnalisé. Si le correctif du pilote est empaqueté sous forme de `.msu`, vous pouvez l'empaqueter comme un paquet WAPT standard.

Il suffit de sélectionner le fichier `.msu` et de cliquer sur *Générer un modèle de paquet → Modèle de paquet → Paquet Windows Update (.msu)* dans la console WAPT pour lancer l'assistant de création simplifiée de package.

Si la mise à jour du pilote est emballée sous forme de `.zip` contenant le fichier `.exe`, vous pouvez créer un paquet WAPT contenant les fichiers nécessaires et le binaire **`setup.exe`** avec le drapeau silencieux correct.

28.5 Les KB Out of band (Hors bande)

Parfois, Microsoft fournit des mises à jours OOB (Out of Band) qui sont en dehors de l'index du `wsusscn2.cab`. Ces mises à jour ne sont pas inclus dans les mises à jour principales car elles peuvent corriger un problème très spécifique ou peuvent avoir des inconvénients.

Si vous souhaitez déployer une KB de mise à jour OOB, vous pouvez la télécharger depuis le catalogue Microsoft https://www.catalog.update.microsoft.com/Home.aspx.

Il suffit de sélectionner le fichier *`.msu`* et de cliquer sur *Générer un modèle de paquet* dans la console WAPT pour lancer l'assistant de création simplifiée de package.

Pour ce faire, vous pouvez suivre *cette documentation* pour construire des fichiers *`.msu`* pour ces mises à jour *Out-of-band (Hors bande)*.

> **Attention:** Vous devez vous montrer prudent avec les mises à jour OOB car elles peuvent détruire votre système, assurez-vous de lire les pré-requis sur le rapport Microsoft correspondant à la mise à jour et de tester cette dernière méticuleusement.

28.6 Configurer WAPTWUA sur l'agent WAPT

WAPTWUA se configure dans `wapt-get.ini` dans la section `[waptwua]`.

Vous aurez alors plusieurs options:

Table1: Les options de configuration dans la section `[waptwua]` dans le `wapt-get.ini`

Options (Valeurs par défaut)	Description	Exemple
`enabled` (défaut `False`)	Activer ou désactiver WAPTWUA sur cette machine.	enabled = True
`direct_download` (défaut `False`)	Télécharger les mises à jour directement depuis les serveurs Microsoft.	direct_download = True
`default_allow` (défaut `False`)	Configuré si la mise à jour est autorisée ou pas par défaut.	default_allow = True
`download_scheduling` (défaut `None`)	Configure la récurrence des scans des Windows Update (ne fera rien s'il y a un paquet de règles *waptwua* ou que le fichier `wsusscn2.cab` n'a pas changé).	download_scheduling = 1d
`install_scheduling` (défaut `None`)	Configure la récurrence des installations Windows Update (ne fera rien s'il n'y a aucune mise à jour en attente).	install_scheduling = 2h
`install_at_shutdown` (défaut `False`)	Définit si les mises à jour sont déclenchées lors de l'arrêt de l'hôte.	install_at_shutdown = True
`install_delay` (défaut `None`)	Configure un délai d'installation entre la publication dans le dépôt et l'installation.	install_delay = 15d
`allowed_severities` (défaut `None`)	Définit une liste de criticité qui sera automatiquement accépté durant un scan WAPT windows update. ex: *Important*, *Critical*, *Moderate*.	allowed_severities = Important
`include_potentially_superseded_updates` (défaut `False`)	Définit si l'agent de mise à jour de Windows affichera à la fois les dernières KB et les KB consolidées (True), ou seulement les dernières KB (False).	include_potentially_superseded_u = True

Indication: Ces options peuvent être configurées lors de la génération de l'agent.

Exemple de section `[waptwua]` dans le fichier `wapt-get.ini`:

```
[waptwua]
enabled = True
default_allow = False
direct_download = False
download_scheduling = 7d
install_at_shutdown = True
install_scheduling = 12h
install_delay = 3d
```

Lorsque vous créez le `waptagent.exe` depuis votre console, ces options correspondent à cela:

Exemple de code source pour un paquet qui modifie les paramètres `[waptwua]`:

```
def install():

  inifile_writestring(WAPT.config_filename,'waptwua','enabled','true')
  inifile_writestring(WAPT.config_filename,'waptwua','install_at_shutdown','true')
  inifile_writestring(WAPT.config_filename,'waptwua','download_scheduling','7d')
  inifile_writestring(WAPT.config_filename,'waptwua','allowed_severities','Critical,Important')

  print('Reload WAPT configuration')
```

(suite sur la page suivante)

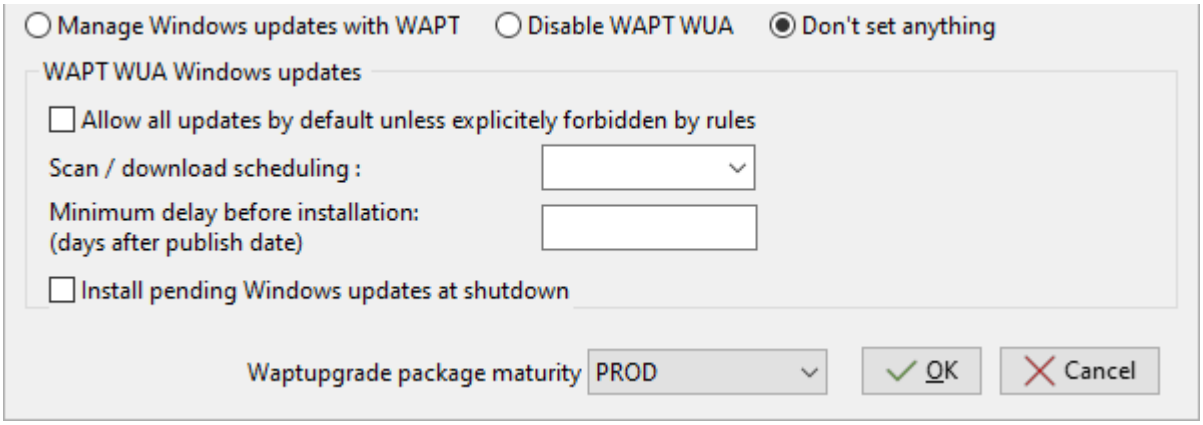


Figure2: Options de menu pour l'agent de mise à jour Windows WAPT

(suite de la page précédente)

```
WAPT.reload_config_if_updated()
```

28.7 Utiliser WAPTWUA depuis la console

Les WAPTWUA sont gérées avec deux onglets dans la console WAPT.

Sous-onglet règles WUA dans l'onglet Dépôt privé WAPT

L'onglet *Paquet WAPTWUA* vous permet de créer des paquet de règles *waptwua*.

- Lorsque ce type de paquet est instalé sur une machine, il indique à l'agent WAPTWUA les KBs (Knowledge Base articles) autorisées ou interdites.
- Lorsque plusieurs paquets de règles *waptwua* sont installés sur la machine, les différentes règlesvont fusionner.
- Lorsqu'une `cab` n'est ni mentionnée comme autorisée ni mentionnée comme interdite, les agents WAPT vont alors prendre la valeur du `default_allow` dans `wapt-get.ini`.

Note:

- Si la configuration de l'agent WAPTWUA est définit à `default_allow = True`, alors il sera nécessaire de spécifier les `cab` interdites.
- Si la configuration de l'agetn WAPTWUA est définit à `default_allow = False` , il sera nécessaire de spécifier les `cab` autorisées.

Indication:

- Pour tester les mises à jour sur un petut groupe de postes, vous pouvez configurer la valeur par défaut de WAPTWUA avec `maturity = PREPROD`.
- Vous pouvez tester les mises à jour sur un petit groupe de postes et si tout se passe bien, vous pouvez lancer les mises à jour à votre flotte complète de postes.

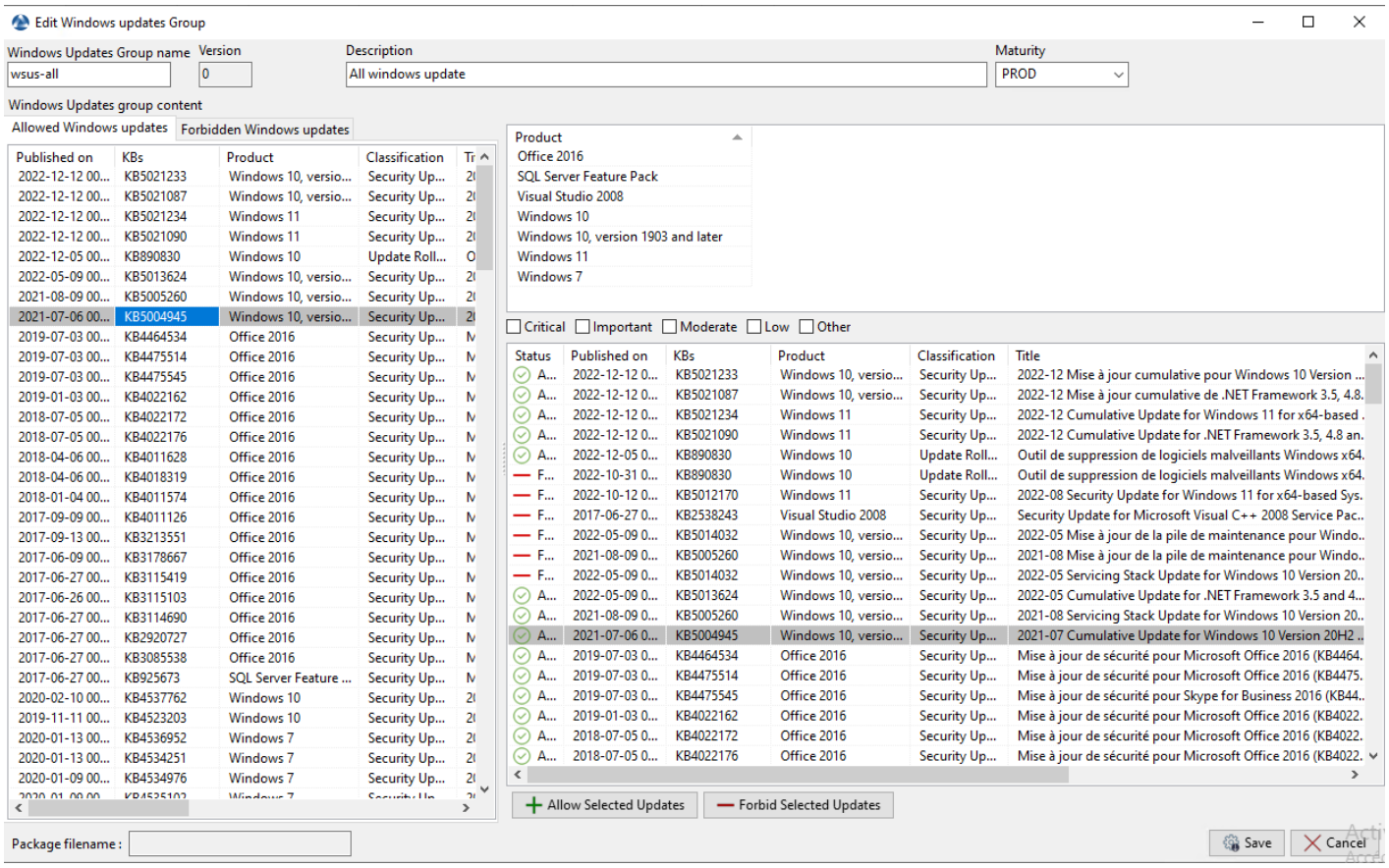


Figure3: Création de paquets de regroupement de profils dans la console WAPT

L'onglet liste des Windows Updates

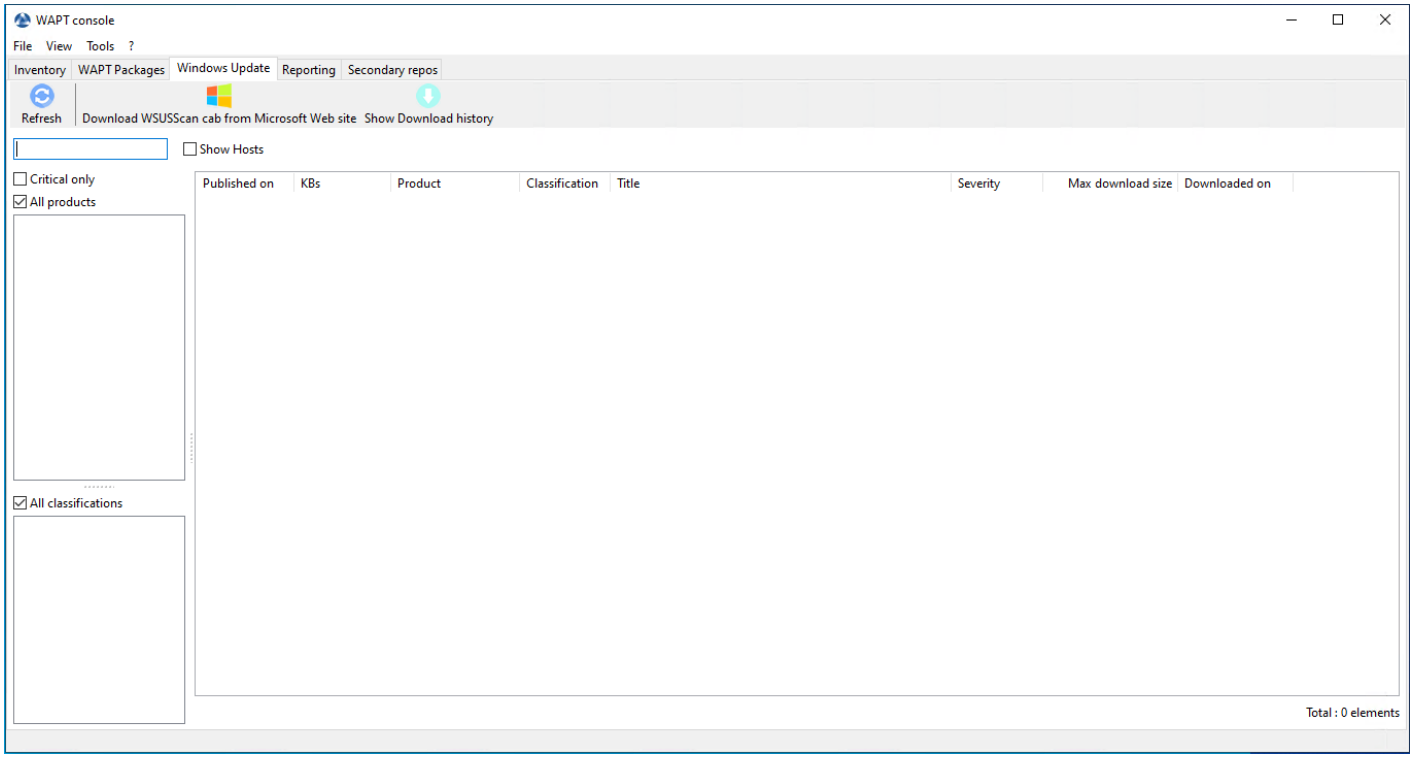


Figure4: Les mises à jour Windows en attente affichées dans la console WAPT

L'onglet *Liste Windows Update* liste toutes les Mises àjour Windows demandées.

Important: Le serveur WAPT ne scanne pas le `wsussc2.cab` lui-même, il laisse l'utilitaire Windows Update Agent présent sur tous les hôtes Windows le faire. Si une mise à jour semble manquer dans la liste, vous **DEVEZ** exécuter un scan sur l'un des hôtes présents dans la console WAPT. Si vous exécutez un scan WUA WAPT sur un client Windows 10, les fichiers CAB et Windows 10 seront affichés dans l'onglet *Windows Update*.

Le panneau de gauche affiche les catégories des mises à jour, vous permettant de filtrer par:

- criticité ;
- produit ;
- classification.

Dans le panneau de droite, si la colonne *Téléchargée le* est vide, cela signife que les mises à jour n'ont pas encore été téléchargées par le serveur WAPT et n'est pas présent sur le serveur WAPT (Cette mise à jour n'est pas manquante sur les postes).

- Vous pouvez forcer le téléchargement de la mise à jour en faisant *clic-droit → Télécharger les mises à jour séelctionnées*.
- Vous pouvez aussi forcer le téléchargement du fichier `wsusscn2.cab` avec le bouton *Télécharger le cab WSUSScan depuis le site de Microsoft*.
- Vous pouvez voir le téléchargement des mises à jour Windows sur le serveur avec le bouton *Afficher la tâche de téléchargement* .

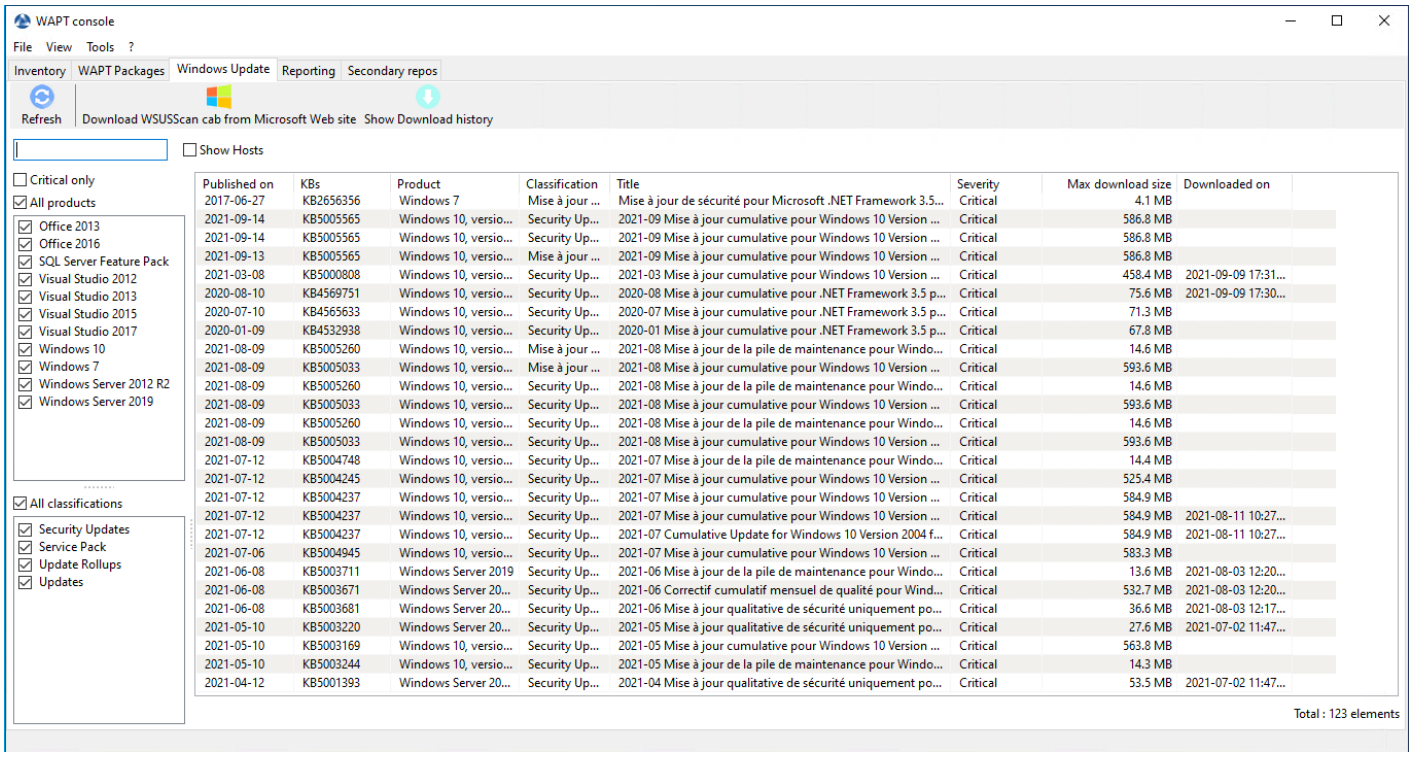


Figure5: Les mises à jour Windows en attente affichées dans la console WAPT

Indication: Toutes les 30 minutes, le serveur WAPT va chercher les mises à jour qui ont été demandées au moins une fois par les client WAPT et qui n'ont pas été téléchargées en mises en cache. » Si une mise à jour est en attente, le serveur WAPT var le télécharger depuis les sites officiels de Microfoft.

Vous pouvez forcer ce scan avec le bouton *Télécharger le cab WSUSScan depuis le site de Microsoft*; dans l'onglet *Mises à jour Windows → Liste Windows Updates*

28.7.1 Nettoyer des vieilles Windows Update

Vous pouvez exécuter le nettoyage manuellement ou automatiquement.

Automatiquement

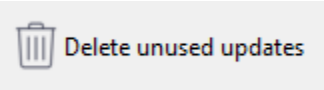
Si la KB n'est pas installée sur la machine, elle est automatiquement supprimée sur le Serveur WAPT entre 2h30 et 3h30 du matin chaque jour. Il est possible de désactiver la suppression automatique des KB avec l'option `cleanup_kbs` dans le fichier `waptserver.ini` de configuration du Serveur WAPT.

Ajouter le paramètre dans *le fichier de configuration du serveur* :

```
cleanup_kbs = False
```

Utiliser WAPTWUA depuis la console

Pour nettoyer votre dossier `waptwua`, vous pouvez aller dans l'onglet *Mises à jour Windows* et cliquer sur *Effacer les mises à jour non-affectées*. Cela supprimera toutes les KB inutiles stocké sur le Serveur WAPT.



Redémarrez le serveur WAPT

Il est possible de supprimer manuellement du serveur WAPT tout fichier Windows Update qui n'est plus nécessaire.

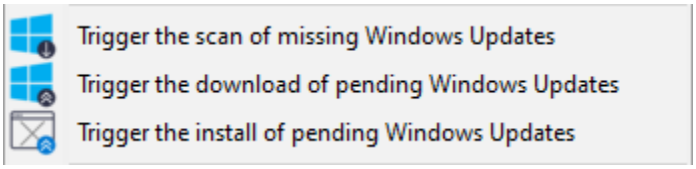
Le serveur WAPT va seulement re-télécharger les mises à jour supprimées si un des hôtes équipés le demande.

Sut le serveur WAPT, les mises à jour téléchargées sont sotckées:

- Sous Linux dans `/var/www/waptwua`.
- Sur les hôtes Windows dans `C:\wapt\waptserver\repository\waptwua`.

28.7.2 Lancer WUA sur les clients

Depuis la console vous avez trois options.



- Le bouton *Lancer la recherche de mises à jour* va lancer le scan sur le client et va lister toutes les mises à jour marquées pour l'OS.
- Le bouton *Lancer le téléchargement des Mises à jour Windows en attente* va lancer le téléchargement des mises à jour en attente sur le client.
- Le bouton *Lancer l'installation des mises à jour Windows en attente* va lancer l'installation des mise à jour téléchargées sur le client.

Indication: Lorsque des mises à jour en attente sont sotckées en cache pour être installer, l'agent WAPT va déclencher le service WUA.

L'agent WAPT va activer et démarrer le service WUA temporairement pour installer les mises à jour. Lorsque les mises à jour sont installées, waptservice va couper et désactiver le service WUA jusqu'au prochain cycle.

28.7.3 Etat des mises à jour sur l'hôte

Les mises à jour Windows peuvent avoir 4 états sur un poste.

Statut	Description
OK	Une mise à jour Windows qui s'est correctement installé.
MISSING	Une mise à jour Windows qui n'a pas encore été téléchargé sur le serveur WAPT.
PENDING	Le serveur WAPT sait qu'il doit télécharger une mise à jour depuis les serveurs officiels de Microsoft.
DISCARDED	Une mise à jour Windows interdites par des règles.

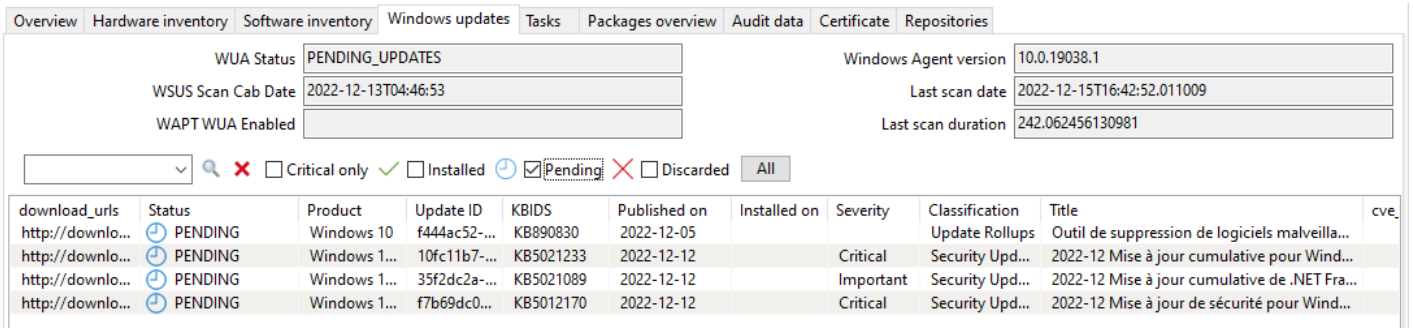


Figure6: Les mises à jour Windows en attente affichées dans la console WAPT

28.7.4 Notion d'UpdateID

Dans WAPT nous n'utilisons pas les kbids mais les **updateids**.

Cela nous permet d'être plus fin dans la gestion des mises à jour.

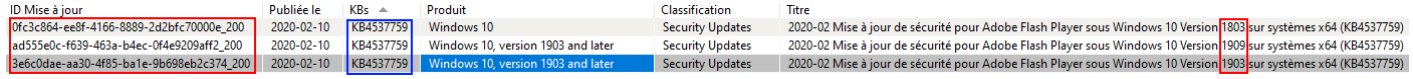


Figure7: Liste montrant les KB en double dans la console WAPT

Dans cet exemple, la KB4537759 apparaît de multiples fois car il y a 3 différents updateids:

- win10 1803;
- win10 1903;
- win10 1909;

Vous devriez également autoriser une mise à jour et non pas une *kb ids*.

28.8 WAPT ne force pas Windows à désinstaller une Windows Update

Attention: <La désinstallation d'une mise à jour de Windows peut être dangereuse pour l'hôte**.

Désinstaller une mise à jour Windows peut être dangereux pour la machine. Quand une mise à jour est détectée comme interdite par WAPT, sa désinstallation ne sera **PAS** forcée.

Si vous voulez vraiment désinstaller une mise à jour, vous devriez créer un paquet pour désinstaller la KB.

Voici un exemple:

```
from setuphelpers import *

uninstallkey = []

def install():
```

(suite sur la page suivante)

(suite de la page précédente)

```
  with EnsureWUAServRunning():
    run('wusa /uninstall /KB:4023057')
```

CHAPTER 29

Synchroniser les inventaires de WAPT vers GLPI

29.1 Principe de fonctionnement

WAPT Enterprise propose une synchronisation entre les inventaires de vos postes et GLPI ITSM Software.

Cette méthode synchronise automatiquement les changements sur votre infrastructure informatique avec le serveur GLPI.

Note: WAPT peut se synchroniser avec GLPI 10 en utilisant l'API JSON native.

WAPT peut se synchroniser avec GLPI version 9.x en utilisant le plugin **FusionInventory** au format XML.

Attention: GLPI par WAPT ne fonctionne pas avec l'authentification Kerberos pour GLPI.

Si vous utilisez Kerberos pour GLPI, excluez `glpi/plugins/fusioninventory/` de l'authentification NGINX.

Attention: Le mode d'installation a changé depuis WAPT 2.2.2. La synchronisation passe maintenant par un packaging WAPT et n'est plus directement intégrée dans le serveur WAPT. L'interface graphique de configuration de GLPI dans la WAPTConsole a été supprimée.

29.2 Installer les dépendances requises

Pour pouvoir recevoir les inventaires sur votre seveur GLPI, vous aurez besoin du plugin **FusionInventory** sur votre serveur GLPI.

Note: Vous pouvez suivre ce guide pour installer FusionInventory.

Après avoir installé FusionInventory, vous aurez un **point d'accès** sur votre serveur WAPT pour envoyer les inventaires vers :

```
http:/glpi.mydomain.lan/glpi/plugins/fusioninventory/
```

29.3 Configuration de WAPTAgent et du packaging de synchronisation

Installez et configurez l'agent WAPT sur l'ordinateur qui exécutera la synchronisation. L'agent WAPTAgent est installé par défaut sur le WAPTServer, il suffit de le configurer.

Pour configurer le WAPTAgent, veuillez vous référer à la documentation correspondante.

Ensuite, vous devez installer le packaging de synchronisation de GLPI :

- pour GLPI 9.x, vous devez installer le packaging *tis-glpi-plugin-export-to-glpi9*
- pour GLPI 10.x, vous devez installer le packaging *tis-glpi-plugin-export-to-glpi10*

Vous devez configurer un programme d'audit sur l'agent

```
[global]
...
waptaudit_task_period=120m
...
```

Dans le répertoire $WAPT_HOME/private, modifier les fichiers glpi.ini et wapt_api.ini

```
[glpi]
username = glpi
password = xxxxxxx
url = https://glpi.xx.xxxxx.xx/plugins/fusioninventory/
```

```
[wapt]
username = waptregister
password = waptregister2021!
url = https://srvwapt.ad.tranquil.it
```

Pour tester la configuration actuelle, vous pouvez déclencher un audit

```
wapt-get audit tis-glpi-plugin-export-to-glpi9
# or
wapt-get audit tis-glpi-plugin-export-to-glpi10
```

29.4 Éléments actuels envoyés par WAPT au serveur GLPI

Table1: Description des éléments

Valeur	Envoyé	Non Envoyé
Nom de l'ordinateur	✅	
Nom d'utilisateur	✅	
Description	✅	
Nom de l'OS	✅	
Version de l'OS	✅	
Langue	✅	
CPU	✅	
Mémoire	✅	
Batterie	✅	
Type de chassis	✅	
Physique ou virtuel	✅	
Configuration de la carte réseau	✅	
Liste d'imprimante et les propriétés	✅	
Logiciel installé[1]	✅	
Lecteur réseau	✅	
Variables d'environnement[2]	✅	
Modèles des écrans	✅	
Modèle de la souris et du clavier		❌
Modèles des cartes contrôleurs (excepté la carte graphique)		❌
Version de l'antivirus		❌
État du parefeu		❌
Liste groupe local		❌
Liste et état de la banque de mémoire		❌
Liste des ports UBS et des périphériques connectés		❌
Statut de l'imprimante		❌
Lecteurs de carte		❌
Liste d'appx à l'échelle du système		❌

29.5 Erreurs possibles dans l'inventaire rapporté sur le serveur GLPI

Les inventaires téléchargés par le serveur WAPT vers le serveur GLPI peuvent être incomplets ou comporter des erreurs par rapport aux inventaires téléchargés directement par l'agent FusionInventory déployé sur les hôtes. L'une des raisons est que WAPT vise à ne rapporter que les valeurs les plus importantes.

Si vous pensez que des éléments importants manquent ou sont signalés de manière erronée, veuillez signaler le problème à l'équipe de développement de Tranquil IT.

Pour le rapport, l'équipe de développement a besoin de 2 fichiers `.xml`.

1. Tout d'abord, installez l'agent FusionInventory sur l'ordinateur sur lequel vous observez un élément d'inventaire manquant ou déclaré à tort.

[1] Sans compter l'installation des Appx à l'échelle du système
[2] Actuellement, les variables d'environnement du système et de l'utilisateur sont incluses.

2. Exécutez l'agent FusionInventory et extrayez le rapport dans un fichier `.xml`.

Windows

```
"C:\Program Files\FusionInventory-Agent\fusioninventory-inventory" > %TEMP%\inventory.xml
```

Linux

```
fusioninventory-inventory > /tmp/inventory.xml
```

MAC

```
fusioninventory-inventory > /tmp/inventory.xml
```

1. Définissez le répertoire de débogage dans le fichier *waptserver.ini*.

```
glpi_inventory_debug_directory = /tmp/glpi
```

4. Redémarrez le serveur WAPT
5. Récupérer le fichier `/tmp/glpi/UUID.xml` du serveur WAPT, l'UUID étant l'identifiant de l'hôte.
6. Envoyez les 2 fichiers à l'équipe de développement.

CHAPTER 30

Utiliser la fonctions de requête dans WAPT

30.1 Principe de fonctionnement

https://youtu.be/UjBfelmJyKo

WAPT **Enterprise** offre des fonctionnalités de reporting avancées.

En effet, qui mieux que vous pouvez savoir ce dont vous avez besoin dans votre rapport.

Avec WAPT nous vous proposons d'écrire vos requêtes SQL dont le résultat s'affichera dans la console WAPT.

Le diagramme de la structure de la base de données est disponible ici `wapt_db_data_structure.svg`.

30.2 Concepteur de requêtes WAPT

Le concepteur de requêtes vous offre la possibilité de modifier vos propres requetes sur la base de données PostgreSQL de WAPT.

Note: La base de données PostgreSQL est définie en mode **Lecture seule**, de sorte que les requêtes exécutées à partir du Report Designer qui tentent de mettre à jour, de supprimer ou d'insérer des données échouent.

Vous pouvez importer des requêtes SQL depuis le dépôt Tranquil IT en cliquant sur *Reporting → Importer des requêtes… → Depuis une Url* ou sur *Reporting → Importer des requêtes… → Depuis un fichier*.

Si vous choisissez d'importer une requête depuis le dépôt Tranquil IT, sélectionnez une ou plusieurs requêtes, puis cliquez sur *Sauvegarder les requêtes sélectionnées*. Les nouvelles requêtes apparaîtront dans la Console WAPT.

Si vous importez une requête depuis un fichier, sélectionnez votre requête dans l'explorateur de fichiers. La requête peut être un fichier au format `.query` ou `.json`.

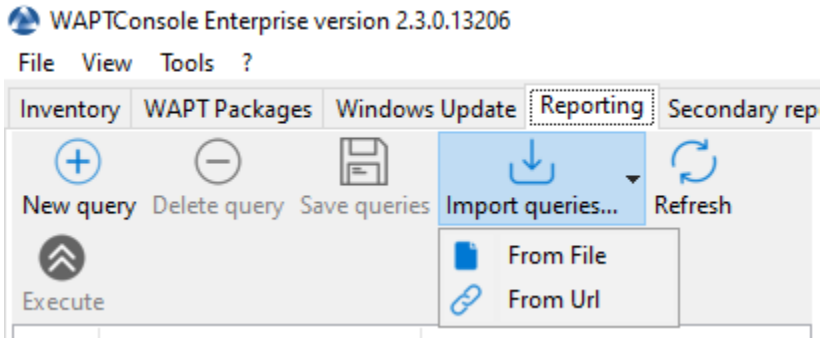


Figure1: Importer une requête SQL dans la Console WAPT

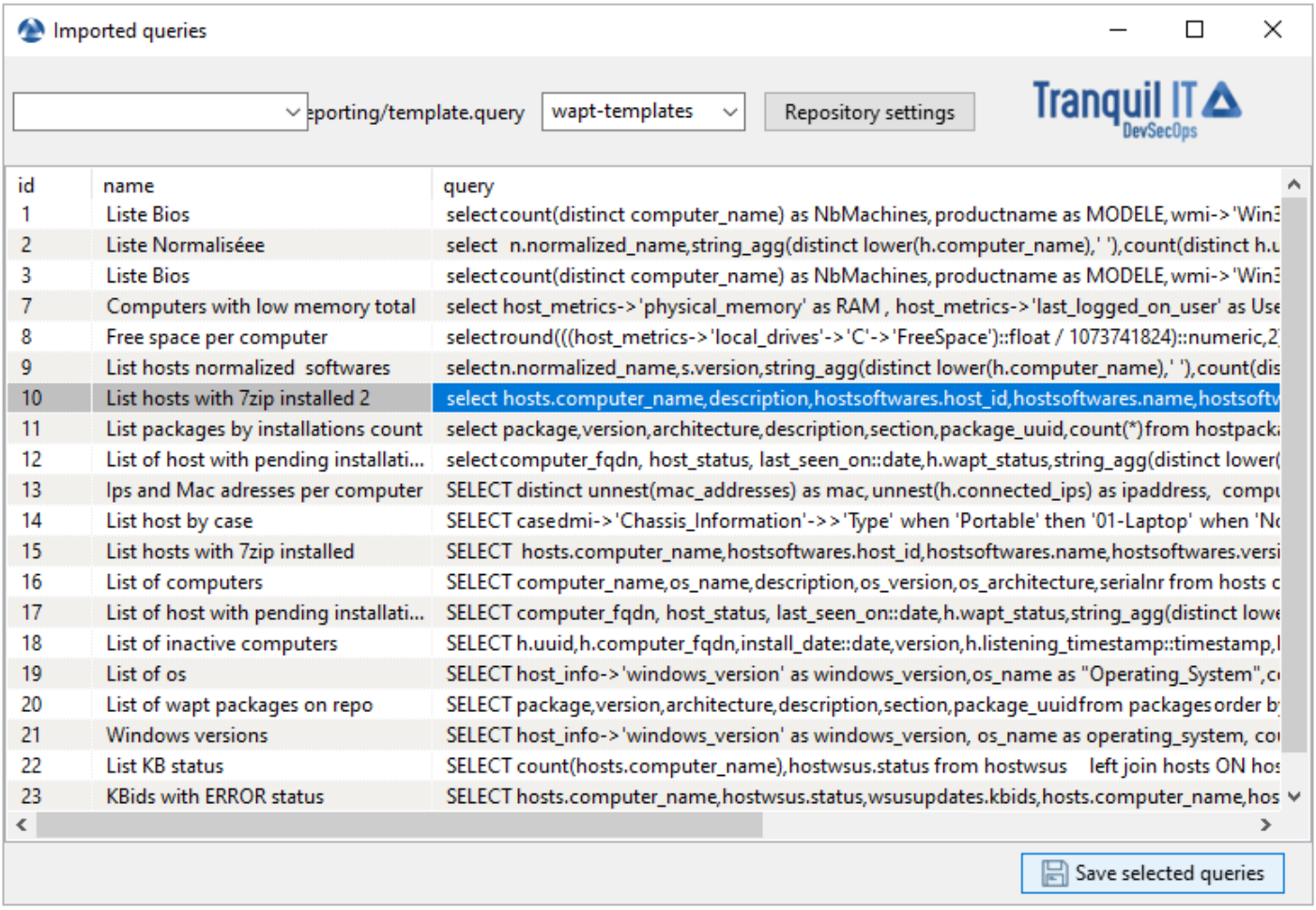


Figure2: Importer une requête SQL dans la Console WAPT depuis une URL

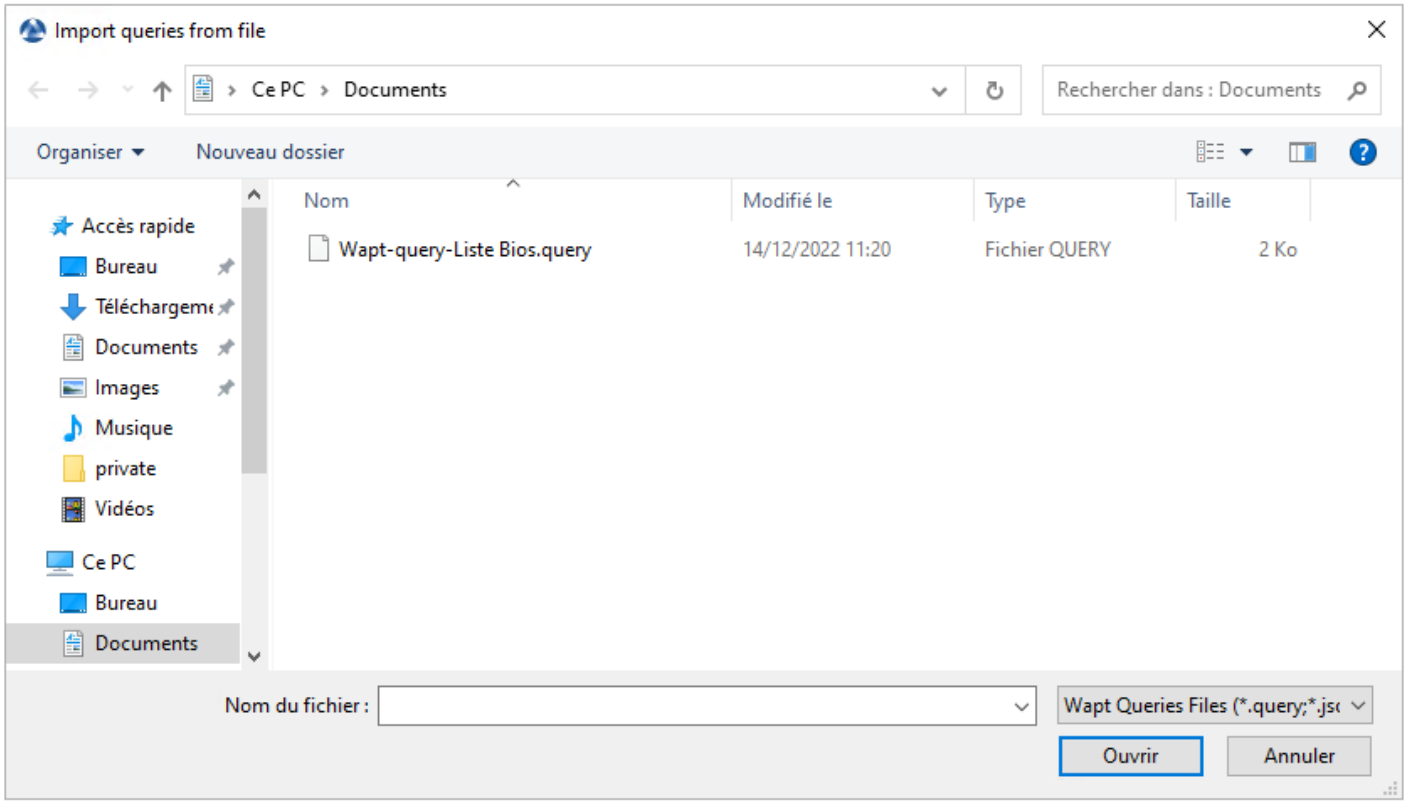


Figure3: Importer une requête SQL dans la Console WAPT à partir d'un fichier

Note: Quelle que soit la méthode que vous utilisez pour importer des requêtes, n'oubliez pas de cliquer sur le bouton *Sauvegarder* pour sauvegarder les requêtes importées.

Pour créer une nouvelle requête, cliquer sur *Reporting* → *Mode conception* → *Nouvelle requête*.

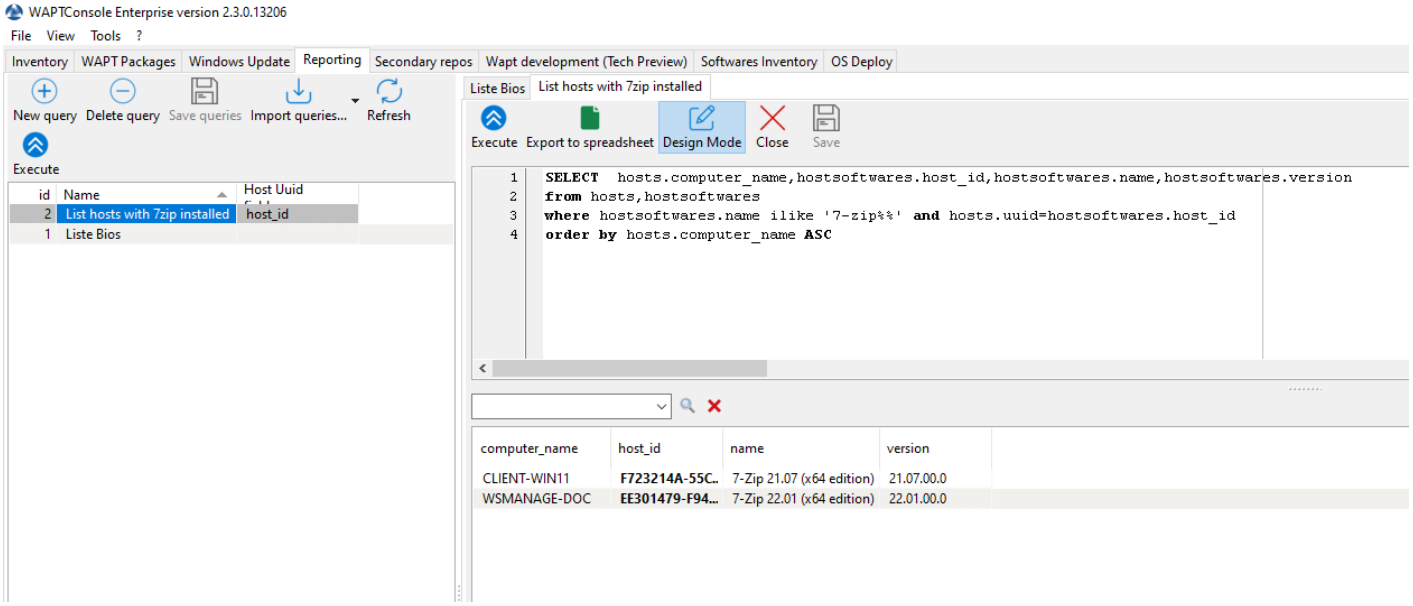


Figure4: Conception d'un rapport de requête SQL dans la Console WAPT

Indication:

- Pour renommer une requête, appuyer sur la touche `F2`.
- Dans l'encadré du haut, vous pouvez écrire votre requête SQL.

Pour éditer / modifier / Sauvegarder vos requêtes:

- Le bouton *Recharger* est utilisé pour recharger les requêtes sauvegardées sur le serveur, par exemple, si un collègue vient juste d'éditer une nouvelle requête.
- Le bouton *Nouvelle requête* va ajouter une requête vide à la liste.
- Le bouton *Supprimer la requête* va supprimer la requête sélectionnée sur le serveur WAPT.
- Le bouton *Enregistrer tout* va sauvegarder votre requête au serveur WAPT.
- Le bouton *Exécuter* va exécuter la requête sélectionnée.
- Le bouton *Exporter vers tableur* va exporter le résultat de votre requête dans un feuille de calcul.
- Le bouton *Dupliquer* va dupliquer une requête existante pour éviter de repartir d'une requête vide.

Vous disposez de plusieurs options lorsque vous cliquez avec le bouton droit de la souris sur une requête.

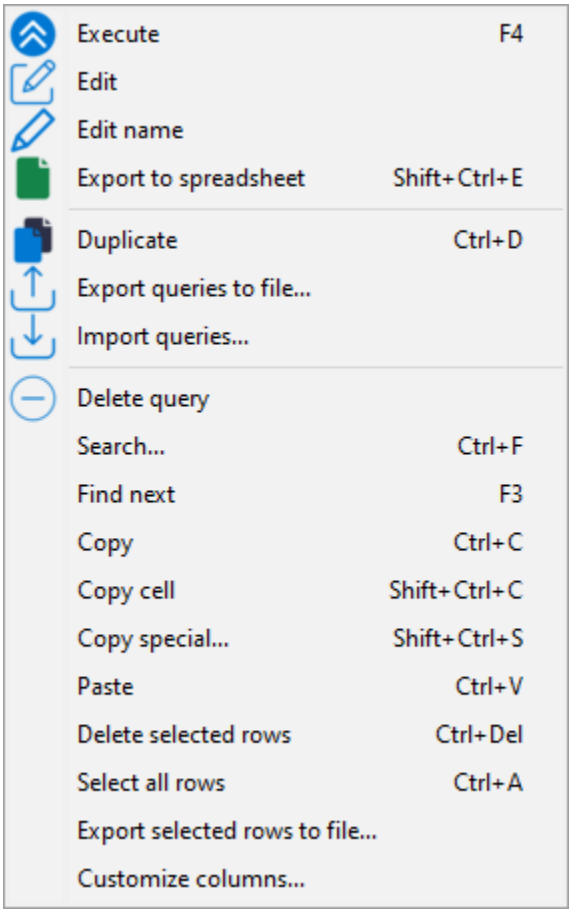


Figure5: Options disponibles pour un rapport de requête SQL dans la Console WAPT

Table1: Liste des actions disponibles sur une sélection de requête dans la Console WAPT

Nom	Description
Executer	Exécute la requête SQL.
Éditer	Modifie la requête SQL.
Éditer le nom	Modifie le nom de la requête SQL.
Exporter vers un tableur	Exporte le résultat de la requête SQL vers un fichier formaté en *csv*.
Dupliquer	Duplique la requête SQL.
Exporter les requêtes vers un fichier...	Exporte les requêtes SQL sélectionnées vers un fichier. Cette méthode permet de partager ou de sauvegarder les requêtes SQL.
Importer des requêtes...	Importe des requêtes à partir d'un fichier.
Supprimer des requêtes	Supprime les requêtes sélectionnées.

Note:

- Les requêtes sont sauvegardées dans la base de données PostgreSQL WAPT.
- Le raccourci `CTRL+espace` vous permet de construire votre requête de façon plus efficace.

30.3 Utiliser des requêtes comme filtre dans l'onglet inventaire

Vous pouvez créer un filtre à utiliser dans l'onglet *Inventaire* basé sur une requête SQL. Dans cet exemple, nous utilisons une requête qui liste les machines sur lesquelles le logiciel 7zip a été installé.

Indication: Cette méthode est puissante car elle permet de rechercher des titres de logiciels qui n'ont pas été installés à l'aide de WAPT.

Sur le résultat de la requête, faites un clic droit sur *host_id*, qui est un identifiant unique, sélectionner cette entrée et cliquer sur le bouton *Choisir cet identifiant unique*.

Lorsque vous avez terminé, enregistrer la requête en appuyant sur le bouton *Sauvegarder* puis aller dans l'onglet *Inventaire*.

Ensuite, activer le panneau *Recherche avancée* et sélectionner la requête dans le champ déroulant *Filtrer les machines basé sur la requête SQL*.

Vous verrez alors une liste de machines basée sur la requête sélectionnée.

30.4 Exemple de requêtes

30.4.1 Requêtes Ordinateur

Counting hosts

```
select count(*) as "number_of_hosts" from hosts
```

Listing computers

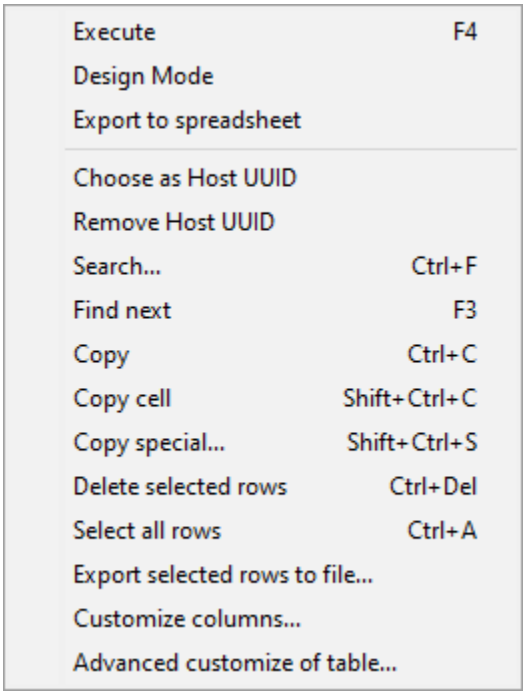


Figure6: Filtrer sur l'UUID de la machine pour créer une vue dynamique dans l'onglet d'inventaire

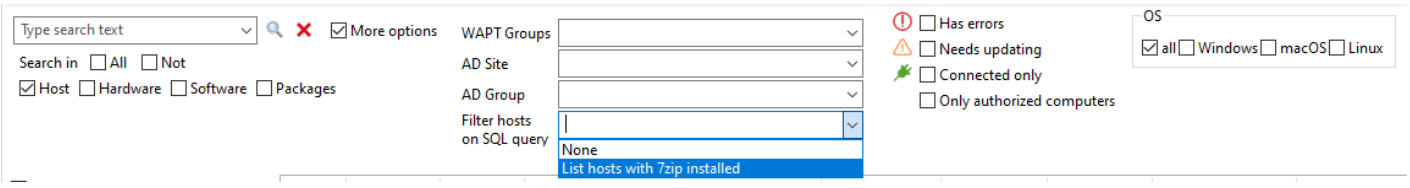


Figure7: Filtrer l'inventaire des machines à l'aide d'une requête SQL

```
select
computer_name,
os_name,
os_version,
os_architecture,
serialnr
from hosts
order by 4,3,1
```

Listing computers MAC addresses and IP

```
select distinct unnest(mac_addresses) as mac,
unnest(h.connected_ips) as ipaddress,
computer_fqdn,h.description,
h.manufacturer||' '||h.productname as model,
h.serialnr,
h.computer_type
from hosts h
order by 1,2,3
```

Listing Windows versions

```
select
host_info->'windows_version' as windows_version,
os_name as operating_system,
count(os_name) as nb_hosts
from hosts
group by 1,2
```

Listing operating systems

```
select host_info->'windows_version' as windows_version,
os_name as "Operating_System",
count(os_name) as "number_of_hosts"
from hosts
group by 1,2
```

Listing hosts not seen in a while

```
select
h.uuid,
h.computer_fqdn,
install_date::date,
version,
h.last_seen_on::timestamp,
h.connected_users from hostsoftwares s
left join hosts h on h.uuid=s.host_id
where s.key='WAPT_is1'
and h.last_seen_on<'20190115'
```

Filtering hosts by chassis types

```
select case
dmi->'Chassis_Information'->>'Type'
when 'Portable' then '01-Laptop'
when 'Notebook' then '01-Laptop'
when 'Laptop' then '01-Laptop'
when 'Desktop' then '02-Desktop'
when 'Tower' then '02-Desktop'
when 'Mini Tower' then '02-Desktop'
else '99-'||(dmi->'Chassis_Information'->>'Type')
end as type_chassis,
string_agg(distinct coalesce(manufacturer,'?') ||' '|| coalesce(productname,''),', '),
count(*) as "number_of_hosts" from hosts
group by 1
```

Listing of hosts with their Windows Serial Key

```
select
computer_name,
os_name,
os_version,
host_info->'windows_product_infos'->'product_key' as windows_product_key
from hosts
order by 3,1
```

30.4.2 Requête WAPT

Listing WAPT packages in the WAPT Server repository

```
select
package,
version,
architecture,
description,
section,
package_uuid,
count(*)
from packages
group by 1,2,3,4,5,6
```

Listing hosts needing upgrade

```
select
computer_fqdn,
host_status,
last_seen_on::date,
h.wapt_status,
string_agg(distinct lower(s.package),' ')
from hosts h
left join hostpackagesstatus s on s.host_id=h.uuid and s.install_status != 'OK'
where (last_seen_on::date > (current_timestamp - interval '1 week')::date
```

(suite sur la page suivante)

(suite de la page précédente)

```
and host_status!='OK')
group by 1,2,3,4
```

30.4.3 Requête Paquets

Listing packages with their number of installation

```
select
package,
version,
architecture,
description,
section,
package_uuid,
count(*)
from hostpackagesstatus s
where section not in ('host','unit','group')
group by 1,2,3,4,5,6
```

30.4.4 Requête logiciel

Listing WAPT Discovery Agents

```
select
h.uuid,
h.computer_name,
install_date::date,
version,
h.listening_timestamp::timestamp,
name
from hostsoftwares s
left join hosts h on h.uuid=s.host_id
where
s.key='WAPT_is1'
and (name ilike 'WAPT%%Discovery%%' or name ilike 'WAPT %%')
```

Listing hosts with their 7zip version associated

```
select
hosts.computer_name,
hostsoftwares.host_id,
hostsoftwares.name,
hostsoftwares.version
from hosts, hostsoftwares
where hostsoftwares.name ilike '7-zip%%'
and hosts.uuid=hostsoftwares.host_id
order by hosts.computer_name asc
```

Listing hosts with their software

```
select
n.normalized_name,
s.version,string_agg(distinct lower(h.computer_name),' '),
count(distinct h.uuid)
from hostsoftwares s
left join normalization n on (n.original_name = s.name) and (n.key = s.key)
left join hosts h on h.uuid = s.host_id
where (n.normalized_name is not null)
and (n.normalized_name<>'')
and not n.windows_update
and not n.banned
and (last_seen_on::date > (current_timestamp - interval '3 week')::date)
group by 1,2
```

Listing normalized software

```
select
n.normalized_name,
string_agg(distinct lower(h.computer_name),' '),
count(distinct h.uuid)
from hostsoftwares s
left join normalization n on (n.original_name = s.name) and (n.key = s.key)
left join hosts h on h.uuid = s.host_id
where (n.normalized_name is not null)
and (n.normalized_name<>'')
and not n.windows_update
and not n.banned
and (last_seen_on::date > (current_timestamp - interval '3 week')::date)
group by 1
```

Vous pouvez aussi trouver plus d'exemple de requêtes sur le Forum Tranquil IT.

N'hésitez pas à partager vos requêtes sur le même forum avec une explication de ce que fais votre requête, idéalement avec une capture d'écran ou une table affichant un échantillon du résultat de votre requête.

30.5 Normaliser les noms de logiciels

Parfois, la version du logiciel ou son architecture fait partie intégrante du nom du logiciel. Quand le logiciel s'enregistre dans l'inventaire du serveur WAPT, il apparaît en différents logiciels alors qu'ils sont pareils pour nous humains.

Pour résoudre ce problème, le nom des logiciels peut être normalisé dans WAPT. Aller dans l'onglet *Inventaire des logiciels*.

- Cliquez sur *Normaliser les nom de logiciels* dans le menu *Outils*.
- Sélectionnez le logiciel à standardiser, par exemple, toutes les version différentes d'Adobe Flash player.
- Sur la colonne *Normalisé*, appuyez sur `F2` pour assigner un nom standard sur le logiciel sélectionné. Puis appuyez sur `Entrée`.

Note:

- Pour sélectionner plusieurs programmes, sélectionnez les avec les combinaisons de touches `shift-up/down`.

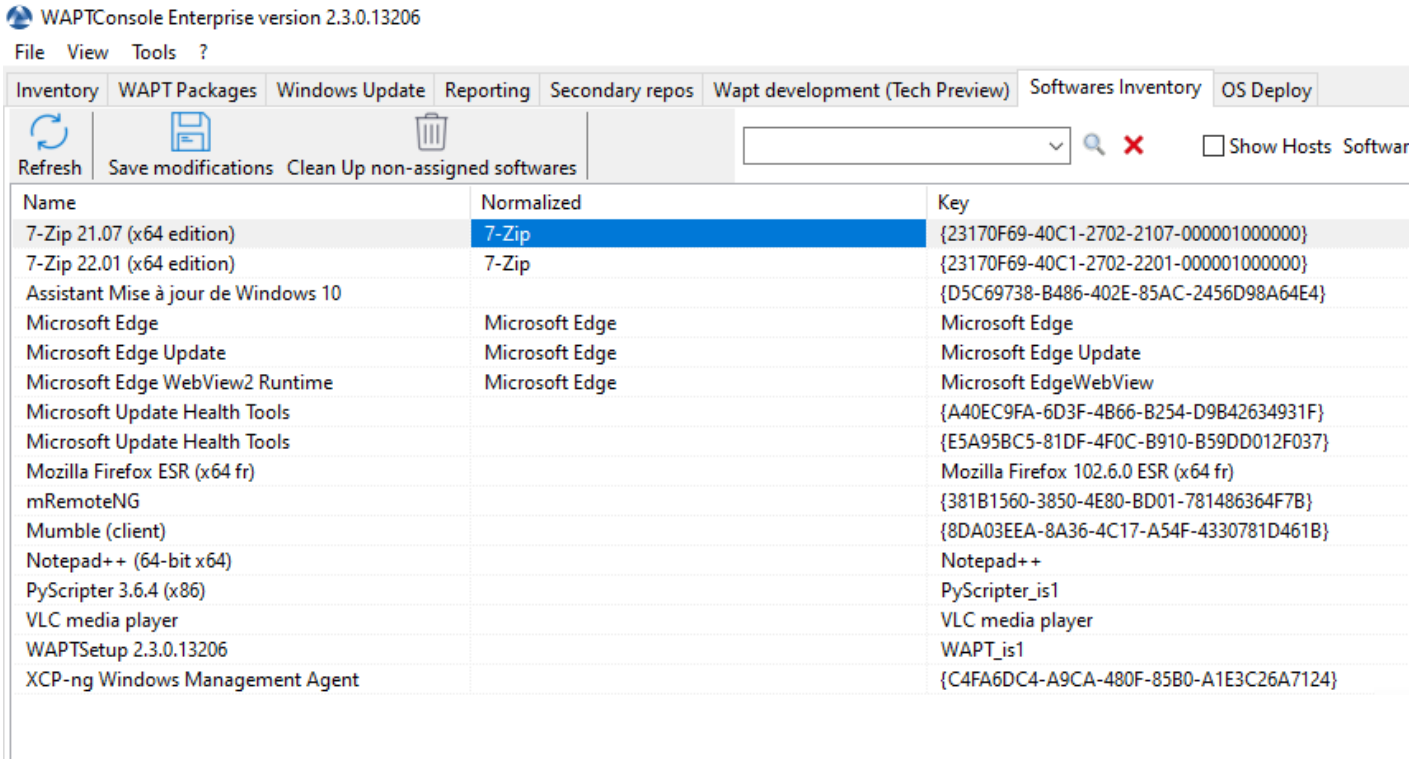


Figure8: Normaliser le nom du logiciel

- Vous pouvez aussi marquer un logiciel comme *Mise à jour Windows* ou *Banni* (Appuyez sur `la barre espace` dans la colonne correspondante).

- Appuyer sur le bouton *Sauvegarder* pour charger les changements sur le Serveur WAPT.

Vous pouvez maintenant lancer vos requêtes avec ce nom standardisé.

Indication: La case à cocher *Afficher les machines* permet de voir les titres des logiciels qui sont installés sur les machines.

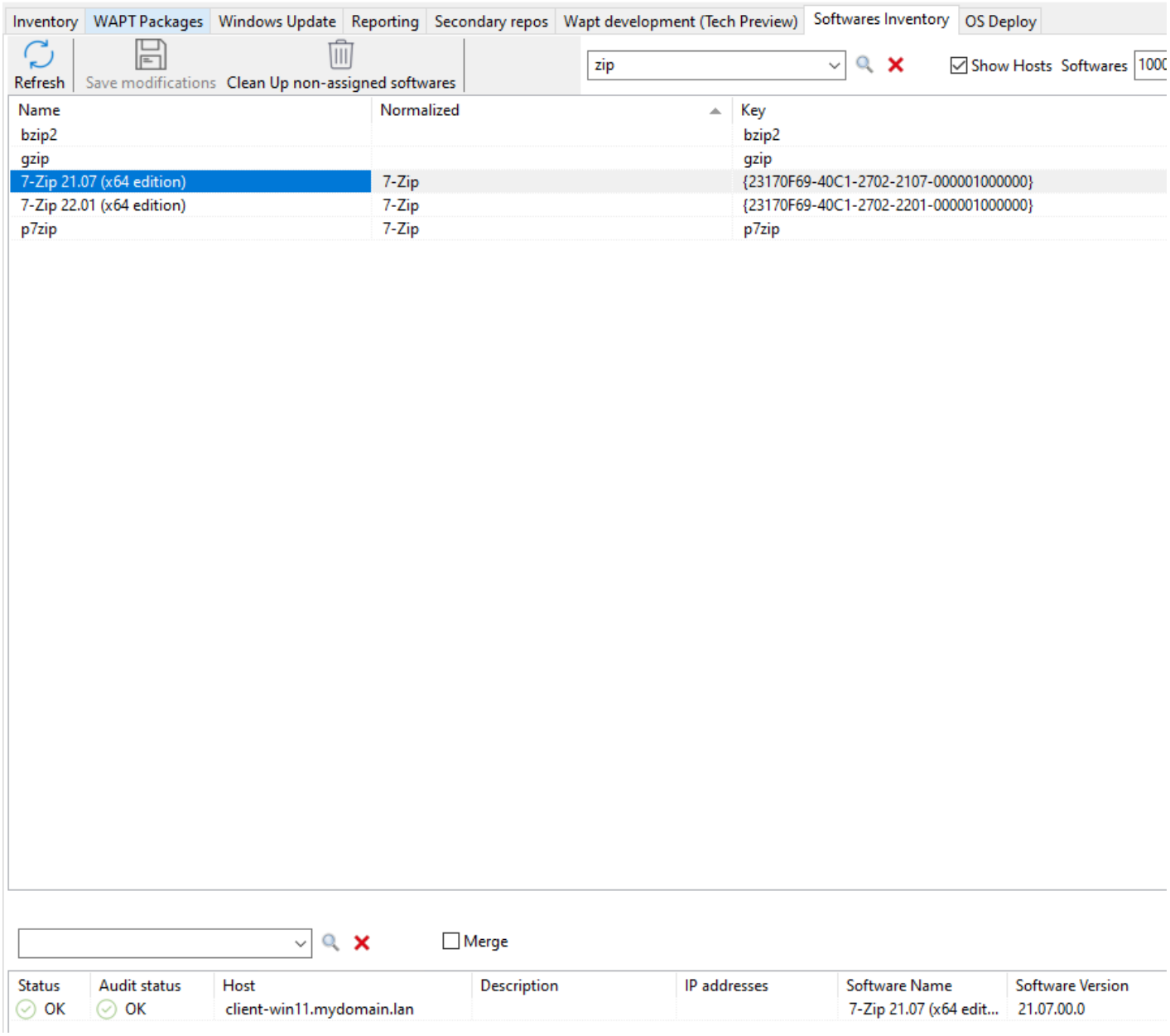


Figure9: Liste des machines avec le logiciel sélectionné dans l'onglet Inventaire des logiciels

30.5.1 Utiliser des titres de logiciels normalisés comme filtre dans l'onglet inventaire

Vous pouvez créer un filtre dans l'onglet *Inventaire* qui utilise des noms de logiciels normalisés. Pour ce faire, normaliser les noms des logiciels dans *Inventaire des logiciels* puis sélectionner une ou plusieurs machines. Dans l'onglet *Inventaire des logiciels* des machines sélectionnées, glissez et déposez le logiciel dans la liste d'inventaire, cela créera une vue.

Indication: Cette méthode est puissante car elle permet de rechercher des titres de logiciels qui n'ont pas été installés à l'aide de WAPT.

Ne pas oublier de normaliser les noms des logiciels au préalable.

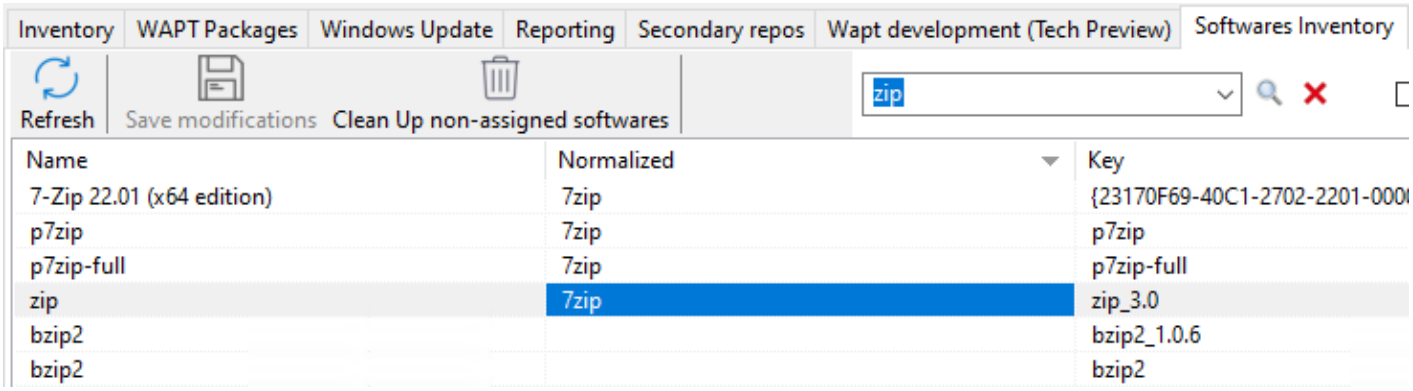


Figure10: Normaliser les noms de logiciels

Figure11: Ajouter un nom normalisé de logiciel à l'inventaire des logiciels pour les machines sélectionnées

Vous verrez une liste de machines triées selon le nom normalisé du logiciel.

30.6 Se connecter à la base de données WAPT avec un client PostgreSQL

Vous pouvez connecter votre base de données WAPT à un client si vous préférez utiliser un client PostgreSQL.

Pour ce faire, vous allez devoir changer quelques fichiers de configuration sur votre serveur WAPT.

- Tout d'abord, trouvez la version de votre base de données PostgreSQL.

```
ps -ef | grep -i sql
postgres   512     1  0 Jan05 ?        00:00:24 /usr/lib/postgresql/12/bin/postgres -D /var/lib/
↪postgresql/12/main -c config_file=/etc/postgresql/12/main/postgresql.conf
```

- Modifiez `pg_hba.conf` de la version PostgreSQL utilisée. Dans `/etc/postgresql/12/main/pg_hba.conf` pour Debian et `/var/lib/pgsql/12/data/pg_hba.conf` pour Centos sous **# IPv4 local connections section**, ajoutez votre adresse.

```
host    wapt            all             192.168.0.65/32             md5

where 192.168.0.65 is your IP address that is authorized
to connect to the WAPT database.
```

- Autorisez PostgreSQL à écouter sur toutes les interfaces dans `/etc/postgresql/12/main/postgresql.conf` pour Debian et `/var/lib/pgsql/12/data/postgresql.conf` pour Centos, section **Connection Settings**.

```
listen_addresses = '*'
```

- Redémarrez le service pour votre version de PostgreSQL.

```
systemctl restart postgresql@12-main.service
```

- Pour se connecter au PostgreSQL sur le serveur wapt.

```
sudo -u postgres psql template1
```

- Puis renseignez le mot de passe de l'utilisateur wapt.

```
template1=# ALTER USER wapt WITH PASSWORD 'PASSWORD';
```

CHAPTER 31

Utilisation du Self-Service de WAPT

31.1 Présentation

Avec WAPT, vos utilisateurs peuvent disposer d'un self-service pour l'installation des logiciels.

C'est différent dans les versions **Discovery** et **Enterprise**.

Fonctionnalité	Discovery	Enterprise
Accès au self-service	✅	✅
Déploiement du paquet de self-service	✅	✅
Filtrage des paquets self-service	❌	✅
Onglet de gestion	❌	✅

31.2 Principe de fonctionnement

Les *Utilisateurs* gagnent en autonomie en déployant des logiciels et des configurations qui sont fiable et autorisés par l'*Organisation*. C'est un gain de temps pour le support informatique utilisateur de l'Organisation.

31.2.1 Discovery

Seuls les Administrateurs locaux et les membres du group *waptself-service* peuvent accéder au self-service sur l'hôte.

Attention: Ces utilisateurs ont accès à tous les paquets de votre dépôt.

31.2.2 Enterprise

Vous pouvez filtrer la liste des paquets self-service disponibles pour vos utilisateurs.

Un paquet *self-service* peut être déployé sur les hôtes pour lister les différentes règles self-service à appliquer sur l'hôte.

Les paquets *self-service* sont basés sur des groupes d'utilisateurs.

Vos utilisateurs pourront installer une sélection de paquets WAPT sans avoir besoin d'être un *Administrateur local*.

31.3 Utilisation de la fonction self-service

31.3.1 Configuration

Discovery

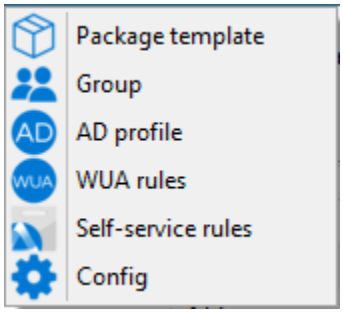
Pour Discovery, créer un groupe de sécurité *waptselfservice* sur votre Active Directory et ajoutez vos utilisateurs.

Note: **TOUS** les utilisateurs du groupe de sécurité *waptselfservice* et **TOUS** les Administrateurs Locaux auront accès à **TOUS** les paquets WAPT du dépôt.

Il n'est pas possible de filtrer les paquets WAPT rendus accessibles aux utilisateurs en mode Discovery.

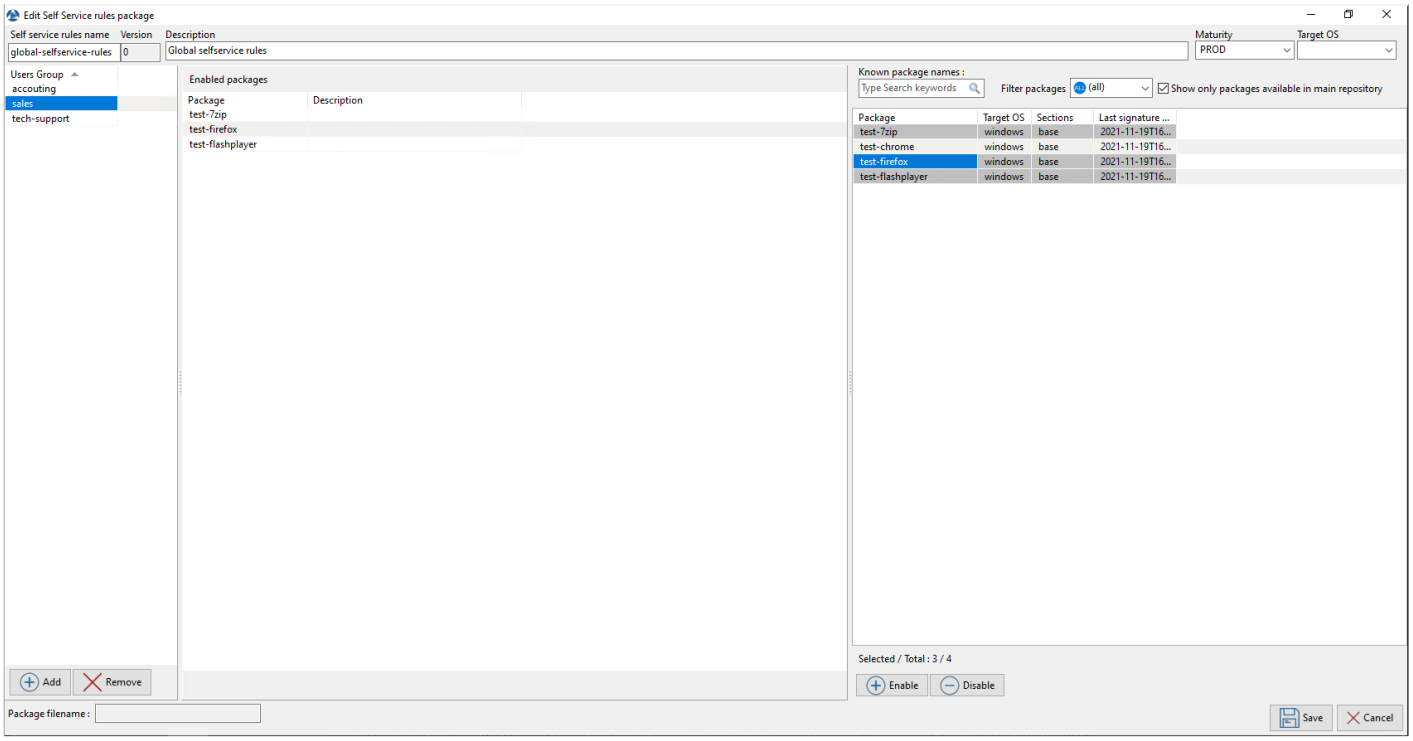
Enterprise

Dans la console, allez dans l'onglet *Dépôt privé* et créez *Règles de self-service*.



Vous pouvez désormais créer votre premier paquet de règle *self-service*.

1. Donnez un nom à votre paquet *self-service*.
2. Donner une description.
3. Cliquez sur *Ajouter* pour ajouter un groupe Active Directory (en bas à gauche).
4. Nommez le groupe *self-service* (avec `F2` ou tapez directement dans la cellule).
5. Filtrage des paquets self-serviceSélectionner la maturité du paquet *self-service*
6. Sélectionner le système d'exploitation cible pour lequel le paquet *self-service* est conçu.
7. Faites glisser et déposez dans la colonne centrale les logiciels et les paquets de configuration autorisés pour ce groupe *self-service*.



8. Ajoutez autant de groupes que vous le souhaitez dans le paquet *self-service*.
9. Sauvegarder le paquet et déployer le paquet sur votre sélection d'hôtes.

Note:

- Le nom du packaging *self-service* **Doit**être le même que le nom du **groupe de sécurité de l'utilisateur Active Directory** auquel les règles *self-service* s'appliqueront. . . .
- Si un groupe apparaît dans plusieurs paquets *self-service*, alors les règles sont fusionnées.
- L'authentification utilisée est l'authentification système par défaut, il est possible de s'authentifier avec *Active Directory*.
- Une fois le paquet déployé, seuls les paquets autorisés figurant dans le(s) groupe(s) *self-service* dont l'"*Utilisateur* est membre seront affichés à l'"*Utilisateur* connecté.

31.4 Utilisation du Self-Service de WAPT

Le self-service est accessible dans le menu de démarrage sous le nom *Self-Service logiciel WAPT*.

Il est aussi disponible directement dans le dossier de WAPT `<base>\waptself.exe`.

Note: L'ientifiant et le mot de passer à entrer lors du lancement du self-service sont ceux de l'Utilisateur (local ou Active directory).

Le self-service affiche alors une liste de paquets disponibles pour l'installation.

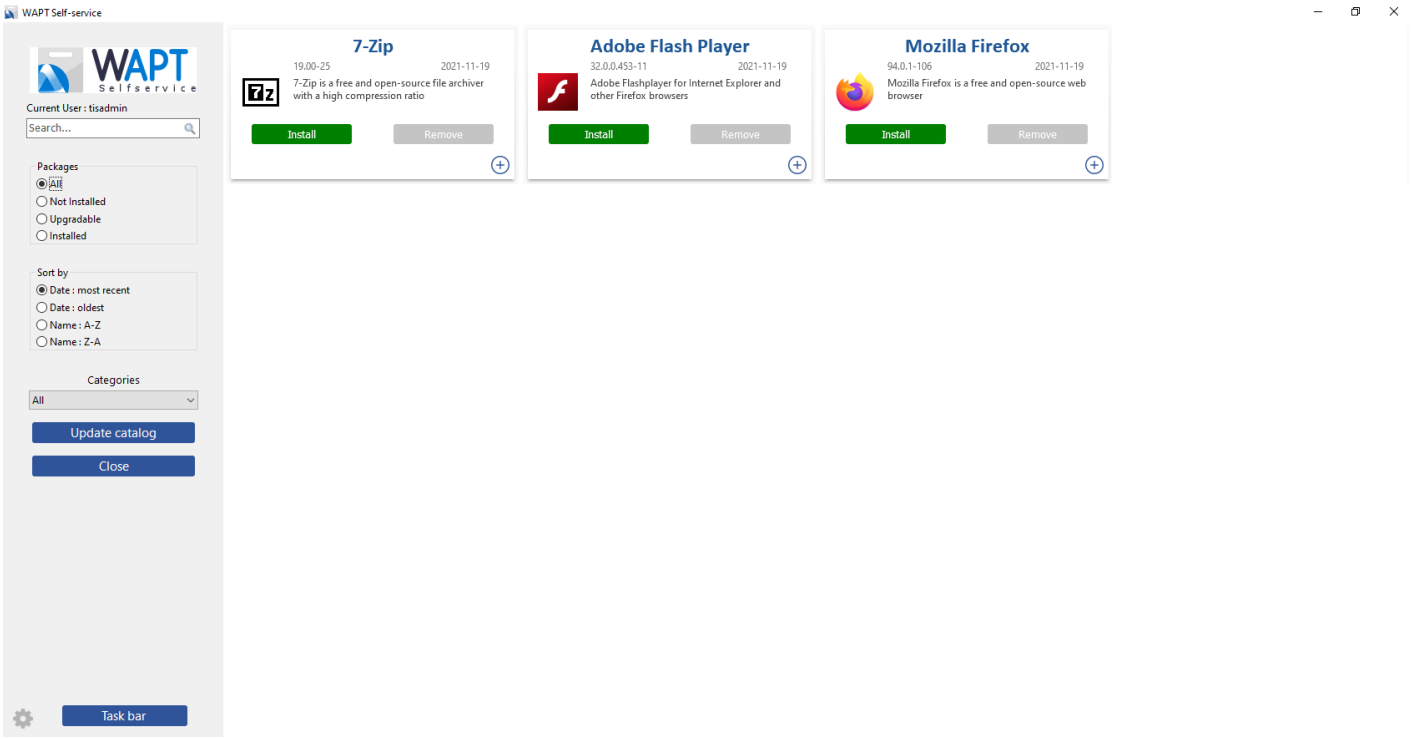
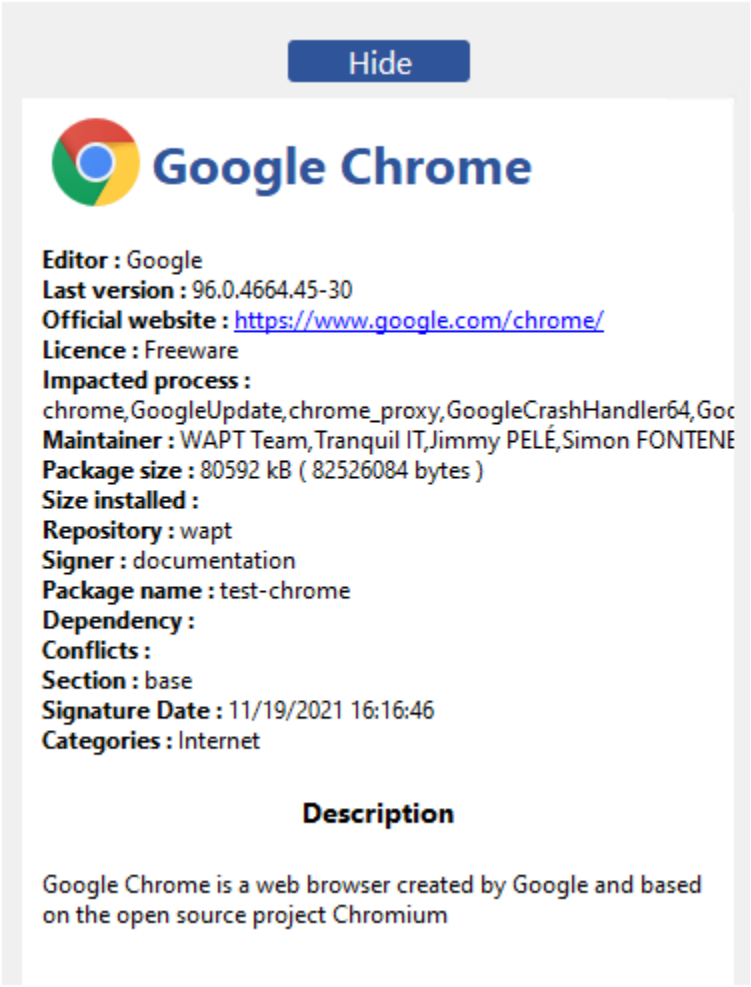
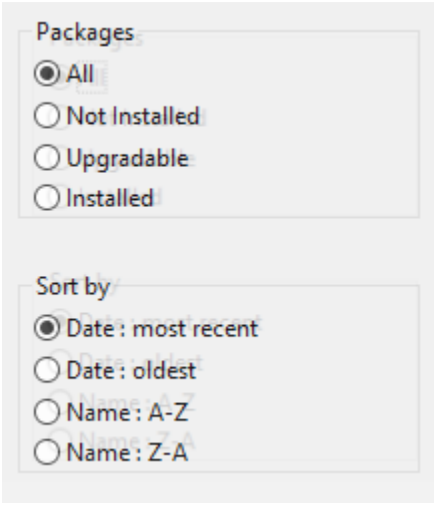


Figure1: Fenêtre principale du libre-service WAPT

- L'utilisateur peur avoir plus de détails sur chaque paquet avec l'icône +.

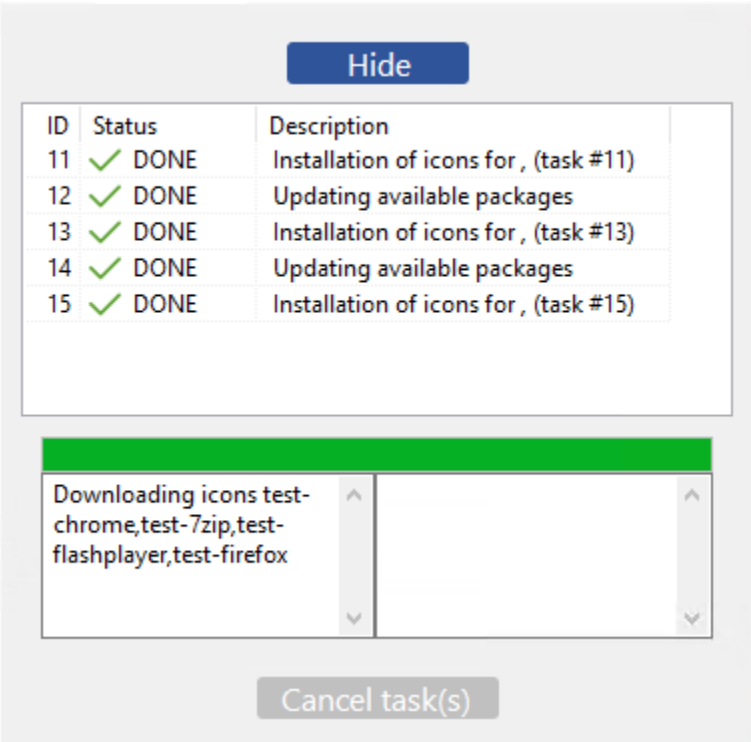


- Différents filtres sont disponibles pour l'utilisateur sur le panneau de gauche.

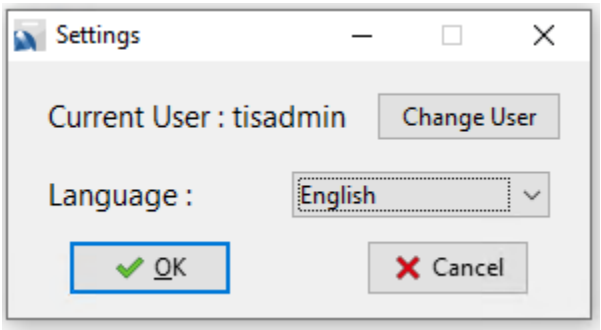


- Le bouton *Mettre à jour le catalogue* est utilisé pour forcer un **`wapt-get update`** sur l'agent WAPT;

- La liste des tâches en cours de l'agent WAPT est disponible avec le bouton *Barre de tâches*;



- Il est possible de changer la langue de l'interface avec le bouton en bas à gauche.



31.4.1 Catégories de paquets disponibles par défaut

Par défaut, WAPT gère ces catégories de paquets :

- Internet;
- Utilitaires;
- Messagerie;
- Securite;
- Syteme et reseau;
- Stockage;

- Media;
- Developpement;
- Bureautique;
- Education.

Il est possible d'*ajouter d'autres catégories* aux paquets que vous développez.

31.5 Configuration de l'agent WAPT pour le WAPT self-service

L'agent WAPT peut être configuré pour le self-service de WAPT.

31.5.1 Configurer une méthode d'authentification différente pour le self-service

Par défaut, l'authentification sur le service WAPT est configurée en mode système.

Ce comportement est défini avec la valeur de `service_auth_type` dans *wapt-get.ini* :

Valeur	Description
`système` *Valeur par défaut*	Le service WAPT transmet l'authentification directement au système d'exploitation ; il récupère également les groupes en interrogeant directement le système d'exploitation.
`waptserver-ldap`	Ce mode permet l'authentification auprès du serveur WAPT. Le serveur WAPT fera une requête LDAP pour vérifier l'authentification et les groupes. Pour que cela fonctionne, vous devez avoir configuré *l'authentification LDAP* sur le serveur WAPT.
`waptagent-ldap`	Ce mode permet l'authentification avec un serveur LDAP identifié dans `wapt-get.ini`. L'agent WAPT fera une requête LDAP pour vérifier l'authentification et les groupes. Pour que cela fonctionne, vous devez avoir configuré *l'authentification LDAP* sur le serveur WAPT.

Vous pouvez consulter cet article décrivant les *paramètres pour le Self-Service de WAPT et l'Authentification du Waptservice* pour plus d'options.

Note: Pour que l'authentification système sous GNU/Linux fonctionne correctement, assurez-vous de correctement configurer l'authentification pam et votre `nsswitch.conf`. La commande **`id username`** doit renvoyer la liste des groupes dont l'utilisateur est membre.

Avertissement: En mode `system` nous supposons que les *Administrateurs Locaux* peuvent voir tous les paquets. Pour changer ce comportement, passez au point suivant.

31.5.2 Configuration de l'authentification pour Administrateur

Par défaut le self-service WAPT utilise l'authentification `system`.

Dans ce mode, les *Administrateurs Locaux* peuvent voir tous les paquets.

Si vous ne voulez pas de ce comportement vous avez 2 possibilités :

- Bloquer l'affichage de tous les paquets pour les *Administrateurs Locaux*.
- Tous les paquets ne sont visibles que pour un groupe d'utilisateurs spécifique.

Bloquer les Administrateurs Locaux sur le self-service

Pour bloquer l'affichage de tous les paquets pour les *Administrateurs Locaux* vous devez ajouter le paramètre `waptservice_admin_filter` dans `wapt-get.ini`.

Valeur	*True*	*False*
`waptservice_admin_filter`	Activer le filtrage d'affichage du *paquet selfservice* pour les administrateurs locaux.	Désactiver le filtrage d'affichage du *paquet selfservice* pour les administrateurs locaux.

Groupe d'utilisateurs Administrateur du self-service

Il est possible d'utiliser un groupe d'utilisateurs spécial pour définir une liste d'Administrateurs dans le self-service.

Créez un groupe de sécurité d'utilisateurs nommé « waptselfservice » et ajoutez des membres.

Tous les membres de ce groupe peuvent voir tous les paquets sur le Self-Service WAPT.

Avec le paramètre `waptservice_admin_filter`, vous avez sécurisé l'accès administrateur de WAPT Self-Service.

31.6 Vidéo de démonstration

https://youtu.be/-_sm8KBwDOw

CHAPTER 32

Utilisation de l'utilitaire WAPT System Tray

WAPTtray est un programme systray. Il fonctionne dans le contexte de l'utilisateur.

WAPTtray se lance à l'ouverture de session si l'option a été cochée lors de l'installation de l'agent WAPT. L'icône apparaîtra dans la barre d'outils de la zone de notification de Windows.

On peut aussi lancer WAPTtray manuellement sur `C:Program Files (x86)waptwapttray.exe`.

32.1 Les fonctionnalités du WAPTtray

Fonctions principales

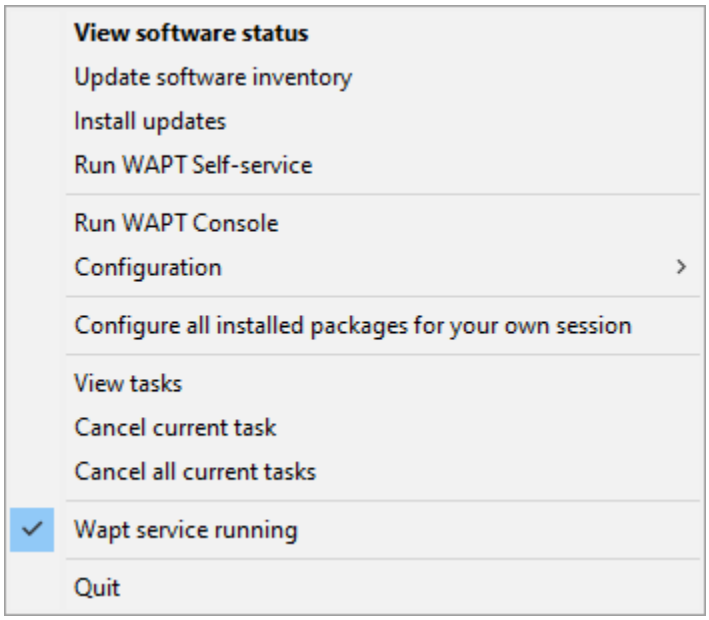


Table1: Liste des fonctionnalités de WAPTtray

Action	Description
Afficher le statut des logiciels	Lance l'interface web local dans un navigateur.
Mettre à jour de l'inventaire des logiciels	Rafraîchir la liste de paquet disponibles. Double-clic sur l'icone fait la même action.
False	Lance l'installation d'une mise à jour en attente.
True	Lance le Self-Service de WAPT.
True	Lance la console WAPT.
True	Voir le tableau suivant pour les options détaillées.
Configurer tous les paquets installés pour votre session	Lance un **`session-setup`** pour configurer tous les paquets installés dans l'environnement utilisateur.
False	Afficher la liste des tâches sur l'interface web locale dans le navigateur.
Annuler la tâche en cours	Annule la tâche en court d'exécution sur l'agent WAPT.
Annuler toutes les tâches en cours	Annule toutes les tâches en cous d'exécution sur l'agent WAPT.
Service WAPT lancé	Arrête et relance le service WAPT.
True	Ferme l'icone dans la barre de notification sans stopper le service WAPT local.

Fonctions de configuration

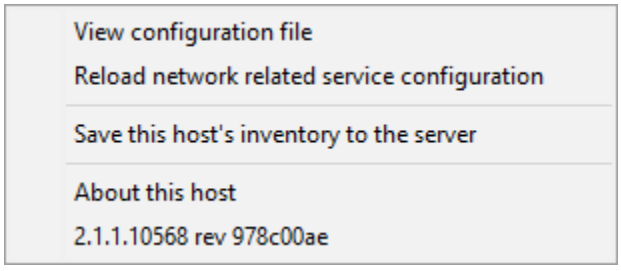


Table2: Liste des options de configuration de l'utilitaire WAPT System Tray

Action	Description
Afficher le fichier de configuration	Ouvre le fichier `C:\Program Files (x86)wapt\wapt-get.ini` avec les privilèges *Administrateur local* (les informations d'identification peuvent être demandées)
Recharger la configuration réseau du service	Relance la connexion au serveur WAPT en cas de reconfiguration du réseau.
Sauvegarder l'inventaire de cette machine sur le serveur	Mise à jour de l'inventaire de l'hôte avec le serveur WAPT.
Informations de la machine	Lance l'interface web locale dans un fichier de navigation avec les privilèges d'*Administrateur Local* (les informations d'identification peuvent être demandées) pour afficher l'inventaire des machines.

32.2 Vidéo de démonstration

https://youtu.be/9iG36IeHuVc

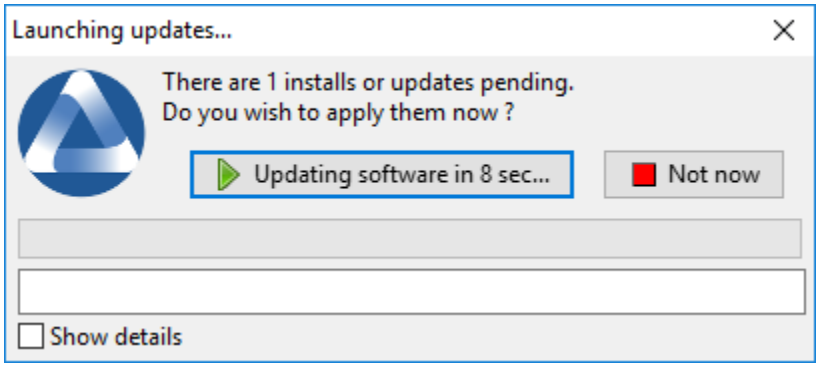
CHAPTER 33

Utilisation de l'utilitaire WAPT Exit

WAPTExit permet de mettre à jour et d'installer les paquets WAPT lorsqu'un hôte s'arrête, à la demande de l'utilisateur, ou à une heure programmée.

Le mécanisme est simple. Si les paquets sont en attente de mise à jour, ils seront installés.

Indication: La méthode WAPTexit est très efficace dans la plupart des situations car elle ne nécessite pas l'intervention du *User* ou du *Administrator*.



WAPTExit s'exécute par défaut à l'arrêt, il est installé avec l'agent WAPT.

Le comportement de WAPTExit est personnalisable dans *wapt-get.ini*.

Avertissement: Si une tâche est en cours d'exécution, l'arrêt est suspendu jusqu'à ce que la tâche soit terminée.

33.1 Déclencher manuellement l'exécution de WAPTexit

On peut aussi lancer WAPTtray manuellement sur `C:\Program Files (x86)\wapt\waptexit.exe`.

33.2 Déclencher WAPTexit avec une tâche plannifiée

On peut déployer une GPO ou un paquet WAPT qui va déclencher le WAPTexit à un moment prédéfini.

Indication: Déclencher WAPTexit avec une tâche planifiée convient mieux aux serveurs qui ne sont pas arrêtés fréquemment.

Vous pouvez adapter *la procédure de déploiement de l'agent WAPT* pour déclencher le script WAPTexit.exe au moment de votre choix.

Indication: Vous pouvez utiliser le script suivant pour votre tâche planifiée, adaptée à vos besoins :

```
waptpython -c "from waptservice.enterprise import start_waptexit start_waptexit('',{'only_priorities
↪':False,'only_if_not_process_running':True, 'install_wua_updates':False,'countdown':300},'schtask
↪')"
```

Avertissement:

- Tous les logiciels en cours d'exécution qui sont mis à jour peuvent être détruits avec une possible perte de données.
- WAPTExit peut échouer à mettre à niveau un logiciel si un logiciel que vous mettez à niveau figure dans la liste `impacted_process` du fichier :file :*control*. Voir *sous* pour plus d'informations.
- La méthode consistant à déclencher WAPTExit à une heure planifiée est la moins recommandée pour les ordinateurs de bureau. Il est préférable de laisser le WAPTExit s'exécuter à l'arrêt ou à la demande de l'utilisateur.

33.3 Paramètres de WAPTExit dans wapt-get.ini

Il est possible de *modifier le comportement de WAPTExit* dans le `wapt-get.ini`.

Il est également possible de modifier le comportement de WAPTExit directement depuis la ligne de commande, voir les points suivants.

33.4 Les options de l'utilitaire WAPT Exit avec la ligne de commande

33.4.1 Empêcher l'annulation des mises à jour

Pour désactiver l'interruption de l'installation des mises à jour, vous pouvez exécuter WAPTExit avec l'argument :

```
waptexit.exe -allow_cancel_upgrade = True
```

33.4.2 Augmenter le temps de déclenchement dans waptexit

Pour spécifier le temps d'attente avant le démarrage automatique des installations, vous pouvez lancer WAPTExit avec l'argument :

```
waptexit.exe -waptexit_countdown = 10000
```

33.5 Ne pas interrompre l'activité de l'utilisateur

Pour indiquer à WAPT de ne pas exécuter une *mise à niveau* des titres logiciels en cours d'exécution sur l'hôte (attribut `impacted_process` du packaging WAPT), l'utilitaire WAPT Exit peut être exécuté avec l'argument `-only_if_not_process_running`.

```
waptexit.exe -only_if_not_process_running = True
```

Sinon, **waptexit** prendra la valeur indiquée dans `C:\Program Files (x86)wapt\wapt-get.ini` .

33.6 Lancement de l'installation de paquets avec un niveau de priorité spécial

Pour dire à WAPT de ne mettre à jour que les paquets WAPT avec une priorité spécifique, vous pouvez exécuter l'utilitaire WAPT Exit avec l'argument `-priorities`.

```
waptexit.exe -priorities = high
```

33.7 Activer/désactiver WAPTexit

Pour activer ou désactiver **waptexit** dans les scripts de stratégie de groupe locale, utilisez :

- pour activer le **waptexit** à l'extinction du poste:

```
wapt-get add-upgrade-shutdown
```

- pour désactiver le **waptexit** à l'extinction du poste:

```
wapt-get remove-upgrade-shutdown
```

33.8 Vidéo de démonstration

CHAPTER 34

Personnaliser WAPT

Il est possible de personnaliser WAPT aux couleurs de votre société.

3 programmes sont personnalisables :

- l'utilitaire WAPT Exit ;
- le self-service WAPT ;
- l'utilitaire WAPT Message.

Il est possible d'utiliser le même logo pour tous les programmes.

Placer l'image dans `<wapt_folder>\templates`.

Le logo doit être nommé `wapt-logo.png`

La taille recommandée pour le logo est 200X55 et le format est `.png`

Pour différents logo par programme, voir les points suivants.

34.1 L'utilitaire WAPT Exit

Il est possible de personnaliser waptexit en plaçant l'image que vous voulez dans `<wapt_folder>\templates`

Le logo doit être nommé `waptexit-logo.png`

La taille recommandée pour le logo est 200X55 et le format est `.png`

S'il n'est pas défini, WAPT utilise `wapt-logo.png`. S'il n'existe pas, utilisez un logo WAPT par défaut.

34.2 WAPT Self-Service

Il est possible de personnaliser waptexit en plaçant l'image que vous voulez dans <wapt_folder>\templates

Le logo doit être nommé waptself-logo.png

La taille recommandée pour le logo est 200X55 et le format est .png

S'il n'est pas défini, WAPT utilise dans l'ordre waptexit-logo.png, waptself-logo.png et enfin le logo WAPT par défaut.

34.3 WAPT Message

Il est possible de personnaliser waptexit en plaçant l'image que vous voulez dans <wapt_folder>\templates

Le logo doit être nommé waptmessage-logo.png

La taille recommandée pour le logo est 200X55 et le format est .png

S'il n'est pas défini, WAPT utilise dans l'ordre waptexit-logo.png, waptself-logo.png et enfin le logo WAPT par défaut.

CHAPTER 35

Utiliser WAPT en ligne de commande

L'agent WAPT fournit un utilitaire d'interface de ligne de commande **`wapt-get`**.

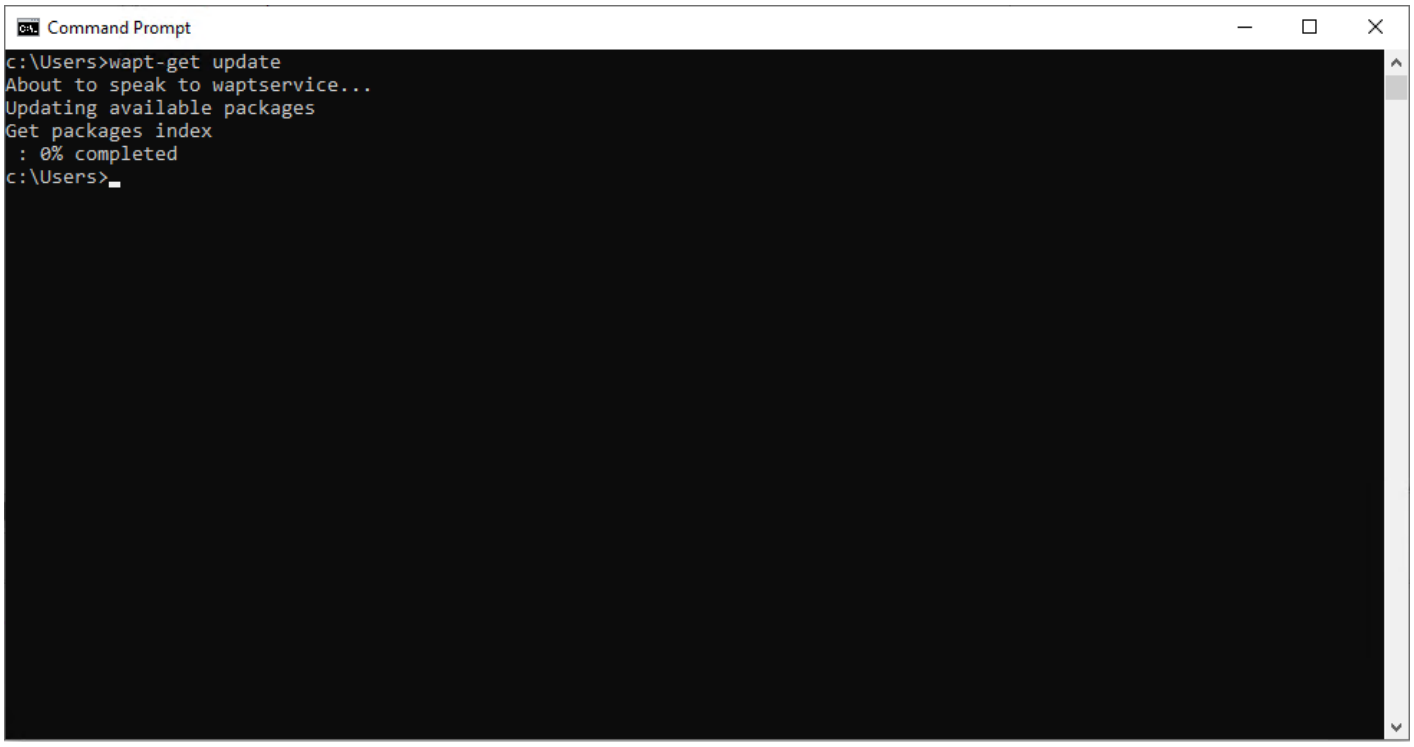


Figure1: L'invite de commande Windows

Note:

- Par défaut, les actions en ligne de commande dans WAPT sont exécutées avec les droits de l'utilisateur qui a lancé le **cmd.exe**.
- Si le **cmd.exe** n'a pas été lancé avec les privilèges de *Local Administrator*, la commande sera transmise au **waptservice**.
- Par sécurité, certaines actions demandent un identifiant et un mot de passe.
- Seuls les *Administrateurs Locaux* et les membres du groupe de sécurité Active Directory *waptselfservice* sont autorisés.
- Pour forcer l'utilisation du service WAPT en tant qu'*Administrateur Local*, ajouter simplement *-S* après `wapt-get.exe`.

Note: Chaque commande qui prend un nom de paquet comme paramètre peut également prendre le *package_uuid* unique du paquet comme paramètre (**wapt-get install**, **wapt-get forget**, etc.). L'utilisation d'un GUID permet de spécifier un paquet unique sans ambiguïté sur son architecture ou sa version. Le *package_uuid* est listé dans la sortie de **wapt-get list** et **wapt-get search**. Par exemple :

35.1 Utilisation des fonctions les plus courantes dans WAPT via ligne de commande

35.1.1 wapt-get install

La commande `wapt-get install <package name>` lance l'installation d'un paquet.

Pour installer Mozilla Firefox, la commande est **wapt-get install tis-firefox**.

Il est possible d'installer plusieurs paquets à la fois :

```
wapt-get install package1 package2
```

Si le paquet n'a pas été téléchargé dans le cache, **wapt-get install** téléchargera d'abord le paquet dans le cache, puis l'Agent WAPT installera le paquet.

Attention: Installer un paquet WAPT avec **wapt-get install** n'ajoute pas le paquet comme dépendance à la machine.

Le paquet est installé sur la machine, mais si l'ordinateur est réimagé, le paquet ne sera pas réinstallé automatiquement.

La commande `wapt-get install tis-firefox` renvoie :

```
Using config file: C:\Program Files (x86)\wapt\wapt-get.ini
installing WAPT packages tis-firefox
https://srvwapt.mydomain.lan/wapt/tis-firefox_94.0.1-106_x64_windows_
↪0f4137ed1502b5045d6083aa258b5c42_6.1_PROD_en_f5335369ad5805e8dbc1f8ae99b2258e.wapt : 14121562 /␣
↪54313787 (26%) (24624 KB/s)
https://srvwapt.mydomain.lan/wapt/tis-firefox_94.0.1-106_x64_windows_
↪0f4137ed1502b5045d6083aa258b5c42_6.1_PROD_en_f5335369ad5805e8dbc1f8ae99b2258e.wapt : 33131357 /␣
↪54313787 (61%) (29414 KB/s)
https://srvwapt.mydomain.lan/wapt/tis-firefox_94.0.1-106_x64_windows_
↪0f4137ed1502b5045d6083aa258b5c42_6.1_PROD_en_f5335369ad5805e8dbc1f8ae99b2258e.wapt : 50511741 /␣
↪54313787 (93%) (30412 KB/s)
https://srvwapt.mydomain.lan/wapt/tis-firefox_94.0.1-106_x64_windows_
↪0f4137ed1502b5045d6083aa258b5c42_6.1_PROD_en_f5335369ad5805e8dbc1f8ae99b2258e.wapt : 54313787 /␣
↪54313787 (100%) (30360 KB/s)
```

(suite sur la page suivante)

(suite de la page précédente)

```
Installing tis-firefox(=94.0.1-106)
Installing: Firefox_Setup_94.0.1.exe
Waiting for key key Mozilla Firefox 94.0.1 (x64 en-US) to appear in Windows registry
Delete C:\Program Files (x86)\wapt\cache\tis-firefox_94.0.1-106_x64_windows_
↪0f4137ed1502b5045d6083aa258b5c42_6.1_PROD_en_f5335369ad5805e8dbc1f8ae99b2258e.wapt

Results:

=== install packages ===
  tis-firefox [x64_en_PROD]     | tis-firefox (94.0.1-106)                  | tis-firefox (50.0.2-
↪73)
```

35.1.2 wapt-get update

La commande **wapt-get update** permet de mettre à jour la liste des paquets disponibles.

L'agent WAPT local téléchargera le fichier `Packages` du dépôt privé et le comparera à sa base de données locale.

- Si de nouvelles mises à jour sont disponibles, l'agent WAPT fait passer le statut des paquets à **TO-UPGRADE**.
- Si de nouveaux logiciels ont été ajoutés sur le dépôt, ils deviennent disponibles pour le téléchargement par l'Agent WAPT.

Note: La commande `wapt-get update` ne télécharge pas les paquets, elle met seulement à jour la base de données locale des paquets.

La commande `wapt-get update` renvoie:

```
Using config file: C:\Program Files (x86)\wapt\wapt-get.ini
Update package list from https://srvwapt.mydomain.lan/wapt, https://srvwapt.mydomain.lan/wapt-host
Total packages: 8
Added packages:

Removed packages:

Discarded packages count: 6
Pending operations:
  install:
  upgrade:
  additional:
  remove:
  immediate_installs:
Repositories URL :
  https://srvwapt.mydomain.lan/wapt
  https://srvwapt.mydomain.lan/wapt-host
```

35.1.3 wapt-get upgrade

La commande **wapt-get upgrade** lance l'installation des paquets WAPT en attente de mise à niveau ou en attente d'installation.

L'agent WAPT local télécharge d'abord ses paquets WAPT dans le cache local, puis l'agent WAPT les installe.

Indication: Il est recommandé de lancer la commande `wapt-get update` avant de lancer une commande `wapt-get upgrade`.

Sans le lancement préalable d'un `update`, l'agent WAPT n'installera rien.

La commande `wapt-get upgrade` renvoie:

```
Installing tis-mumble
Shutting down Mumble
installing Mumble 1.2.3

=== install packages ===
tis-mumble
```

35.1.4 wapt-get remove

La commande **wapt-get remove <nom du paquet>** supprime les paquets WAPT listés de la machine.

La commande prend un argument. Cet argument est le nom paquet ou une liste de noms de paquets avec le préfixe du dépôt.

Pour supprimer Mozilla Firefox, la commande est **wapt-get remove <prefix>-firefox**.

Attention: La suppression d'un paquet WAPT avec `remove` ne supprime pas la dépendance du paquet sur l'hôte.

Le paquet sera effectivement désinstallé de la machine, mais il sera automatiquement réinstallé lors de la prochaine mise à niveau

Pour supprimer complètement un paquet d'un hôte, faites un `remove` pour le paquet ciblé, puis modifiez la configuration de l'hôte via la console WAPT pour supprimer la dépendance du paquet sur l'hôte.

La commande `wapt-get remove tis-firefox` renvoie :

```
Using config file: C:\Program Files (x86)\wapt\wapt-get.ini
Removing tis-firefox ...

Waiting for the removal of key key Mozilla Firefox 94.0.1 (x64 en-US) from Windows registry
=== Removed packages ===
  tis-firefox
```

35.1.5 wapt-get install

La commande **wapt-get uninstall [<nom du paquet>]** désinstalle les paquets listés de la machine si une fonction `def uninstall()` existe dans les fichiers `setup.py` des paquets listés.

La commande prend un argument. Cet argument est le nom paquet ou une liste de noms de paquets avec le préfixe du dépôt.

Attention: Exécuter la fonction de désinstallation du paquet ne supprime pas le paquet en cache sur la machine.

La commande **wapt-get uninstall tis-adwcleaner** renvoie :

```
Using config file: C:\Program Files (x86)\wapt\wapt-get.ini
 Uninstalling tis-adwcleaner ...
 None
 Uninstallation done
```

35.1.6 wapt-get edit

La commande **wapt-get forget <nom du paquet>** supprime le paquet de la base de données locale afin que le cycle de vie du titre du logiciel ou de la configuration ne soit plus géré par WAPT.

La commande prend un argument. Cet argument est le nom paquet ou une liste de noms de paquets avec le préfixe du dépôt.

Attention: Oublier le paquet WAPT ne désinstalle pas le logiciel ni la configuration associée au paquet WAPT.

La commande **wapt-get forget tis-adwcleaner** renvoie :

```
Using config file: C:\Program Files (x86)\wapt\wapt-get.ini

=== Packages removed from status ===
  tis-adwcleaner
```

35.1.7 wapt-get edit

La commande **wapt-get audit [<nom du paquet>]** exécute la fonction d'audit pour les paquets listés si une fonction `def audit()` existe dans les fichiers `setup.py` des paquets listés.

La commande prend un argument. Cet argument est le nom paquet ou une liste de noms de paquets avec le préfixe du dépôt.

De même, la commande **wapt-get audit ALL** exécute la fonction d'audit pour tous les paquets installés sur la machine.

La commande **wapt-get audit tis-firefox** renvoie :

```
Using config file: C:\Program Files (x86)\wapt\wapt-get.ini
Auditing tis-firefox ...
Auditing tis-firefox
OK: Uninstall Key Mozilla Firefox 94.0.1 (x64 en-US) in Windows Registry.
tis-firefox -> OK
```

35.1.8 wapt-get show

La commande `wapt-get show <nom du paquet>` affiche les informations stockées dans le fichier d'index `Packages`.

Si plusieurs versions d'un paquet WAPT sont disponibles sur le dépôt WAPT, chaque version du paquet sera affichée.

La commande `wapt-get show tis-7zip` renvoie :

```
Using config file: C:\Program Files (x86)\wapt\wapt-get.ini
Display package control data for tis-7zip

package           : tis-7zip
version           : 19.00-25
architecture      : x64
section           : base
priority          : optional
name              : 7-Zip
categories        : Utilities
maintainer        : WAPT Team,Tranquil IT,Jimmy PELÉ
description       : 7-Zip is a free and open-source file archiver with a high compression ratio
depends           :
conflicts         :
maturity          : PROD
locale            : all
target_os         : windows
min_wapt_version  : 1.7
sources           : https://www.7-zip.org/download.html
installed_size    :
impacted_process  : 7zFM,7z,7zG
description_fr    : 7-Zip est un logiciel gratuit et open source pour archiver des fichiers avec un␣
↪taux de compression élevé
description_pl    :
description_de    : 7-Zip ist ein Datenkompressionsprogramm mit einer hohen Kompressionsrate
description_es    : 7-Zip es un archivador de ficheros con una alta relación de compresión
description_pt    : O 7-Zip é um compactador de arquivos com alta taxa de compressão
description_it    :
description_nl    :
description_ru    : 7-Zip
audit_schedule    :
editor            : Igor Pavlov
keywords          : 7zip,7,zip,7-zip,file,archiver,high,compression,ratio
licence           : LGPL
homepage          : https://www.7-zip.org/
package_uuid      : dc66ccd1-d987-482e-b792-04e89a3803f7
valid_from        :
valid_until       :
forced_install_on :
changelog         : https://www.7-zip.org/history.txt
min_os_version    : 5.0
max_os_version    :
icon_sha256sum    : eddc038d3625902b6ddeaabd13dd91529e8d457ffbd0c554f96d343ae243a67a
signer            : documentation
```

(suite sur la page suivante)

(suite de la page précédente)

```
signer_fingerprint: 3f2c0a02231a36eafa1f67905f5c083e4b66cb59942f69cbd231d778a1a25b3d
signature         : QzhPeZFrRbjcGzfqRpoWsDP9Plaz6BBVlL3adq/MRM19D61+Aez/
↪JiA8skriCgwSErJXbxOPfxusVqqIpEtyoqh/RlRcnmgCQqk2Fig4gmxpz0rHKokukPQlRk+HdC/
↪uByxSjfp9oXuB3PVG2PZAFifjVBtjEX2QmV+OY6NdMI9dtkxCsn1Xotn2qhu2bwbJWQ0s51rD9emWuQR7l/
↪8WXl+HoquuRho4aCeAOYd6Nta9ktVSR2FM6OO5ZeUOg4fsnMg+hwp2MlDOmBHX37aJm3hLYkGP2xWjpL9YDDxI7ruRXSHyT7YmbILrS0h1m
signature_date    : 2021-11-19T16:15:42.019196
signed_attributes : package,version,architecture,section,priority,name,categories,maintainer,
↪description,depends,conflicts,maturity,locale,target_os,min_wapt_version,sources,installed_size,
↪impacted_process,description_fr,description_pl,description_de,description_es,description_pt,
↪description_it,description_nl,description_ru,audit_schedule,editor,keywords,licence,homepage,
↪package_uuid,valid_from,valid_until,forced_install_on,changelog,min_os_version,max_os_version,
↪icon_sha256sum,signer,signer_fingerprint,signature_date,signed_attributes
filename          : tis-7zip_19.00-25_x64_windows_0f4137ed1502b5045d6083aa258b5c42_5.0_PROD_
↪a10c57d7848cf7b145d6cd64bf4d5389.wapt
size              : 1704227
md5sum            : a10c57d7848cf7b145d6cd64bf4d5389

OK Package control signature checked properly by certificate documentation (fingerprint:␣
↪3f2c0a02231a36eafa1f67905f5c083e4b66cb59942f69cbd231d778a1a25b3d )
```

Note:

```
WARNING: control data signature can not be validated with certificates [<SSLCertificate cn=
↪'documentation' fingerprint=3f2c0a issuer='documentation' validity=2021-11-19 - 2031-11-17 Code-
↪Signing=True CA=True>]
```

Si ce message apparaît, c'est que le certificat n'est pas fiable.

Si vous voulez vérifier le paquet correctement, téléchargez-le dans le cache et exécutez la commande `wapt-get show` sur le paquet local.

Par exemple :

```
wapt-get download tis-7zip
wapt-get show "C:\Program Files (x86)\wapt\cache\tis-7zip_19.00-25_x64_windows_
↪0f4137ed1502b5045d6083aa258b5c42_5.0_PROD_a10c57d7848cf7b145d6cd64bf4d5389.wapt"
```

35.1.9 wapt-get show

La commande **`wapt-get show-params <nom du paquet>`** renvoie une liste de *paramètres* qui seraient transmis à la commande **`wapt-get install <nom du paquet> --params=PARAMS`**.

La commande prend un argument. Cet argument est le nom paquet ou une liste de noms de paquets avec le préfixe du dépôt.

La commande `wapt-get show-params tis-7zip` renvoie :

```
Using config file: C:\Program Files (x86)\wapt\wapt-get.ini
tis-7zip : {True, 'documentation': True}
```

35.1.10 wapt-get show

La commande **wapt-get show-log <nom du paquet>** renvoie les derniers journaux d'audit déposés dans la base de données sqlite locale de l'Agent WAPT.

La commande prend un argument. Cet argument est le nom paquet ou une liste de noms de paquets avec le préfixe du dépôt.

La commande `wapt-get show tis-7zip` renvoie :

```
Using config file: C:\Program Files (x86)\wapt\wapt-get.ini
Package: tis-7zip (21.06-34) PROD
--------------------
Status: OK

Installation log:
--------------------
Installing: 7z2106-x64.msi
Waiting for key key {23170F69-40C1-2702-2106-000001000000} to appear in Windows registry

Installation Parameters:
--------------------
{}

Last audit:
--------------------
Status: OK
Date: 2022-01-06T10:32:38.698272

Output:
Auditing tis-7zip
OK: Uninstall Key {23170F69-40C1-2702-2106-000001000000} in Windows Registry.

Next audit on: 2022-01-06T10:32:38.698272
```

35.1.11 wapt-get search

La commande `wapt-get search` permet de rechercher un ou plusieurs paquets dans les dépôts.

> **Avertissement:** Cette commande renvoie uniquement les paquets WAPT disponibles pour la machine qui exécute la commande, selon la locale, le système d'exploitation, l'architecture ou la maturité de l'hôte.
>
> Si une autre localisation, os, architecture ou maturité est présente dans le dépôt, elle n'est pas listée.

La commande de recherche prend un argument de type mot-clé.

La commande `wapt-get search "Firefox"` renvoie (par exemple) :

```
status     package            version    target_os       architecture maturity locale description  ␣
↪                                                    repo
---------- ------------------ ---------- --------------- ---- ----- --- ----------------------------
↪------------------------------------------------- ----------
```

(suite sur la page suivante)

(suite de la page précédente)

```
-          tis-firefox        94.0.2-106 windows          x64  PROD  fr  Mozilla Firefox est un␣
↪navigateur web gratuit et open source                     wapt
I          tis-config-firefox 68.3-6     windows          all  PROD      Configuration for Mozilla␣
↪Firefox - The package will not have any effect if an*  wapt
I          tis-firefox-esr    91.3.0-105 windows          x64  PROD  fr  Mozilla Firefox Extended␣
↪Support Release (ESR) est une version officielle de*    wapt
```

Value	status	package	version	target_os	architecture	maturity	locale	description	repo
Description	État de l'installation des paquets	Nom du paquet	Version du paquet	OS cible (si défini)	Architecture du CPU (si définie)	Maturité du paquet (si défini)	Locale du paquet (si défini)	Description du paquet	Dossier du paquet sur le serveur

Note: La valeur de *status* définit l'état de l'installation comme suit :

- `-` pour non installé.
- `I` pour installé.

35.1.12 wapt-get download

La commande **`wapt-get download <nom du paquet>`** télécharge le paquet WAPT dans le cache local.

La commande `wapt-get download tis-7zip` renvoie :

```
Using config file: C:\Program Files (x86)\wapt\wapt-get.ini
Downloading packages tis-7zip(=19.00-25)
https://srvwapt.mydomain.lan/wapt/tis-7zip_19.00-25_x64_windows_0f4137ed1502b5045d6083aa258b5c42_5.
↪0_PROD_a10c57d7848cf7b145d6cd64bf4d5389.wapt : 1704227 / 1704227 (100%) (11804 KB/s)

Downloaded packages:
  C:\Program Files (x86)\wapt\cache\tis-7zip_19.00-25_x64_windows_0f4137ed1502b5045d6083aa258b5c42_
↪5.0_PROD_a10c57d7848cf7b145d6cd64bf4d5389.wapt
```

35.1.13 wapt-get download-upgrade

La commande **`wapt-get download-upgrade`** télécharge les paquets à mettre à niveau dans le cache WAPT local.

La commande `wapt-get download-upgrade` renvoie :

```
Using config file: C:\Program Files (x86)\wapt\wapt-get.ini
https://srvwapt.mydomain.lan/wapt/tis-firefox_94.0.1-106_x64_windows_
↪0f4137ed1502b5045d6083aa258b5c42_6.1_PROD_en_f5335369ad5805e8dbc1f8ae99b2258e.wapt : 18466658 /␣
↪54313787 (34%) (32089 KB/s)
https://srvwapt.mydomain.lan/wapt/tis-firefox_94.0.1-106_x64_windows_
↪0f4137ed1502b5045d6083aa258b5c42_6.1_PROD_en_f5335369ad5805e8dbc1f8ae99b2258e.wapt : 36390179 /␣
↪54313787 (67%) (32693 KB/s)
https://srvwapt.mydomain.lan/wapt/tis-firefox_94.0.1-106_x64_windows_
↪0f4137ed1502b5045d6083aa258b5c42_6.1_PROD_en_f5335369ad5805e8dbc1f8ae99b2258e.wapt : 52684289 /␣
↪54313787 (97%) (31564 KB/s)
```

(suite sur la page suivante)

(suite de la page précédente)

```
https://srvwapt.mydomain.lan/wapt/tis-firefox_94.0.1-106_x64_windows_
↪0f4137ed1502b5045d6083aa258b5c42_6.1_PROD_en_f5335369ad5805e8dbc1f8ae99b2258e.wapt : 54313787 /␣
↪54313787 (100%) (30747 KB/s)

=== downloaded packages ===
  C:\Program Files (x86)\wapt\cache\B8D346E7-DDDB-0013-5A8A-425CF3B6199E.wapt
  C:\Program Files (x86)\wapt\cache\tis-firefox_94.0.1-106_x64_windows_
↪0f4137ed1502b5045d6083aa258b5c42_6.1_PROD_en_f5335369ad5805e8dbc1f8ae99b2258e.wapt
```

35.1.14 wapt-get list

La commande `wapt-get list` liste les paquets WAPT qui sont installés sur l'ordinateur.

La commande `wapt-get list` renvoie :

```
Using config file: C:\Program Files (x86)\wapt\wapt-get.ini
package                              version    install_status install_date     description        ␣
↪                                                              package_uuid
----------------------------------- ---------- ---------- ---------------- ------------------------
↪------------------------------------------------------- ------------------------------------
tis-7zip                            21.06-34   OK         2021-12-10T14:57 7-Zip is a free and open-
↪source file archiver with a high compression ratio      717a30cc-0d44-42d1-9538-0f2f298d8603
tis-firefox                         94.0.1-106 OK         2021-12-10T14:58 Mozilla Firefox is a␣
↪free and open-source web browser                            5a91f54a-3e27-44cf-a2b6-6b84012aa3a2
```

package	version	install status	install_date	description	package_uuid
Nom du paquet	Version du paquet	Installation status	Date et heure de l'installation	Description du paquet	UUID unique du paquet

35.1.15 wapt-get upgrade

La commande **`wapt-get list-upgrade`** liste les paquets WAPT qui doivent être mis à niveau sur la machine.

La commande **`wapt-get list-upgrade`** renvoie :

```
Using config file: C:\Program Files (x86)\wapt\wapt-get.ini

=== upgrade packages ===
tis-notepadplusplus(=8.2-10)
```

35.1.16 wapt-get list

La commande **wapt-get -S tasks** vérifie si certaines tâches sont en cours d'exécution ou en attente dans la file d'attente.

La commande **wapt-get -S tasks** renvoie :

```
About to speak to waptservice...
Running task 14: Uninstall of tis-vlc (task #14), status:
```

35.2 Utilisation de lignes de commande spéciales avec WAPT

35.2.1 wapt-get restart-waptservice

La **wapt-get restart-waptservice** redémarre le **:programme:`waptservice`** sous Windows, Linux et macOS.

35.2.2 wapt-get add-config-from-url

La commande **wapt-get add-config-from-url <filelink> <sha256hashfile>** récupère un fichier de configuration dynamique *json* depuis l'url spécifiée et place le fichier dans le répertoire `conf.d` sous le dossier d'installation de wapt.

Le paramètre <`sha256hashfile`> est facultatif.

```
C:\Users\administrator>wapt-get add-config-from-url https://srvwapt.mydomain.lan/wapt/conf.d/
↪default_config_55863a6b54a47255097b6403731b36de716fc7ee9ec824bffad36d5fdc49b6b5.json
New config installed as C:\Program Files (x86)\wapt\conf.d\default.json
```

35.2.3 wapt-get add-config-from-file

La commande **wapt-get add-config-from-file <filepath>** ajoute un fichier de configuration dynamique *json* dans le répertoire `conf.d` sous le dossier d'installation de wapt.

Le chemin d'accès au fichier de configuration dynamique *json* est défini par <`filepath`>.

35.2.4 wapt-get add-config-from-base64

La commande **wapt-get add-config-from-file <base64 file>** ajoute un fichier de configuration dynamique *json* dans le répertoire `conf.d` sous le dossier d'installation de wapt.

Le chemin d'accès au fichier de configuration dynamique *json* est défini par <`fichier bbase64`>.

35.2.5 wapt-get remove-config

La commande **wapt-get remove-config <config-name>** supprime les fichiers de configuration dynamique *json* spécifiés du dossier `conf.d` sous le dossier d'installation de wapt.

35.2.6 wapt-get list-config

La commande **wapt-get list-config** liste les fichiers de configuration dynamique *json* installés qui sont présents dans le dossier `conf.d` sous le dossier d'installation de wapt.

```
C:\Users\administrator>wapt-get list-config
Using config file: C:\Program Files (x86)\wapt\wapt-get.ini

config files are located in C:\Program Files (x86)\wapt\conf.d

* config_base
```

35.2.7 wapt-get list-available-config

La commande **wapt-get list-available-config** liste tous les fichiers de configuration dynamique *json* disponibles sur le Serveur WAPT et indique la commande pour les installer.

Lister les configurations disponibles nécessite que l'utilisateur soit authentifié.

```
C:\Users\administrator>wapt-get list-available-config
Server: https://srvwapt.mydomain.lan
Server UUID: 32464dd6-c261-11e8-87be-cee799b43a00
Server CABundle: 0

Waptserver https://srvwapt.mydomain.lan Admin User () :admin
Waptserver Password: **************

default_config : wapt-get add-config-from-url https://srvwapt.mydomain.lan/wapt/conf.d/default_
↪config_55863a6b54a47255097b6403731b36de716fc7ee9ec824bffad36d5fdc49b6b5.json
     Server: https://srvwapt.mydomain.lan
     Repo: https://srvwapt.mydomain.lan/wapt

default : wapt-get add-config-from-url https://srvwapt.mydomain.lan/wapt/conf.d/default_
↪91ab2cd1901b5e36214224229c3461e49e65f7b065ea6b0eb16bd83c7fcdda57.json
     Server: https://srvwapt.mydomain.lan
     Repo: https://srvwapt.mydomain.lan/wapt

mac_config : wapt-get add-config-from-url https://srvwapt.mydomain.lan/wapt/conf.d/mac_config_
↪2720657c276cbc0ee14734e68fbd0fadc4dea3171625406e10cd9828631e5c72.json
     Server: https://srvwapt.mydomain.lan
     Repo: https://srvwapt.mydomain.lan/wapt
```

35.2.8 wapt-get clean

La commande **wapt-get clean** supprime les paquets du dossier cache.

La commande est lancée après chaque **wapt-get upgrade** pour économiser de l'espace disque.

La commande `wapt-get clean` renvoie :

```
Using config file: C:\Program Files (x86)\wapt\wapt-get.ini
Removed files:
C:\Program Files (x86)\wapt\cache\tis-mumble_1.2.3-1_all.wapt
C:\Program Files (x86)\\wapt\cache\tis-vlc_1.2.3-2_all.wapt
```

35.2.9 wapt-get upgradedb

La commande `wapt-get upgradedb` met à jour le schéma de la base de données WAPT locale si nécessaire.

La commande `wapt-get upgradedb` renvoie :

```
Using config file: C:\Program Files (x86)\wapt\wapt-get.ini
WARNING upgrade db aborted: current structure version 20210420 is newer or equal to requested␣
↪structure version 20210420
No database upgrade required, current 20210420, required 20210420
```

35.2.10 wapt-get add-upgrade-shutdown - wapt-get remove-upgrade-shutdown

Ces 2 commandes modifient le **:fichier:`C:\\Windows\\System32\\GroupPolicy\\Machine\\Scripts\\scripts.ini`** sur les appareils Windows.

- La commande `wapt-get add-upgrade-shutdown` ajoute un objet de stratégie de sécurité locale **waptexit**, permettant l'exécution de **waptexit** à l'arrêt du système.

La commande `wapt-get add-upgrade-shutdown` renvoie :

```
Using config file: C:\Program Files (x86)\wapt\wapt-get.ini
0
```

Le `scripts.ini` contient :

```
[Shutdown]
0CmdLine = C:\Program Files (x86)\wapt\waptexit.exe
0Parameters =
```

- La commande `wapt-get remove-upgrade-shutdown` supprime l'objet de politique de sécurité locale **waptexit**, désactivant l'exécution de **waptexit** pendant l'arrêt du système.

La commande `wapt-get add-upgrade-shutdown` renvoie :

```
Using config file: C:\Program Files (x86)\wapt\wapt-get.ini
0
```

Le `scripts.ini` contient :

```
[Shutdown]
```

35.2.11 wapt-get register

La commande **wapt-get register <description>** envoie l'inventaire du matériel et des logiciels de l'ordinateur au Serveur WAPT.

Le paramètre <description> est facultatif.

Indication: Une description peut être passée comme argument à la commande **wapt-get register**, la description sera affichée dans la console WAPT dans la colonne *description*.

Vous pouvez bénéficier de WAPT pour améliorer votre gestion informatique en affectant une étiquette d'inventaire comme description pour vos machines par exemple.

Note: Si la machine est déjà enregistrée, le réenregistrement de la machine à l'aide d'une description met à jour les informations enregistrées.

La commande `wapt-get register "John Doe PC"` renvoie :

```
Using config file: C:\Program Files (x86)\wapt\wapt-get.ini
Registering host against server: https://srvwapt.mydomain.lan
Host correctly registered against server https://srvwapt.mydomain.lan.
```

35.2.12 wapt-get register

La commande `wapt-get unregister` supprime l'inventaire matériel et logiciel de la machine du Serveur WAPT.

La commande `wapt-get list` renvoie :

```
Using config file: C:\Program Files (x86)\wapt\wapt-get.ini
Unregistering host from server: https://srvwapt.mydomain.lan
Please get login for api/v3/hosts_delete:admin
Password:
Host correctly unregistered against server https://srvwapt.mydomain.lan.
```

35.2.13 wapt-get inventory

La commande **wapt-get inventory** affiche les informations d'inventaire de la machine au format *json*.

La commande `wapt-get inventory` renvoie (en partie) :

```
Using config file: C:\Program Files (x86)\wapt\wapt-get.ini
{
  "host_info": {
    "description": "John Doe PC",
```

(suite sur la page suivante)

(suite de la page précédente)


```
    "system_manufacturer": "Xen",
    "system_productname": "HVM domU",
    "computer_name": "Documentation",
    "computer_fqdn": "Documentation.srvwapt.mydomain.lan",
    "dnsdomain": "mydomain.lan",
    "workgroup_name": "Documentation",
    "domain_name": null,
    "domain_controller": null,
    "domain_controller_address": null,
    "domain_info_source": "history",
    "networking": [
    {
      "iface": "{085AB96368A-05A3B96-43EC-B773-0C0BB96794D9}",
      "mac": "a2:1d:6e:fc:8d:e6",
      "addr": [
      {
        "addr": "192.168.0.1",
        "netmask": "255.255.255.0",
        "broadcast": "192.168.0.255",
        "connected": true
      },
      {
        "addr": "fe80::2437:567f:79c8:f964",
        "netmask": "ffff:ffff:ffff:ffff::/64",
        "broadcast": "fe80::ffff:ffff:ffff:ffff%3",
        "connected": true
      }
      ]
    }
    ],
    "gateways": [
    "192.168.0.254"
    ],
    "dns_servers": [
    "192.168.0.11"
    ],
    "connected_ips": [
    "192.168.0.1",
    "fe80::2437:567f:79c8:f964"
    ],
    "mac": [
    "a2:fc:1d:6e:8d:e6"
    ],
...
```

35.2.14 wapt-get update-status

La commande **wapt-get update-status** envoie l'état actuel de la machine au Serveur WAPT.

Note: Si un composant matériel a été modifié sur l'ordinateur, le `update-status` ne renvoie pas cette information au serveur d'inventaire WAPT.

Pour ce faire, la commande à utiliser est `inventory`.

La commande `wapt-get update-status` renvoie :

```
Using config file: C:\Program Files (x86)\wapt\wapt-get.ini
Updated host status correctly sent to the WAPT Server https://srvwapt.mydomain.lan. {'success':␣
↪True, 'msg': 'update_host', 'result': {'uuid': 'B8D346E7-DDDB-0013-5A8A-425CF3B6199E', 'computer_
↪fqdn': 'documentation.mydomain.lan', 'status_hashes': {'dmi':
↪'124b8bcef5b690afea7cf8001351a22132885123', 'wmi': 'ae5dbb5627b7b3a5a31d5914a9dbf48b85b133da',
↪'host_info': 'e737a82da15fbe9cae88ba9b4a9662a73657d959', 'audit_data': None, 'wapt_status':
↪'bcb76ad07cf1b6f814082ec5a58c4fee0364a640', 'audit_status':
↪'c34adb535c711b59d4408f00f77b7392687d7e56', 'host_metrics':
↪'9fc68bd98c82e0e9bece0ce3afaeeb63a3ed1db1', 'waptwua_status':
↪'4f9dcf0af339ce28d7354283fd4e6bdaf17b85c8', 'waptwua_updates':
↪'c5cf38908fc549f499ade5b17ce221ff0ced377f', 'wuauserv_status':
↪'7c30215c3c34566e5b0c69c9e1dbfe3e6117b837', 'host_capabilities':
↪'c31286122a213f3bb313531541582bb2ba1d0a81', 'installed_packages':
↪'3279f3bf4d5ed5086b198fa94a6a6f422f519ab3', 'last_update_status':
↪'347c5a8c01e182f1e03e5c9d0fe07dd87ab79153', 'installed_softwares':
↪'d582a6f7325af35eae17cb7ecdca59ef0d137dda', 'authorized_certificates':
↪'2974f9535f813fc454b735193c31828b132a6ba0', 'waptwua_updates_localstatus':
↪'c5cf38908fc549f499ade5b17ce221ff0ced377f'}, 'server_uuid': '82295c4d-4944-11ec-bac6-a25b5d7da3d5
↪'}, 'request_time': 0.046843767166137695}
```

35.2.15 wapt-get setlocalpassword

La commande **wapt-get setlocalpassword** permet de définir un mot de passe local pour l'installation des paquets WAPT.

La commande `wapt-get setlocalpassword` renvoie :

```
Local password:
Confirm password:
Local auth password set successfully
```

35.2.16 wapt-get reset-uuid

La commande **wapt-get reset-uuid** récupère le *UUID* de la machine à partir du BIOS et l'envoie au Serveur WAPT.

La commande `wapt-get reset-uuid` renvoie :

```
Using config file: C:\Program Files (x86)\wapt\wapt-get.ini
New UUID: B0F23D44-86CB-CEFE-A8D6-FB8E3343FE7F
```

35.2.17 wapt-get generate-uuid

La commande **wapt-get generate-uuid** génère un *UUID* aléatoire pour la machine et le renvoie au Serveur WAPT.

La **wapt-get generate-uuid** est utile s'il existe des *bugs de BIOS* avec certaines machines de la flotte.

La commande `wapt-get generate-uuid` renvoie :

```
Using config file: C:\Program Files (x86)\wapt\wapt-get.ini
New UUID: RND-0279A1F4-BBBE-43AE-A591-F82652E0104B
```

Note: Tous les UUID générés aléatoirement commencent par *RND-*.

35.2.18 wapt-get get-server-certificate

La commande **wapt-get get-server-certificate** télécharge le certificat SSL du Serveur WAPT afin que l'Agent WAPT puisse établir une connexion HTTPS sécurisée avec le Serveur WAPT.

Le certificat téléchargé est déposé dans `<wapt>\ssl\server`.

La commande `wapt-get get-server-certificate` renvoie :

```
Server certificate written to C:\Program Files (x86)\wapt\ssl\server\srvwapt.mydomain.lan.crt
```

35.2.19 wapt-get enable-check-certificate

La commande `wapt-get enable-check-certificate` télécharge le certificat SSL du serveur WAPT et active la communication sécurisée avec le serveur.

La commande **wapt-get enable-check-certificate** est utilisée pour *activer la vérification du certificat SSL / TLS*.

La commande `wapt-get enable-check-certificate` renvoie :

```
Using config file: C:\Program Files (x86)\wapt\wapt-get.ini
Server certificate : C:\Program Files (x86)\wapt\ssl\server\template-auto.test.lan.crt
Certificate CN: template-auto.test.lan
Pining certificate C:\Program Files (x86)\wapt\ssl\server\template-auto.test.lan.crt
```

35.2.20 wapt-get upgrade

La commande **wapt-get check-upgrades** indique l'état de la mise à jour / mise à niveau la plus récente pour la machine.

La commande **wapt-get check-upgrades** renvoie :

```
Using config file: C:\Program Files (x86)\wapt\wapt-get.ini
{
  "running_tasks": [],
  "errors": [],
  "date": "2022-01-06T14:09:26.554391",
  "upgrades": [
    "tis-notepadplusplus(=8.2-10)"
  ],
  "pending": {
    "install": [],
    "upgrade": [
    "tis-notepadplusplus(=8.2-10)"
    ],
    "additional": [],
    "remove": [],
    "immediate_installs": []
  }
}
```

35.2.21 wapt-get duplicate

La commande **wapt-get add-licence** ajoute une licence WAPT sur le Serveur WAPT.

La commande **wapt-get add-licence** renvoie :

```
Using config file C:\Program Files (x86)\wapt\wapt-get.ini
Server: https://srvwapt.mydomain.lan
Server UUID: 82295c4d-4944-11ec-bac6-a25b5d7da3d5
Server CABundle: 0

{"licence_nr":"6f011e23-cb70-40a4-b340-0d18ae1e2f02","product":"WAPT Enterprise","features":["full
↪"],"licenced_to":"documentation","domain":"","contact_email":"documentation@tranquil.it","count":
↪"10","valid_from":"2021-06-14T00:00:0","valid_until":"2022-01-12T00:00:0","renewal_url":null,
↪"signed_attributes":["licence_nr","product","features","licenced_to","domain","contact_email",
↪"count","valid_from","valid_until","renewal_url","signed_attributes","signer","signature_date",
↪"signer_certificate","server_uuid"],"signer":"","signature_date":"2022-01-13T16:38:56","signer_
↪certificate":"-----BEGIN CERTIFICATE-----\nMIIEIjCCAwqgAwIBAgIUIOMdx8FmRdmCNTHxOfKecSp/
↪cAAwDQYJKoZIhvcNAQEL\nBQAwgZcxCzAJBgNVBAYTAkZSMSIwIAYDVQQHDBlTYWludCBTZWJhc3RpZW4gc3Vy\
↪nIExvaXJlMRwwGgYDVQQKDBNUcmFucXVpbCBJVCBTeXN0ZW1zMSAwHgYDVQQDDBdy\
↪nZWxpY2VuY2luZy50cmFucXVpbC5pdDEkMCIGCSqGSIb3DQEJARYVdGVjaG5pcXVl\
↪nQHRyYW5xdWlsLml0MB4XDTIxMDYwODE0MTQ0MVoXDTMxMDYwNjE0MTQ0MVowgZcx\
↪nCzAJBgNVBAYpk6dZrIrw9Kb5hee+1EgqEbudCBTZWJhc3RpZW4gc3VyIExvaXJl\
↪nMRwwGgYDVQQKDBNUcmFucXVpbCBJVCBTeXN0ZW1zMSAwHgYDVQQDDBdyZWxpY2Vu\
↪nY2luZy50cmFucXVpbC5pdDEkMCIGCSqGSIb3DQEJARYVdGVjaG5pcXVlQHRyYW5x\
↪ndWlsLml0MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAzT43W8OhWXAe\nhDB+IWwQm9IGGdR0VY/klKcSheo/
↪8jGlNziyH6BANhjFKYNX9UtQ+ghzv6BGfSTH\nyua1aXEQM89sSKFOoJztoD1L9FZtuWQb/vfLWkisP8fRPvH4B/
↪tYG+5nchGa6+6r\nqGSGSpWcnj6CovgQR01ATUuHN3NV1N7q48hBT/ZT9R5U3sEi+hNK4eRIeZ220Pzm\
↪nDoNGkVKlEiczgXuM77ezYp8UWvpk6dZrIrw9Kb5hee+1EgqEbgVmdARoaOPGTK8h\
↪n8VW+milWsl4TEY19kxXWvva+M6wX00ipJ2LxEiu5+dlOok9E8i40
↪6S3C4twIDAQABo2QwYjAPBgNVHRMBAf8EBTADAQH/MCAGA1UdJQEB/wQWMBQG\
↪nCCsGAQUFBwMDBggrBgEFBQcDAjAdBgNVHQ4EFgQUpRT6Co2uoWZMCwP7FKiF73+j\nfAEwDgYDVR0PAQH/
↪BAQDAgHWMA0GCSqGSIb3DQEBCwUAA4IBAQAdXX5IkpuH/Gek\nPPHC4KvE/
↪6GsU0kqLI1w5ML5pbF1zyCCL0nm4f8w2JJIJ2Ycdb4QVD27kJqgZcH1\nniYQ3RCIh6aasS8qpCOf90KkpvKMJiyk/
```

(suite sur la page suivante)

(suite de la page précédente)

```
Login to server api/v3/licences
Waptserver https://srvwapt.mydomain.lan Admin User () :admin
Waptserver Password: ************
Licence properly activated on server
```

35.2.22 wapt-get clean

La commande **wapt-get check-licences** montre les licences enregistrées sur le Serveur WAPT.

La commande **wapt-get check-licences** renvoie :

```
Using config file C:\Program Files (x86)\wapt\wapt-get.ini
Server: https://srvwapt.mydomain.lan
Server UUID: 36bf01bc-c8f5-11eb-bf04-36127be97253
Server CABundle: 0

Total licences count: 10
Licenced to: documentation

Valid Nr:b7b6e537-3cb7-4d9a-3cb7-2448020e2e51 Count:10 From:2022-01-13T00:00:0 Expire:2023-01-
↪12T23:59:0 Server:36bf01bc-c8f5-11eb-bf04-36127be97253 Licencee:documentation
```

35.2.23 wapt-get download

La commande **wapt-get dnsdebug** affiche les données de configuration réseau de la machine, notamment les données locales DNS (Domain Name Service) liées à WAPT.

La commande **wapt-get dnsdebug** renvoie :

```
DNS Server : dns.mydomain.lan
DNS Domain : mydomain.lan
Main repo url: https://srvwapt.mydomain.lan/wapt
wapt SRV: []
waptserver SRV: []
CNAME: []
```

35.3 Utilisation de la ligne de commande pour la configuration de la session utilisateur

35.3.1 wapt-get session-setup

La commande `wapt-get session-setup <nom du paquet> [<ALL>]` lance les personnalisations de niveau utilisateur des paquets WAPT installés.

La commande **wapt-get session-setup** exécute la fonction *def session_setup()* définie dans le fichier `setup.py` du paquet WAPT si la fonction existe.

Note: L'argument *ALL* lancera `session-setup` pour tous les paquets WAPT installés.

La commande `wapt-get session-setup ALL` renvoie :

```
Configuring tis-7zip ... No session-setup. Done
Configuring tis-ccleaner ... Already installed. Done
Configuring tis-vlc ... No session-setup. Done
Configuring mdl-tightvnc ... No session-setup. Done
Configuring tis-brackets ... No session-setup. Done
Configuring mdl-firefox-esr ... No session-setup. Done
Configuring tis-paint.net ... No session-setup. Done
```

35.4 Utilisation de la ligne de commande pour créer des paquets WAPT

35.4.1 wapt-get list-registry

La commande `wapt-get list` liste les paquets WAPT qui sont installés sur l'ordinateur.

La commande peut prendre un argument insensible à la casse pour rechercher le mot-clé spécifié.

Les informations retournées sont :

Information	Définition	Disponible sur Windows	Disponible sur Linux	Disponible sur macOS
`UninstallKey`	Recherche l'identifiant de la clé de désinstallation dans la ruche du registre.	✅	❌	❌
`Software`	Recherche le nom du logiciel dans la ruche du registre.	✅	✅	✅
`Version`	Recherche la version du logiciel dans la ruche du registre.	✅	✅	✅
`UninstallString`	Recherche la chaîne de désinstallation du logiciel dans la ruche du registre.	✅	❌	❌

Note:

- Sous Windows, WAPT effectue des recherches dans deux emplacements du registre :
  - `ComputerHKEY_LOCAL_MACHINESOFTWAREMicrosoftWindowsCurrentVersionUninstall`.
  - `ComputerHKEY_LOCAL_MACHINESOFTWAREWOW6432NodeMicrosoftWindowsCurrentVersionUninstall`.
- Sous Linux, WAPT effectue des recherches en utilisant `Applications`.
- Sur macOS, WAPT recherche dans `/var/lib/dpkg/info/`.

La sortie de **`wapt-get list-registry`** est un tableau listant les *clés de désinstallation* pour chaque logiciel correspondant au terme recherché.

La commande **`wapt-get list-registry firefox`** renvoie (sous Windows) :

```
Using config file: C:\Program Files (x86)\wapt\wapt-get.ini
UninstallKey                          Software                                Version          ␣
↪Uninstallstring
-------------------------------------------------------------------------------------------------
↪-----------------------------------------------------
Mozilla Firefox 45.5.0 ESR (x64 fr)    Mozilla Firefox 45.5.0 ESR (x64 fr)    45.5.0
↪"C:\Program Files\Mozilla Firefox\uninstall\helper.exe"
```

35.4.2 wapt-get sources

La commande **wapt-get sources <nom du paquet>** télécharge les sources depuis un dépôt de gestion du code source comme Git ou SVN.

La commande `wapt-get sources tis-firefox` ne renvoie rien ;

35.4.3 wapt-get make-template

Avertissement: Cette méthode est dépréciée, utilisez plutôt la *console WAPT pour créer un modèle de packaging*.

La commande **wapt-get make-template <installer-path> [<packagename> [<source directoryname>]]** permet de créer un modèle de paquet à partir d'un installateur *msi* ou *exe*.

La commande **wapt-get make-template C:\Users\User\Downloads\tightvnc.msi tis-tightvnc** renvoie :

```
Using config file: C:\Users\Documentation\AppData\Local\waptconsole\waptconsole.ini
Template created. You can build the WAPT package by launching
  C:\Program Files (x86)\wapt\wapt-get.exe build-package C:\waptdev\tis-tightvnc-wapt
You can build and upload the WAPT package by launching
  C:\Program Files (x86)\wapt\wapt-get.exe build-upload C:\waptdev\tis-tightvnc-wapt
```

Indication:

- Si vous avez préalablement installé le paquet tis-waptdev sur votre ordinateur de développement, l'éditeur **PyScripter** se lancera automatiquement et ouvrira le paquet en mode développement.

35.4.4 wapt-get make-host-template

Avertissement: Cette méthode est principalement destinée aux scripts, en général les paquets de la machine sont créés automatiquement avec la console WAPT.

La commande `wapt-get make-host-template <nommachine> [[<paquet>,<paquet>,...] [répertoire]]` crée un paquetage hôte WAPT vide à partir d'un modèle.

La commande `wapt-get make-host-template host01.mydomain.lan` renvoie :

```
Using config file: C:\Users\Documentation\AppData\Local\waptconsole\waptconsole.ini
Template created. You can build the WAPT package by launching
  C:\Program Files (x86)\wapt\wapt-get.exe build-package C:\waptdev\host01.mydomain.lan-wapt
You can build and upload the WAPT package by launching
  C:\Program Files (x86)\wapt\wapt-get.exe build-upload C:\waptdev\host01.mydomain.lan-wapt
```

35.4.5 wapt-get make-group-template

Avertissement: Cette méthode ne doit être utilisée que si vous ne pouvez pas utiliser la Console WAPT pour créer un paquet.

La commande **wapt-get make-group-template <nom du groupe>** crée un paquet WAPT *group* vide à partir d'un modèle.

La commande `wapt-get make-group-template documentation` renvoie :

```
Template created. You can build the WAPT package by launching
  C:\Program Files (x86)\wapt\wapt-get.exe build-package C:\waptdev\accounting-wapt
You can build and upload the WAPT package by launching
  C:\Program Files (x86)\wapt\wapt-get.exe build-upload C:\waptdev\accounting-wapt
```

35.4.6 wapt-get build-package

La commande `wapt-get build-package <chemin vers le répertoire package>` construit un package WAPT et le signe avec la clé privée associée au `personal_certificate_path` défini dans le `waptconsole.ini`.

Note: Le chemin vers la clé privée, le préfixe par défaut et le chemin de développement par défaut **Doivent** être correctement définis dans le fichier `wapt-get.ini`.

La commande **wapt-get build-package c:waptdevtis-dropbox** renvoie :

```
Using config file: C:\Users\documentation\AppData\Local\waptconsole\waptconsole.ini
Building packages 1 packages
Personal certificate is documentation
Please type the password to decrypt the private key C:\Users\documentation\private\documentation.pem
Password:
Private key is <SSLPrivateKey 'C:\\Users\\documentation\\private\\documentation.pem'>
Personal certificate is SSLCertificate cn=documentation
Private key is <SSLPrivateKey 'C:\\Users\\documentation\\private\\documentation.pem'>
Building c:\waptdev\tis-dropbox
Signing c:\waptdev\tis-dropbox with key <SSLPrivateKey 'C:\\Users\\documentation\\private\\
↪documentation.pem'> and certificate documentation (C:\Users\documentation\private\documentation.
↪crt)
Package c:\waptdev\tis-dropbox signed : signature : BN7j6lwloY...Iu9QVulA=
...done building. Package filename c:\waptdev\tis-dropbox_104.4.175-7_windows_
↪0f4137ed1502b5045d6083aa258b5c42_5.1_PROD.wapt
1 packages successfully built
0 packages failed
```

(suite sur la page suivante)

(suite de la page précédente)

```
You can upload to repository with
  C:\Program Files (x86)\wapt\wapt-get.exe upload-package "c:\waptdev\tis-dropbox_104.4.175-7_
↪windows_0f4137ed1502b5045d6083aa258b5c42_5.1_PROD.wapt"
```

Avertissement: Le nom du répertoire ne définit pas le nom du paquet, ni son préfixe, ces valeurs sont définies par le fichier `control`.

35.4.7 wapt-get sign-package

La commande `wapt-get sign-package <chemin vers le paquet>` signe un paquet avec la clé privée associée à `personal_certificate_path` définie dans le `waptconsole.ini`.

Attention: **`wapt-get sign-package`** ne renomme pas le paquet WAPT avec le préfixe de l'*Organization*.

La commande `wapt-get sign-package c:\waptdev\tis-dropbox` renvoie :

```
Using config file: C:\Users\documentation\AppData\Local\waptconsole\waptconsole.ini
Signing packages c:\waptdev\tis-dropbox
Personal certificate is SSLCertificate cn=documentation
Please type the password to decrypt the private key C:\Users\documentation\private\documentation.pem
Password:
Private key is <SSLPrivateKey 'C:\\Users\\documentation\\private\\documentation.pem'>
Signing c:\waptdev\tis-dropbox
  OK: Package c:\waptdev\tis-dropbox signed : signature : b'nJYfYswDWi'...b'v790D7uA='
```

35.4.8 wapt-get build-upload

La commande `wapt-get build-upload <chemin vers le paquet>` construit et télécharge un paquet WAPT sur le dépôt WAPT principal.

Indication: En passant l'argument *-i* à **`wapt-get build-upload`**, le numéro de version du paquet WAPT est incrémenté avant que le paquet ne soit téléchargé, afin d'éviter de devoir modifier manuellement le fichier :*control*.

La commande `wapt-get -i build-upload c:\waptdev\tis-dropbox` renvoie :

```
Using config file: C:\Users\documentation\AppData\Local\waptconsole\waptconsole.ini
Building packages 1 packages
Personal certificate is documentation
Please type the password to decrypt the private key C:\Users\documentation\private\documentation.pem
Password:
Private key is <SSLPrivateKey 'C:\\Users\\documentation\\private\\documentation.pem'>
Personal certificate is SSLCertificate cn=documentation
Private key is <SSLPrivateKey 'C:\\Users\\documentation\\private\\documentation.pem'>
```

(suite sur la page suivante)

(suite de la page précédente)

```
Building c:\waptdev\tis-dropbox
Signing c:\waptdev\tis-dropbox with key <SSLPrivateKey 'C:\\Users\\documentation\\private\\
↪documentation.pem'> and certificate documentation (C:\Users\documentation\private\documentation.
↪crt)
Package c:\waptdev\tis-dropbox signed : signature : s9FOLFQvYw...c9T3Hv1A=
...done building. Package filename c:\waptdev\tis-dropbox_104.4.175-7_windows_
↪0f4137ed1502b5045d6083aa258b5c42_5.1_PROD.wapt
1 packages successfully built
0 packages failed
Building and uploading packages to https://srvwapt.mydomain.lan
Please get login for https://srvwapt.mydomain.lan/api/v3/upload_xxx:admin
Password:
c:\waptdev\tis-dropbox_104.4.175-7_windows_0f4137ed1502b5045d6083aa258b5c42_5.1_PROD.
↪wapt[==============================] 126459984/126459984 - 00:00:40
Package uploaded successfully: 1 Packages uploaded, 0 errors
```

35.4.9 wapt-get duplicate

La commande **wapt-get duplicate <source_package> <duplicated_package>** duplique un paquet téléchargé depuis le dépôt et l'ouvre en tant que projet en utilisant l'"IDE (Integrated Development Environment) qui a été spécifié dans la *configuration de la Console WAPT*.

Avertissement: N'utilisez pas cette commande pour dupliquer un paquet *host*.

Table1: Arguments autorisés lors de la duplication d'un paquet WAPT.

Arguments	Définition	Requis
<directory> ou <source_package>	Définit le chemin de répertoire vers le paquet WAPT ou le nom d'un paquet spécifique ayant une extension de fichier *.wapt*.	✓
<duplicated_package>	Définit le nom du nouveau paquet.	✓
<duplicated_package_version>	Change la version du paquet dans le fichier `control`. Si la version n'est pas définie, la même version est dupliquée.	✗
<duplicated_package_target_directory>	Définit le chemin d'accès au répertoire cible du paquet dupliqué. Si le répertoire cible n'est pas défini, le paquet dupliqué sera déposé dans le même répertoire que le paquet source.	✗

La commande **wapt-get duplicate tis-firefox tis-firefox-custom** renvoie :

```
Package duplicated. You can build the new WAPT package by launching
  C:\Program Files (x86)\wapt\wapt-get.exe build-package C:\waptdev\tis-firefox-custom-wapt
You can build and upload the new WAPT package by launching
  C:\Program Files (x86)\wapt\wapt-get.exe build-upload C:\waptdev\tis-firefox-custom-wapt
```

35.4.10 wapt-get edit

Avertissement: Cette méthode ne doit être utilisée que si vous ne pouvez pas utiliser la Console WAPT pour créer un paquet.

La commande **wapt-get edit <nom du paquet>** télécharge et ouvre le paquet dans un IDE pour le modifier.

La commande prend un argument. Cet argument est le nom paquet ou une liste de noms de paquets avec le préfixe du dépôt.

La commande **wapt-get edit tis-firefox** renvoie :

```
Using config file: C:\Users\documentation\AppData\Local\waptconsole\waptconsole.ini
https://srvwapt.mydomain.lan/wapt/tis-firefox_94.0.1-106_x64_windows_
↪0f4137ed1502b5045d6083aa258b5c42_6.1_PROD_en_f5335369ad5805e8dbc1f8ae99b2258e.wapt : 1629411 /␣
↪54313787 (3%) (2686 KB/s)
https://srvwapt.mydomain.lan/wapt/tis-firefox_94.0.1-106_x64_windows_
↪0f4137ed1502b5045d6083aa258b5c42_6.1_PROD_en_f5335369ad5805e8dbc1f8ae99b2258e.wapt : 8147055 /␣
↪54313787 (15%) (5679 KB/s)
https://srvwapt.mydomain.lan/wapt/tis-firefox_94.0.1-106_x64_windows_
↪0f4137ed1502b5045d6083aa258b5c42_6.1_PROD_en_f5335369ad5805e8dbc1f8ae99b2258e.wapt : 15207836 /␣
↪54313787 (28%) (7367 KB/s)
https://srvwapt.mydomain.lan/wapt/tis-firefox_94.0.1-106_x64_windows_
↪0f4137ed1502b5045d6083aa258b5c42_6.1_PROD_en_f5335369ad5805e8dbc1f8ae99b2258e.wapt : 19552932 /␣
↪54313787 (36%) (7249 KB/s)
https://srvwapt.mydomain.lan/wapt/tis-firefox_94.0.1-106_x64_windows_
↪0f4137ed1502b5045d6083aa258b5c42_6.1_PROD_en_f5335369ad5805e8dbc1f8ae99b2258e.wapt : 24984302 /␣
↪54313787 (46%) (7471 KB/s)
https://srvwapt.mydomain.lan/wapt/tis-firefox_94.0.1-106_x64_windows_
↪0f4137ed1502b5045d6083aa258b5c42_6.1_PROD_en_f5335369ad5805e8dbc1f8ae99b2258e.wapt : 29329398 /␣
↪54313787 (54%) (7143 KB/s)
https://srvwapt.mydomain.lan/wapt/tis-firefox_94.0.1-106_x64_windows_
↪0f4137ed1502b5045d6083aa258b5c42_6.1_PROD_en_f5335369ad5805e8dbc1f8ae99b2258e.wapt : 33674494 /␣
↪54313787 (62%) (6951 KB/s)
https://srvwapt.mydomain.lan/wapt/tis-firefox_94.0.1-106_x64_windows_
↪0f4137ed1502b5045d6083aa258b5c42_6.1_PROD_en_f5335369ad5805e8dbc1f8ae99b2258e.wapt : 40735275 /␣
↪54313787 (75%) (7534 KB/s)
https://srvwapt.mydomain.lan/wapt/tis-firefox_94.0.1-106_x64_windows_
↪0f4137ed1502b5045d6083aa258b5c42_6.1_PROD_en_f5335369ad5805e8dbc1f8ae99b2258e.wapt : 45623508 /␣
↪54313787 (84%) (7326 KB/s)
https://srvwapt.mydomain.lan/wapt/tis-firefox_94.0.1-106_x64_windows_
↪0f4137ed1502b5045d6083aa258b5c42_6.1_PROD_en_f5335369ad5805e8dbc1f8ae99b2258e.wapt : 53227426 /␣
↪54313787 (98%) (7603 KB/s)
https://srvwapt.mydomain.lan/wapt/tis-firefox_94.0.1-106_x64_windows_
↪0f4137ed1502b5045d6083aa258b5c42_6.1_PROD_en_f5335369ad5805e8dbc1f8ae99b2258e.wapt : 54313787 /␣
↪54313787 (100%) (7663 KB/s)
Package edited. You can build and upload the new WAPT package by launching
```

Indication:

- Si vous avez préalablement installé le paquet tis-waptdev sur votre ordinateur de développement, l'éditeur **PyScripter** se lancera automatiquement et ouvrira le paquet en mode développement.

- Vous pouvez éditer un paquet local en allant dans le dossier du paquet puis en tapant **wapt-get edit .**.
- Une autre méthode, vous pouvez éditer un paquet local à partir de son nom de répertoire ou du nom du paquet wapt, exemple **wapt-get edit tis-vlc.wapt**.

35.4.11 wapt-get edit-host

Avertissement: Cette méthode ne doit être utilisée que si vous ne pouvez pas utiliser la Console WAPT pour créer un paquet.

La commande **wapt-get edit-host <host FQDN>** édite un paquet WAPT *host*.

La commande **wapt-get edit-host RND-0279A1F4-BBBE-43AE-A591-F82652E0104B** renvoie :

```
Using config file: C:\Users\documentation\AppData\Local\waptconsole\waptconsole.ini
Package edited. You can build and upload the new WAPT package by launching
C:\Program Files (x86)\wapt\wapt-get.exe -i build-upload c:\waptdev\RND-0279A1F4-BBBE-43AE-A591-
↪F82652E0104B_0-wapt
```

35.4.12 wapt-get update-package-sources

La commande **wapt-get update-package-sources <chemin vers le paquet>** met à jour la fonction `def update_package()` dans le fichier :*setup.py*.

La commande **wapt-get update-package-sources tis-firefox** renvoie :

```
Using config file: C:\Users\documentation\AppData\Local\waptconsole\waptconsole.ini
Latis Mozilla Firefox version is: 95.0.2
Download URL is: https://download-installer.cdn.mozilla.net/pub/firefox/releases/95.0.2/win64/en-US/
↪Firefox%20Setup%2095.0.2.exe
Downloading: Firefox_Setup_95.0.2.exe
Firefox_Setup_95.0.2.exe[==============================] 54810424/54810424 - 00:00:07
Software version updated (from: 94.0.1 to: 95.0.2)
Packages updated :
c:\waptdev\tis-firefox_0-wapt
```

35.5 Utilisation des lignes de commande pour la gestion de WaptWUA

35.5.1 wapt-get clean

La **wapt-get waptwua-scan** analyse l'état des mises à jour de Windows par rapport aux règles actuelles et renvoie le résultat au Serveur WAPT.

La commande **wapt-get waptwua-scan** renvoie :

```
Using config file: C:\Program Files (x86)\wapt\wapt-get.ini
Ensure wuauserv Auto Update option is disabled
Scanning with windows updates rules:
{
"direct_download": false,
"default_allow": false,
"filter": "Type='Software' or Type='Driver'",
"download_scheduling": "7d",
"install_scheduling": null,
"install_delay": null,
"postboot_delay": "10m"
}
Downloading wsusscn2.cab file from https://srvwapt.mydomain.lan/waptwua/wsusscn2.cab
wsusscn2.cab[================================] 1024297844/1024297844 - 00:02:04
Windows updates rules have been changed
Looking for updates with filter: Type='Software' or Type='Driver'
  Connecting to local update searcher using offline wsusscn2 file...
  Offline Update searcher ready...
Waiting for WUA search to complete
Done searching
WUA Search completed !
Updates scan done.
Writing status in local wapt DB
Status: OK
(0, 0, 0)
None
re-enabling wuauserv previous state: 0
```

35.5.2 wapt-get download

La commande **wapt-get waptwua-download** analyse l'état de l'agent de mise à jour Windows par rapport aux règles actuelles, puis télécharge les kb manquants et enfin envoie le résultat au Serveur WAPT.

La commande **wapt-get waptwua-download** renvoie :

```
Using config file: C:\Program Files (x86)\wapt\wapt-get.ini
Ensure wuauserv Auto Update option is disabled
Downloading wsusscn2.cab file from https://srvwapt.mydomain.lan/waptwua/wsusscn2.cab
wsusscn2.cab[================================] 1024297844/1024297844 - 00:00:26
Start of install for all pending Windows updates
Scanning with params:
{
"direct_download": false,
"default_allow": false,
"filter": "Type='Software' or Type='Driver'",
"download_scheduling": "7d",
"install_scheduling": null,
"install_delay": null,
"postboot_delay": "10m"
}
```

(suite sur la page suivante)

(suite de la page précédente)

```
Scanning with windows updates rules:
{
"direct_download": false,
"default_allow": false,
"filter": "Type='Software' or Type='Driver'",
"download_scheduling": null,
"install_scheduling": null,
"install_delay": null,
"postboot_delay": "10m"
}
Bypassing scan, no change since last successful scan
Writing status in local wapt DB
Status: OK
{'downloaded': [], 'missing': []}
None
re-enabling wuauserv previous state: 0
```

35.5.3 wapt-get install

La commande **wapt-get waptwua-install** installe les mises à jour Windows en attente sur la machine.

La commande **wapt-get waptwua-install** renvoie :

```
Using config file: C:\Program Files (x86)\wapt\wapt-get.ini
Ensure wuauserv Auto Update option is disabled
Downloading wsusscn2.cab file from https://srvwapt.mydomain.lan/waptwua/wsusscn2.cab
wsusscn2.cab[==============================] 1024297844/1024297844 - 00:00:26
Start of install for all pending Windows updates
Scanning with params:
{
"direct_download": false,
"default_allow": false,
"filter": "Type='Software' or Type='Driver'",
"download_scheduling": "7d",
"install_scheduling": null,
"install_delay": null,
"postboot_delay": "10m"
}
Scanning with windows updates rules:
{
"allowed_products": null,
"allowed_classifications": null,
"allowed_severities": null,
"allowed_updates": null,
"forbidden_updates": null,
"allowed_kbs": null,
"forbidden_kbs": null,
"default_allow": false
}
```

(suite sur la page suivante)

(suite de la page précédente)

```
Looking for updates with filter: Type='Software' or Type='Driver'
  Connecting to local update searcher using offline wsusscn2 file...
  Offline Update searcher ready...
Waiting for WUA search to complete
Done searching
WUA Search completed !
Updates scan done.
Installed 07609d43-d518-4e77-856e-d1b316d1b8a8 : MSXML 6.0 RTM Security Update (925673)
Installed bb49cc19-8847-4986-aa93-5e905421e55a : Security Update for Microsoft Visual C++ 2005␣
↪Service Pack 1 Redistributable Package (KB2538242)
Installed 729a0dcb-df9e-4d02-b603-ed1aee074428 : Security Update for Microsoft Visual C++ 2008␣
↪Service Pack 1 Redistributable Package (KB2538243)
Installed 719584bc-2208-4bc9-a650-d3d6347eb32e : Security Update for Microsoft Visual C++ 2010␣
↪Service Pack 1 Redistributable Package (KB2565063)
Installed a8761130-35b6-41ce-8b67-2d35bb2d0846 : 2021-02 Cumulative Update for .NET Framework 3.5␣
↪and 4.8 for Windows 10, version 20H2 for x64 (KB4601050)
Installed 30f75e5d-2c46-42be-aef6-97ae730452be : 2021-07 Cumulative Update for Windows 10 Version␣
↪20H2 for x64-based Systems (KB5004945)
Installed 6e88be6e-d470-4e7e-9f36-01479049aadb : 2021-08 Servicing Stack Update for Windows 10␣
↪Version 20H2 for x64-based Systems (KB5005260)
Installed a15155a4-1299-41ff-9a39-28a33ce7cadd : 2021-12 .NET Core 3.1.22 Security Update for x64␣
↪Client (KB5009193)
Installed 38db0ad6-27f8-4bf9-ab2a-cffc4d7bc390 : Windows Malicious Software Removal Tool x64 - v5.
↪96 (KB890830)
Scanning with windows updates rules:
{
"direct_download": false,
"default_allow": false,
"filter": "Type='Software' or Type='Driver'",
"download_scheduling": "7d",
"install_scheduling": null,
"install_delay": null,
"postboot_delay": "10m"
}
Windows updates rules have been changed
Writing status in local wapt DB
Status: OK
[]
None
re-enabling wuauserv previous state: 2
```

35.5.4 wapt-get waptwua-status

La commande **wapt-get waptwua-status** renvoie l'état le plus récent de Windows Update pour la machine.

```
Using config file: C:\Program Files (x86)\wapt\wapt-get.ini
{'enabled': None,
'last_error': 'OperationalError: cannot rollback - no transaction is active',
'last_install_batch': [],
'last_install_date': None,
'last_install_reboot_required': None,
'last_install_result': None,
'last_scan_date': '2022-01-07T10:20:50.213644',
'last_scan_duration': 1490.500022649765,
'missing_downloads': [],
'rules_packages': [],
'status': 'SCANNING',
'wsusscn2cab_date': '2021-12-14T04:06:46'}
None
```

35.6 Utiliser la ligne de commande pour interagir avec les utilisateurs

35.6.1 wapt-get upgrade

La commande **wapt-get propose-upgrade** propose aux utilisateurs connectés de lancer les mises à niveau en attente.

La commande **wapt-get propose-upgrade** renvoie :

```
Using config file: C:\Program Files (x86)\wapt\wapt-get.ini
{'result': 1, 'summary': 'waptexit launched for 1 sessions'}
```

35.7 Utilisation des lignes de commande pour la configuration initiale

35.7.1 wapt-get register

La commande `wapt-get create-keycert [<options>]` crée une paire de clés RSA et un certificat X509.

Table2: Liste des options disponibles pour créer des certificats.

Option	Description	Valeur par défaut
`--CommonName`	Affiche le nom du certificat.	/
`--CommonName64`	Affiche le nom du certificat, encodé en base64 (si accents, espaces etc…).	/
`--CodeSigning` **|enterprise_feature||**	Définit si la paire certificat / clé sera autorisée à signer des paquets logiciels.	`0`
`--CA` **|enterprise_feature||**	Définit si le certificat / la paire de clés peut être utilisé(e) pour signer d'autres certificats (c'est-à-dire pour être autorisé(e) à se comporter comme une autorité de certification principale ou intermédiaire).	`0`
`--ClientAuth`	Définit une propriété (usage) du certificat.	- pour non installé
`--PrivateKeyPassword`	Définit le mot de passe pour déverrouiller la clé si `--NoPrivateKeyPassword` n'est pas utilisé.	Mot de passe généré aléatoirement
`--PrivateKeyPassword64` si `--PrivateKeyPassword` n'est pas utilisé	Définit le mot de passe pour déverrouiller la clé, encodé en base64 (si accents, espaces etc…).	Mot de passe généré aléatoirement
`--NoPrivateKeyPassword`	Définit la clé privée comme n'étant pas protégée par un mot de passe si `--PrivateKeyPassword` ou `--PrivateKeyPassword64` ne sont pas utilisés.	Vide
`-F`	Force l'écrasement du certificat existant.	/
`--Pays`	Définit le nom du pays du titulaire du certificat à enregistrer dans le certificat.	/
`--Localité`	Définit le nom de la ville du titulaire du certificat à enregistrer dans le certificat.	/
`--Organisation`	Définit le nom de l'organisation du titulaire du certificat à enregistrer dans le certificat.	/
`--OrgUnit`	Définit le nom de l'unité d'organisation (service) du titulaire du certificat à enregistrer dans le certificat.	/
`--Email`	Définit l'adresse e-mail du titulaire du certificat à enregistrer dans le certificat.	/
`--CAKeyFilename`	Définit le chemin vers la clé privée (`.pem`) d'une Autorité de Certification.	Paramètre `default_ca_key_path` dans `waptconsole.ini`
`--CACertFilename`	Définit le chemin vers le certificat public (`.crt`) d'une Autorité de Certification.	Paramètre `default_ca_cert_path` dans `waptconsole.ini`
`--CAKeyPassword`	Définit le mot de passe pour déverrouiller la clé d'une Autorité de Certification.	/
`-NoCAKeyPassword`	Définit la clé de l'Autorité de Certification comme n'étant pas protégée par un mot de passe.	/
`--BaseDir`	Définit le dossier dans lequel la clé privée et le certificat public seront déposés.	Répertoire `personal_certificate_path` dans `waptconsole.ini`
`-EnrollNewCert`	Copie le certificat dans `waptssl`	/
`-SetAsDefaultPersonalCert`	Définit le `personal_certificate_path` dans `waptconsole.ini`.	/

La commande `wapt-get clean` renvoie :

```
Using config file C:\Users\Administrator\AppData\Local\waptconsole\waptconsole.ini
BaseDir: C:\private\
Common name of certificate to create: documentation
Private Key Filename: C:\private\documentation.pem
Certificate Filename: C:\private\documentation.crt
New private key password: QR.-DVp6MPGW
```

Avertissement: Si `default_ca_key_path` et `default_ca_cert_path` sont définis dans `waptconsole.ini`, alors vous devez placer le certificat CA au même endroit sinon cette erreur apparaît :

```
wapt-get create-keycert
Using config file C:\Users\tisadmin\AppData\Local\waptconsole\waptconsole.ini
BaseDir: C:\Users\tisadmin\private\
Common name of certificate to create: CRT
Exception at 00483595: Exception:
CA Certificate C:\Program Files (x86)\wapt\ssl does not exist.
```

35.7.2 wapt-get build-package

La **`wapt-get build-waptagent [</ConfigFilename>]`** compile et télécharge un paquet **`waptagent.exe`** et un **`waptupgrade.exe`** en utilisant le paramètre `/ConfigFilename` pour spécifier le contenu du *fichier de configuration* `wapt-get.ini` des Agents WAPT.

Note: Par défaut, la commande utilise les éléments de configuration du fichier `waptconsole.ini` de la console WAPT.

La commande **`wapt-get build-waptagent`** renvoie :

```
Building customized waptagent.exe installer
................................................................................
Built C:\Users\documentation\AppData\Local\Temp\wapt20220107T12203700000.tmp\waptupgrade\waptagent.
↪exe
Private key Password for C:\Users\documentation\private\documentation.crt : ********
Building waptupgrade package
Waptserver https://srvwapt.mydomain.lan Admin User () :admin
Waptserver Password: ************
Uploading customized waptagent.exe installer
Uploading C:\Users\documentation\AppData\Local\Temp\wapt20220107T12203700000.tmp\waptupgrade\
↪waptagent.exe to waptserver https://srvwapt.mydomain.lan
OK
Uploading C:\Users\documentation\AppData\Local\Temp\wapt20220107T12203700000.tmp\tis-waptupgrade_2.
↪1.2.10605-0_all_PROD_all.wapt to waptserver https://srvwapt.mydomain.lan
OK : 1 Packages uploaded, 0 errors. Errors:
```

35.8 Utilisation des lignes de commande pour la gestion des référentiels

35.8.1 wapt-get upload-package

La commande **wapt-get upload-package <chemin vers le paquet>** télécharge un paquet sur le dépôt principal de WAPT.

La commande **wapt-get upload-package C:\waptdev\tis-tightvnc.wapt** renvoie :

```
Using config file: C:\Users\documentation\AppData\Local\waptconsole\waptconsole.ini
Uploading packages to https://srvwapt.mydomain.lan
Please get login for https://srvwapt.mydomain.lan/api/v3/upload_xxx:admin
Password:
c:\waptdev\tis-tightvnc.wapt[==============================] 54316019/54316019 - 00:00:17
OK : 1 Packages uploaded, 0 errors
```

35.8.2 wapt-get sign-package

Indication: Cette commande s'applique aux dépôts Windows ** UNIQUEMENT**.

La commande **wapt-get scan-packages <répertoire>** reconstruit un fichier `Packages` pour un dépôt de paquets WAPT.

La commande **wapt-get scan-packages C:waptwaptserverrepositorywapt** renvoie :

```
Using config file: C:\Program Files (x86)\wapt\wapt-get.ini
Packages filename: C:\wapt\waptserver\repository\wapt
Processed packages:
  C:\wapt\waptserver\repository\wapt\tis-firefox.wapt
  C:\wapt\waptserver\repository\wapt\tis-tightvnc.wapt
  C:\wapt\waptserver\repository\wapt\tis-7zip.wapt
Skipped packages:
```

35.8.3 wapt-get sign-package

Indication: Cette commande s'applique aux dépôts Linux ** UNIQUEMENT**.

La commande **wapt-scanpackages <répertoire>** reconstruit un fichier `Packages` pour un dépôt de paquets WAPT.

La commande **wapt-scanpackages /var/www/wapt/** ne renvoie rien.

35.8.4 Re-signer des paquets sur le Serveur WAPT en utilisant une ligne de commande

Use this method if re-signing from the WAPT console method does not complete successfully. These commands are **ONLY** available for WAPT Servers running Linux.

Avertissement: Avant d'utiliser cette méthode, assurez-vous que votre serveur WAPT est sûr et n'est pas sous le contrôle d'une entité tierce non autorisée.

- Copiez vos `.crt` et `.pem` dans `/tmp/` sur le serveur WAPT en utilisant **Winscp** ou un outil équivalent.
- Il est alors possible de re-signer tous les paquets en une seule fois sur le serveur WAPT avec les commandes suivantes.

```
wapt-signpackages -d /var/www/wapt-host -c /tmp/wapt_pub_key.crt -k /tmp/wapt_priv_key.pem -s
wapt-signpackages -d /var/www/wapt -c /tmp/wapt_pub_key.crt -k /tmp/wapt_priv_key.pem -s
wapt-scanpackages /var/www/wapt/
```

Si l'erreur **Access violation** apparaît, la raison en est que le paquet WAPT est trop volumineux.

Modifier le paquet et vérifier *cette procédure pour transférer un paquet volumineux*.

Danger: Supprimer les fichiers *.crt* et *.pem* de `/tmp/` sur le Serveur WAPT ou le serveur deviendra un bien sensible.

35.9 Utilisation de lignes de commande spéciales avec WAPT

Option	Définition
`--version`	Affiche le numéro de version de WAPT et quitte.
`-h | --help`	Affiche ce message d'aide et quitte.
`-c CONFIG | --config=CONFIG`	Définit le chemin d'un autre fichier comme `wapt-`
`-l LOGLEVEL | --loglevel=LOGLEVEL`	Définit le niveau des fichiers journaux en suivant
`-D | --direct`	Indique de ne pas utiliser le service http pour **upd**
`-S | --service`	Demande un utilisateur du Waptservice.
`-u | --update-packages`	Exécute **`wapt-get update`** avant la commande s
`-f | --force`	Force la commande.
`-p PARAMS | --params=PARAMS`	Configure les paramètres comme un objet JSon.
`-r WAPT_URL | -repo WAPT_URL | --repository=WAPT_URL`	Remplace l'URL du dépôt WAPT principal tel qu
`-y | --hide`	Masque la Console WAPT pendant l'exécution de
`-F FILTER_ON_HOST_CAP | --use-host-caps=FILTER_ON_HOST_CAP`	Filtre les paquets en fonction des capacités actuell
`-i | --inc-release`	Incrémente le numéro de version de la version lors
`-a UPDATE_SERVER_STATUS | --update-server-status=UPDATE_SERVER_STATUS`	Envoie le statut mis à jour de la machine (soft, pac
`keep-signature-date`	Conserve la date de signature du paquet actuel, et
`-s SECTION_FILTER | --sections=SECTION_FILTER`	Ajoute un filtre **section** à **`wapt-get search`**. La
`-o REDIRECT_OUTPUT | --output=REDIRECT_OUTPUT`	Redirige la sortie vers un fichier *.ini* dont le che
`-j | -json`	Bascule sur une sortie formatée en *json* dans le b
`-e ENCODING | --encoding=ENCODING`	Change l'encodage des caractères pour la sortie.
`-x EXCLUDES | --excludes=EXCLUDES`	Définit une liste de fichiers ou de répertoires sépar
`-k PERSONAL_CERTIFICATE_PATH | --certificate=PERSONAL_CERTIFICATE_PATH`	Définit le chemin vers le certificat PEM X509 po
`-w PRIVATE_KEY_PASSWD | --private-key-passwd=PRIVATE_KEY_PASSWD`	Définit le chemin d'accès au fichier contenant le

suite sur la page suivante

Table 3 – suite de la page précédente

Option	Définition
-U USER | --user=USER	Définit un utilisateur interactif.
-g USERGROUPS | --usergroups=USERGROUPS	Définit les groupes de l'utilisateur final comme u
-t MAX_TTL | --maxttl=MAX_TTL	Définit la durée maximale d'exécution en minute
-L LANGUAGE | --language=LANGUAGE	Remplace la locale pour l'installation des paquet
-m MD | --message-digest=MD	Définit le type de résumé de message pour **wapt**
-n | --newest-only	Renvoie uniquement la version la plus récente de
--locales=LOCALES	Remplace le filtre local des paquets lorsque vous
--maturity=MATURITY	Définit / modifie la maturité du paquet lors de la
--pin-server-cert	Épingle le certificat du serveur lors de l'enregistr
--wapt-server-url=SET_WAPTSERVER_URL	Définit l'url du Serveur WAPT lorsque le paramè
--wapt-repo-url=SET_WAPTREPO_URL	Définit l'url du dépôt WAPT lorsque le paramètr
--wapt-server-user=WAPT_SERVER_USER	Définit l'utilisateur autorisé à télécharger des pac
--wapt-server-passwd=WAPT_SERVER_PASSWD	Définit le mot de passe de l'utilisateur autorisé à
--log-to-windows-events	Enregistre les étapes dans le journal des événem
--use-gui	Force l'utilisation de GUI Helper même si elle n
--no-ide	Indique à WAPT de ne pas lancer l"IDE lors de l

CHAPTER 36

Configuration avancée de l'Agent WAPT

Le fichier de configuration `wapt-get.ini` définit le comportement de l'agent WAPT.

Table1: Emplacement du `wapt-get.ini` par le système

Système	Localisation
Windows	`C:\Program Files(x86)\wapt\wapt-get.ini`
Linux	`/opt/wapt/wapt-get.ini`
Mac OS	`/opt/wapt/wapt-get.ini`

La section `[global]` est obligatoire.

```
[global]
```

Après l'installation standard, la configuration par défaut est la suivante :

```
[global]
waptupdate_task_period=120
wapt_server=https://srvwapt.mydomain.lan
repo_url=https://srvwapt.mydomain.lan/wapt/
use_hostpackages=1
```

Tous les paramètres ne sont pas disponibles lors de la génération de l'agent. Il est possible de faire des changements dans `wapt-get.ini` manuellement ou en déployant un paquet WAPT avec les nouveaux paramètres de configuration.

Un exemple de paquet est disponible dans le dépôt Tranquil IT.

```
# -*- coding: utf-8 -*-
from setuphelpers import *

uninstallkey = []
```

(suite sur la page suivante)

(suite de la page précédente)

```
def install():

  print('Modify max_gpo_script_wait')
  inifile_writestring(WAPT.config_filename,'global','max_gpo_script_wait',180)

  print('Modify Preshutdowntimeout')
  inifile_writestring(WAPT.config_filename,'global','pre_shutdown_timeout',180)

  print('Disable Hyberboot')
  inifile_writestring(WAPT.config_filename,'global','hiberboot_enabled',0)

  print('Disable Notify User')
  inifile_writestring(WAPT.config_filename,'global','notify_user',0)

  print('Reload WAPT configuration')
  WAPT.reload_config_if_updated()
```

La définition de la fonction `inifile_writestring` est :

```
inifile_writestring(inifilename,section,key,value)
```

36.1 Description des sections disponibles

Table2: Description des sections disponibles pour l'agent WAPT

Section	Description
`[global]`	Options globales de l'agent WAPT.
`[wapt]`	Options du dépôt principal.
`[wapt-template]`	Options du dépôt externe.
`[wapt-host]`	Options pour le dépôt pour les paquets *host*.
`[waptwua]`	Options de l'agent WUA.
`[repo-sync]`	Pour synchroniser plusieurs dépôts.

Toutes les sections sont détaillées ci-dessous.

36.2 Description des options disponibles par section

36.2.1 [global]

Paramètres généraux

Table3: Description des options disponibles pour l'agent WAPT sur la section globale

Options / Valeur par défaut	Description	Exemple
allow_remote_reboot (défaut False)	Permet de redémarrer le ou les hôtes sélectionnés à distance à partir de la console WAPT.	allow_remote_reboot = True
allow_remote_shutdown (défaut False)	Permet d'arrêter le ou les hôtes sélectionnés à distance à partir de la console WAPT.	allow_remote_reboot = True
check_certificates_validity (défaut False)	Force la vérification de la date et de la CRL du certificat du paquet.	check_certificates_validity = True
dbpath (défaut WAPT root dir)\wapt\db\waptdb.sqlite)	Chemin d'accès au fichier de la base de données locale.	dbpath = C:\Program Files(x86\db\waptdb.sqlite
download_after_update_with_waptupdate_task_period (défaut True)	Définit si le téléchargement des paquets en attente doit être lancé après une mise à jour avec waptupdate_task_period.	download_after_update_with_v = False
host_organizational_unit_dn (défaut None)	Permet de forcer une unité organisationnelle sur l'agent WAPT (pratique pour attribuer un *fake OU* pour un PC hors-domaine). Assurez-vous qu'il respecte une casse cohérente (ne pas mélanger les « dc « s et les « DC « s, par exemple), que vous pouvez trouver dans la console (dans les champs DN/computer_ad_dn pour chaque hôte)	host_organizational_unit_dn = OU=TOTO,OU=TEST,DC=M
host_profiles (défaut None)	Permet de définir une liste de paquets WAPT que l'agent WAPT doit installer.	host_profiles = tis-firefox,tis-java
language (langue par défaut du client WAPT)	Force la langue par défaut pour l'interface graphique (pas pour le filtrage des paquets)	language = fr
locales (locale par défaut du client WAPT)	Permet de définir la liste des langues de l'agent WAPT pour pré-filtrer la liste des paquets visibles par l'agent WAPT (pour le filtrage des paquets). Le paramètre accepte plusieurs entrées (par exemple, locales=fr,en) ordonnées par préférence.	locales = en
log_to_windows_events (défaut False)	Envoie les journaux WAPT dans le journal des événements de Windows.	log_to_windows_events = True
loglevel (défaut warning)	Niveau de journalisation de l'agent WAPT. Les valeurs possibles sont : debug, info, warning, critical.	loglevel = critical
maturities = (default PROD)	Liste des maturités de paquets qui peuvent être visualisées et installées par l'agent WAPT. La valeur par défaut est PROD. Seules les valeurs DEV, PREPROD et PROD sont utilisées par Tranquil IT, cependant toute valeur peut être utilisée pour s'adapter à vos processus internes.	maturities = PROD, PREPROD
repo_url (défaut l'adresse du dépôt WAPT)	Adresse du dépôt principal de WAPT.	repo_url = https://srvwapt.mydomain.lan/wapt
repositories (défaut None)	Liste des dépôts activés, séparés par une virgule. Chaque valeur définit une section du fichier wapt-get.ini. Plus d'info *ici*.	repositories = dépôt1, dépôt2
send_usage_report (défaut True)	Permet à la console WAPT d'envoyer des statistiques anonymes à Tranquil IT. Mettre à 0 pour désactiver la télémétrie.	send_usage_report = True
service_auth_type (défaut system)	Comment fonctionne l'authentification du self-service. Les valeurs possibles sont : system, waptserver-ldap ou waptagent-ldap.	service_auth_type = waptserver-ldap
uninstall_allowed (défaut True)	S'il est possible ou non pour l'utilisateur de désinstaller des applications via le self-service.	uninstall_allowed = False
use_ad_groups (défaut False)	Pour l'utilisation de *paquets groupe* (par défaut False).	use_ad_groups = True
use_fqdn_as_uuid (défaut False)	Permet d'utiliser le FQDN plutôt que l'UUID du BIOS comme identifiant unique de la machine dans WAPT.	use_fqdn_as_uuid = True
use_hostpackages (défaut False)	Définit si *les paquets hôtes* doivent être utilisés. utiliser_hostpackages = False désactive les mises à jour implicites (paquets hôtes, paquets unités, paquets profils). C'est utile si vous voulez isoler un hôte et utiliser WAPT	use_hostpackages = True

Note:

- S'il n'y a pas d'attribut `repo_url`, un dépôt dans la section `[wapt]` devra être explicitement défini. Il devra être activé en l'ajoutant à l'attribut `repositories`.
- S'il n'y a pas d'attribut `wapt_server`, aucun serveur WAPT ne sera utilisé.

Paramètres du serveur WAPT

Ces options définissent le comportement de l'agent WAPT lors de la connexion au serveur WAPT.

Table4: Description des options disponibles pour l'agent WAPT dans la section [globale] pour la configuration du serveur

Options / Valeur par défaut	Description	Exemple
`public_certs_dir` (défaut `None`)	Dossier des certificats autorisés à vérifier la signature des paquets WAPT.	`public_certs_dir = C:\Program Files (x86)\wapt\ssl` (sous Windows) `public_certs_dir = /opt/wapt/ssl/` (sous Linux et MacOS)
`use_kerberos` (par défaut `False`)	Utiliser l'authentification kerberos pour l'enregistrement initial sur le serveur WAPT (par défaut False).	`use_kerberos = True`
`verify_cert` (défaut `False`)	Voir la documentation sur l'activation de la *vérification des certificats HTTPS*.	`verify_cert = True`
`wapt_server` (défaut `None`)	URL du serveur WAPT. Si cet attribut n'est pas présent, aucun serveur WAPT ne sera contacté.	:codewapt_server:`` = `https://srvwapt.mydomain.lan`
`wapt_server_timeout` (défaut `30`)	Délai de connexion HTTPS du serveur WAPT en secondes.	code:*wapt_server_timeout* = `10`

Paramètres pour le wapttray

Table5: Description des options disponibles pour l'agent WAPT sur la section globale pour le wapttray

Options / Valeur par défaut	Description	Exemple
:code:"allow_cancel_upgrade" (défaut `True`)	Empêche les utilisateurs d'annuler les mises à jour des paquets à l'arrêt du poste. Si désactivé, les utilisateurs ne seront pas en capacité d'annuler les mises à jour des paquets à l'arrêt du poste. Si cette valeur n'est pas renseignée, elle sera par défaut a **10**.	allow_cancel_upgrade = True
`hiberboot_enabled` (défaut `None`)	Désactive Hiberboot sur Windows 10 pour **`waptexit`**.	hiberboot_enabled = True
`max_gpo_script_wait` (défaut `None`)	Délai d'exécution des GPO à l'arrêt de l'ordinateur.	max_gpo_script_wait = 180
`pre_shutdown_timeout` (défaut `None`)	Délai d'attente pour les scripts à l'arrêt de l'ordinateur.	pre_shutdown_timeout = 180
:code:"upgrade_only_if_not_process_running" (défaut `False`)	Empêche la mise à niveau du logiciel si celui-ci est en cours d'exécution sur l'hôte (attribut *impacted_process* du packaging).	upgrade_only_if_not_proces = True
`upgrade_priorities` (défaut `None`)	Ne mettre à niveau que les paquets ayant une priorité spécifique.	upgrade_priorities = high
:code:"waptexit_countdown" (défaut `True`)	Délai (en secondes) avant le démarrage automatique des installations.	waptexit_countdown = 25

Paramètres pour l'authentification WAPT Self-Service et Waptservice

Table6: Description des options disponibles pour l'agent WAPT dans la section globale pour le self-service WAPT et l'authentification Waptservice

Options / Valeur par défaut	Description	Exemple
`ldap_auth_base_dn` (défaut `None`)	Utile avec *waptagent-ldap*, définit le *dn de base* pour la requête LDAP.	ldap_auth_base_dn = dc=mydomain,dc=lan
`ldap_auth_ssl_enabled` (défaut `False`)	Utile avec *waptagent-ldap*, définit si la requête LDAP doit être chiffré.	ldap_auth_ssl_enabled = True
`ldap_auth_server` (défaut `None`)	Utile avec *waptagent-ldap*, définit le serveur LDAP à contacter.	ldap_auth_server = srvads.mydomain.lan
`service_auth_type` (défaut `system`)	Définit le système d'authentification du service WAPT, les valeurs disponibles sont *system*, *waptserver-ldap*, *waptagent-ldap*.	service_auth_type = waptagent-ldap
`verify_cert_ldap` (défaut `False`)	Utile avec *waptagent-ldap*, définit si le certificat doit être vérifié.	verify_cert_ldap = True
`waptservice_admin_filter` (défaut `False`)	Appliquer un filtrage d'affichage pour les *paquets self-service* pour les administrateurs locaux.	waptservice_admin_filter = True
`waptservice_password` (défaut `None`)	mot de passe haché en sha256 lorsque *waptservice_user* est utilisé (la valeur *NOPASSWORD* désactive la nécessité d'un mot de passe).	waptservice_password = 5e884898da
`waptservice_user` (défaut `None`)	Force un utilisateur à s'authentifier sur le service WAPT.	waptservice_user = admin

Paramètres pour le wapttray

Table7: Description des options disponibles pour l'agent WAPT sur la section globale pour le wapttray

Options / Valeur par défaut	Description	Exemple
`notify_user` (défaut `False`)	Empêche `wapttray` d'envoyer des notifications (popup).	notify_user = True

Paramètres du Proxy

Table8: Description des options disponibles pour l'agent WAPT sur la section globale pour le proxy

Options / Valeur par défaut	Description	Exemple
`http_proxy` (défaut `None`)	Définit l'adresse du proxy HTTP.	`http_proxy = http://user:pwd@host_fqdn:port`
`use_http_proxy_for_repo` (défaut `False`)	Utiliser un proxy pour accéder aux dépôts.	`use_http_proxy_for_repo = True`
`use_http_proxy_for_server` (défaut `False`)	Utiliser un proxy pour accéder au serveur WAPT.	`use_http_proxy_for_server = True`

Paramètres pour la création de paquets WAPT

Table9: Description des options disponibles pour l'agent WAPT dans la section globale pour la création de paquets WAPT

Options / Valeur par défaut	Description	Exemple
`default_package_prefix` (défaut `tis`)	Définit le préfixe par défaut pour les paquets nouveaux ou importés. Le préfixe est sensible à la casse, nous recommandons d'utiliser les minuscules.	`default_package_prefix = doc`
`default_sources_root` (défaut `C:\waptdev` sur Windows ou `~/waptdev` sur Linux)	Définit le répertoire de stockage des paquets en cours de développement.	`default_sources_root = C:\\waptdev`
`personal_certificate_path` (défaut `None`)	Définit le chemin d'accès à la clé privée de l'administrateur.	`personal_certificate_pat = None`

36.2.2 [wapt-wua]

Reportez-vous à *configurer WAPTWUA sur l'agent WAPT*.

36.2.3 Paramètres pour l'utilisation de dépôts multiples

Pour ajouter plus de dépôts, de nouvelles sections `[nom_du_dépôt]` peuvent être ajoutées dans le `wapt-get.ini`.

Note: Les dépôts actifs sont listés dans l'attribut « repositories » de la section `[global]`.

Attention: Ce paramètre peut être configuré à la fois dans la configuration de l'agent WAPT et dans le fichier de configuration de la console WAPT `C:\Users\%username%\AppData\Local\waptconsole\waptconsole.ini`.

Pour des informations sur la configuration de la console WAPT, veuillez vous référer à *cette élément de la documentation*.

Section [wapt]

Indication: Si cette section n'existe pas, les paramètres sont lus à partir de la section `[global]`.

[wapt-templates]

Dépôts distants externes qui seront utilisés dans la console WAPT pour importer des nouveaux paquets ou leur mises à jour. Le dépôt Tranquil IT est défini par défaut.

Section [wapt-host]

Dépôt pour les paquets hôtes. Si cette section n'existe pas, les emplacements par défaut utilisés seront le dépôt principal.

Plus d'informations sur cette utilisation peuvent être trouvées dans *cette article sur le travail avec plusieurs dépôts publics ou privés*.

CHAPTER 37

Configuration de la console WAPT

Indication: la configuration de la console WAPT est stocké à 2 endroits:

- `C:\Users\%username%\AppData\Local\waptconsole\waptconsole.ini`.
- `C:\Users\%username%\AppData\Roaming\waptconsole\waptconsole.ini`.

Ces fichiers sont générés automatiquement lors du premier lancement de **waptconsole** et sont générés à partir du fichier `wapt-get.ini` configuré sur le poste de travail de l'*Administrateur* ;

37.1 Description des sections disponibles

Table1: Description des sections disponibles pour l'agent WAPT

Section	Description
`[global]`	Options globales de la Console WAPT
`[sections]`	Définit les options du dépôt externe. `[wapt-template]` *repositories*
`[waptwua]`	Options WUA

Toutes les sections sont détaillées ci-dessous.

Les autres sections présentes dans `C:\Users\%username%\AppData\Roaming\waptconsole\waptconsole.ini` ne sont pas modifiables manuellement, elles ne sont donc pas détaillées.

Attention: Pour les paramètres présents à la fois dans `wapt-get.ini` et `waptconsole.ini`, les valeurs sont définies dans `wapt-get.ini` et copiées dans `waptconsole.ini`. Ne modifiez pas manuellement ces paramètres.

37.2 Description des options disponibles par section

37.2.1 [global]

Plusieurs options sont disponibles dans la section `[global]` du fichier `waptconsole.ini`.

Table2: Description des options disponibles sur `AppData\Local`

Options / Valeur par défaut	Description	Exemple
`advanced_mode` (défaut `False`)	Lance la Console WAPT en mode débogage.	
`allow_remote_reboot` (défaut `False`)	Permet de redémarrer le ou les hôtes sélectionnés	
`allow_remote_shutdown` (défaut `False`)	Permet d'arrêter le ou les hôtes sélectionnés à dist	
`client_certificate` (défaut `None`)	Définit si le dépôt distant utilise l'authentification	
`client_private_key` (défaut `None`)	Définit si le dépôt distant utilise l'authentification	
`check_certificates_validity` (défaut `False`)	Force la vérification de la date et de la CRL du cer	
`default_maturity` (défaut `None`)	Définit la maturité de téléchargement par défaut po	
`default_package_prefix` (défaut `tis`)	Définit le préfixe par défaut pour les paquets nouv	
`default_sources_root` (défaut `C:\waptdev` sur Windows ou `~/waptdev` sur Linux)	Définit le répertoire de stockage des paquets en co	
`grid_hosts_plugins` (défaut `W10=`)	Liste *Les plugins externes* pour la console WAPT.	
`host_profiles` (défaut `None`)	Définit une liste de paquets WAPT que l'agent WA	
`hiberboot_enabled` (défaut `False`)	Désactive Hiberboot sur Windows 10 pour **wapte**	
`http_proxy` (défaut `None`)	Définit l'adresse du serveur proxy dans la console	
`last_usage_report` (défaut `None`)	Indique la date à laquelle la console WAPT a été u	
`lastwaptserveruser` (défaut `None`)	Fournit le dernier utilisateur connecté sur cette co	
`max_gpo_script_wait` (défaut `180`)	Définit le délai d'exécution des GPO à l'arrêt de l'	
`personal_certificate_path` (défaut `None`)	Définit le chemin d'accès au certificat associé à la	
`pre_shutdown_timeout` (défaut `180`)	Définit le délai d'attente pour les scripts à l'arrêt d	
`repo_url` (défaut l'adresse du dépôt WAPT)	Définit l'adresse du dépôt principal de WAPT.	
`send_usage_report` (défaut `True`)	Permet à la console WAPT d'envoyer des statistiqu	
`sign_digests` (défaut `sha256`)	Liste des algorithmes de signature autorisés pour l	
`use_ad_groups` (défaut `False`)	Permet l'utilisation des *paquets unit*.	
`use_fqdn_as_uuid` (défaut `False`)	Permet d'utiliser le FQDN plutôt que l'UUID du H	
`use_kerberos` (par défaut `False`)	Permet d'utiliser l'authentification kerberos pour l	
`use_hostpackages` (défaut `False`)	Permet d'utiliser *les paquets hôtes*.	
`use_http_proxy_for_repo` (défaut `False`)	Permet d'utiliser un proxy pour se connecter au dé	
`use_http_proxy_for_server` (défaut `False`)	Permet d'utiliser un proxy pour se connecter au se	
`use_repo_rules` (défaut `False`)	Permet d'utiliser *replication pour les dépôts*.	
`verify_cert` (défaut `False`)	Pour la *vérification des certificats SSL / TLS*.	
`wapt_server` (défaut `None`)	Définit le port du serveur PostgreSQL.	

Table3: Description des options disponibles sur `AppData\Roaming`

Options / Valeur par défaut	Description	Exemple
`advanced_mode` (défaut `False`)	Lance la Console WAPT en mode débogage.	advanced_mode = True
`enable_external_tools` (défaut `False`)	Affiche les actions qui appellent des applications externes (RDP, outils Windows etc…).	enable_external_tools = True
`enable_management_features` (défaut `False`)	Affiche le bouton pour créer des certificats auto-signés ou pour créer l'installateur de l'agent WAPT.	enable_management_features = True
`hide_unavailable_actions` (défaut `False`)	Masque les actions qui ne sont pas disponibles pour l'agent WAPT	hide_unavailable_actions = True
`HostsLimit` (défaut `2000`)	Limite des hôtes affichés dans la console WAPT.	HostsLimit = 300
`language` (langue par défaut du client WAPT)	Force la langue par défaut pour l'interface graphique (pas pour le filtrage des paquets)	language = fr
`lastappinifilename` (défaut `None`)	Définit le fichier *`.ini`* utilisé pour stocker la configuration de la console WAPT.	lastappinifilename = C:\Users\%username%\AppData\Roaming\waptconsol
`show_host_audit_data_tab` (défaut `False`)	Affiche l'onglet *Données d'audit* sur l'inventaire des machines.	show_host_audit_data_tab = True
`use_ad_groups` (défaut `False`)	Permet l'utilisation des *paquets unit*.	use_ad_groups = True
`use_fqdn_as_uuid` (défaut `False`)	Permet d'utiliser le FQDN plutôt que l'UUID du BIOS comme identifiant unique de la machine dans WAPT (par défaut False).	use_fqdn_as_uuid = True
`waptconsole.version` (défaut `None`)	Affiche la version de la Console WAPT.	waptconsole.version = 2.0.0.9424
`waptwua_enabled` (défaut `False`)	Affiche l'onglet *Mises à jour Windows* sur la Console WAPT.	waptwua_enabled = True

37.2.2 [sections]

Vous pouvez ajouter plusieurs dépôts externes en ajoutant `[sections]` dans `C:\Users\%username%\AppData\Local\waptconsole\waptconsole.ini`.

> **Attention:** Ce paramètre peut être configuré à la fois dans la configuration de l'agent WAPT et dans la configuration de la console WAPT `C:\Users\%username%\AppData\Local\waptconsole\waptconsole.ini`.
>
> Pour des informations sur la configuration de l'agent WAPT, veuillez vous référer à *ce point*.

Voir les paramètres et configurations disponibles *ici*.

CHAPTER 38

Configuration du Serveur WAPT

Le fichier de configuration du serveur WAPT sur les systèmes GNU/ Linux et macOS se trouve dans `/opt/wapt/conf/waptserver.ini` ou dans `/opt/wapt/waptserver/waptserver.ini`.

Le fichier de configuration du serveur WAPT sur les systèmes Windows se trouve dans `C:\wapt\conf\waptserver.ini`.

Attention: La modification de ces fichiers est réservée aux utilisateurs avancés !

38.1 Section [option] de waptserver.ini

Plusieurs options peuvent être définies dans cette section.

```
[options]
```

Table1: Paramètres disponibles pour la section [options] du fichier `waptserver.ini`

Options / Valeur par défaut	Description	Exemple
`agents_folder` (défaut `watpagent in wapt repository`)	Définit où les agents WAPT sont stockés sur le serveur WAPT.	
`allow_unauthenticated_connect` (défaut `None`)	Définit si les connexions websocket doivent être authentifiées. Si :code:`use`	
`allow_unauthenticated_registration` (défaut `False`)	Permet l'enregistrement initial de l'agent WAPT en utilisant un login et un	
`allow_unsigned_status_data` (défaut `False`)	Débogage uniquement - Permet à l'agent d'envoyer des données d'état non	
`application_root` (défaut `None`)	Définit un chemin racine personnalisé pour le serveur WAPT.	
`authentication_logs` (défaut `True`)	Active les journaux d'authentification.	
`auto_create_waptagent_from_config` (défaut `False`)	Utilise le fichier de configuration automatique lorsque waptsetup est installé	
`client_certificate_lifetime` (défaut `3650`)	Définit la durée de vie du certificat de la machine (en jours).	
`cleanup_kbs` (défaut `True`)	Définit si les KB Windows non utilisés :ref:` doivent être automatiquement	

suite sur la page suivante

Table 1 – suite de la page précédente

Options / Valeur par défaut	Description	Exemple
clients_read_timeout (défaut 5)	Définit le délai d'attente du client websocket (en secondes).	
clients_signing_certificate (défaut None)	Définit le certificat de signature de la machine.	
clients_signing_crl_days (défaut 30)	Définit la périodicité de la signature du certificat de la CRL (Certificate Rev	
clients_signing_crl (défaut None)	Définit le chemin de la CRL du certificat de signature machine.	
clients_signing_crl_url (défaut None)	Définit l'URL de la CRL du certificat de signature machine.	
clients_signing_key (défaut None)	Définit le chemin de la clé de signature du certificat de la machine.	
client_tasks_timeout (défaut 5)	Définit le délai maximum autorisé avant que l'agent WAPT bascule en time	
copy_winpe_x64_in_tftp_folder (défaut False)	Si x64, permet de copier tous les WinPE à partir de wads_folder lorsque	
db_connect_timeout (défaut 3)	Définit le délai maximum autorisé avant que les requêtes PostgreSQL n'exp	
db_host (défaut None)	Définit l'url du serveur PostgreSQL (par défaut WAPT utilise une socket Ur	
db_max_connections (défaut 90)	Définit le nombre maximum de connexions simultanées à la base de donnée	
db_name (défaut wapt)	Définit la base de données PostgreSQL à laquelle le serveur WAPT se conn	
db_password (défaut None)	Définit le mot de passe pour authentifier l'utilisateur sur la base de données	
db_port (défaut 5432)	Définit le port du serveur PostgreSQL.	
db_stale_timeout (défaut 300)	Définit le délai d'expiration de la requête à la base de données (en secondes	
db_user (défaut wapt)	Définit l'utilisateur PostgreSQL qui se connecte à la base de données.	
default_ldap_users_acls (défaut view)	Définit l'acl par défaut pour un nouvel utilisateur qui ouvre la Console WAI	
download_wsusscn2 (défaut False)	Télécharge automatiquement le fichier :*wsusscn2.cab*.	
enable_store (défaut False)	Active le store WAPT web (**Déprécié**).	
encrypt_host_packages (défaut False)	Chiffre le paquet hôte avec le certificat du client.	
htpasswd_path (défaut None)	Ajoute l'authentification de base au serveur WAPT.	
http_proxy (défaut None)	Définit le serveur proxy pour permettre au serveur WAPT de récupérer sa C	
known_certificates_folder (défaut répertoire /ssl/)	Ajoute une CA supplémentaire connue pour vérifier les certificats.	
ldap_account_service_login (défaut None)	Définit l'utilisateur UPN Active Directory pour le SSO et/ou le mode wapts	
ldap_account_service_password (défaut None)	Définit le mot de passe de l'utilisateur pour le SSO et/ou le mode waptserve	
ldap_auth_base_dn (défaut None)	Définit le DN de base pour l'authentification LDAP.	
ldap_auth_server (défaut None)	Définit le serveur d'authentification LDAP.	
ldap_auth_ssl_enabled (défaut True)	Définit l'authentification SSL pour les connexions LDAP.	
ldap_nesting_group_support (défaut True)	Permet la recherche de groupes imbriqués dans Active Directory.	
ldap_primary_group_ad_support (défaut True)	Active la recherche sur les utilisateurs du groupe primaire d'Active Director	
list_subnet_skip_login_wads (défaut [])	Liste les sous-réseaux sans exigence d'authentification.	
login_on_wads (default False)	Active l'authentification pour utiliser WADS (le format est user:passwor	
loglevel (défaut warning)	Définit le niveau du journal d'historisation. Les valeurs possibles sont : deb	
max_clients (défaut 4096)	Définit le nombre maximum de connexions simultanées de clients WAPT.	
min_password_length (défaut 10)	Définit la longueur minimale du mot de passe du *SuperAdmin*.	
nginx_http (défaut 80)	Définit le port **HTTP** du serveur web Nginx (Windows uniquement).	
nginx_https (défaut 443)	Définit le port **HTTPS** du serveur web Nginx (Windows uniquement).	
optimized_authentication_logs (défaut True)	Si une de ces options est renseignée, le journal d'historisation ne l'enregistr	
remote_repo_update_delay (default True)	Définit la périodicité à laquelle le serveur WAPT vérifie l'état de synchroni	
remote_repo_websockets (défaut True)	Permet la communication websocket avec les agents WAPT configurés en ta	
secret_key (défaut None)	Définit la chaîne aléatoire pour initialiser le serveur d'application Python Fl	
server_uuid (défaut None)	Définit l'*UUID* du Serveru WAPT (cet identifiant anonyme est utilisé pour	
session_lifetime (défaut 126060)	Définit le temps maximum autorisé pour l'ouverture de la session (en secon	
signature_clockskew (défaut 300)	Définit la différence de temps maximale autorisée pour les websockets (en s	
token_lifetime (défaut 43200)	Définit la durée de vie du jeton d'authentification (en secondes).	
trusted_signers_certificates_folder (défaut None)	Définit le chemin d'accès au répertoire des certificats des signataires de con	
trusted_users_certificates_folder (défaut None)	Définit le chemin d'accès au répertoire des certificats CA des utilisateurs de	

suite sur la page suivante

Table 1 – suite de la page précédente

Options / Valeur par défaut	Description	Exemple
use_kerberos (par défaut False)	Permet à un Agent WAPT de s'enregistrer en utilisant son compte kerberos.	
use_ssl_client_auth (défaut False)	Active *l'authentification par certificat client*.	
verify_cert_ldap (défaut True)	Vérifie si le certificat SSL / TLS d'Active Directory est valide pour les conn	
wads_enable (défaut False)	Active la fonctionnalité WADS et active **wapttftpserver**.	
wads_folder (défaut répertoire wads dans le dépôt WAPT)		Définit le répertoire
wapt_admin_group_dn (défaut None)		Définit le DN LDAP
wapt_admin_group (défaut None)		Définit le(s) groupe(
wapt_folder (défaut /var/www/wapt ou /var/www/html/wapt ou root_dir/waptserver/repository/wapt)		Définit le chemin ve
wapt_huey_db (défaut None)		Définit le chemin ve
wapt_password (défaut None)		Définit le mot de pas
waptserver_port (défaut 8080)		Définit le port du se
wapt_user (défaut admin)		Définit le nom d'util
waptwua_folder (défaut wapt_folder + "wua")		Définit l'emplaceme
wol_port (défaut 7,9)		Définit la liste des p
wapt_bind_interface (défaut 127.0.0.1)		Définit comment éc
ipxe_script_jinja_path (défaut /opt/wapt/waptserver/templates/ipxe-default.j2)		Définit l'emplaceme

38.2 Configuration de Nginx

La configuration par défaut de Nginx est la suivante :

```
server {
  listen                    80;
  listen                    443 ssl;
  server_name               _;
  ssl_certificate           "/opt/wapt/waptserver/ssl/cert.pem";
  ssl_certificate_key       "/opt/wapt/waptserver/ssl/key.pem";
  ssl_protocols             TLSv1.2;
  ssl_dhparam               /etc/ssl/certs/dhparam.pem;
  ssl_prefer_server_ciphers on;
  ssl_ciphers               'EECDH+AESGCM:EDH+AESGCM:AES256+EECDH:AES256+EDH';
  ssl_stapling              on;
  ssl_stapling_verify       on;
  ssl_session_cache         none;
  ssl_session_tickets       off;
  index index.html;

  location ~ ^/wapt.* {
    proxy_set_header Cache-Control "store, no-cache, must-revalidate, post-check=0, pre-check=0";
    proxy_set_header Pragma "no-cache";
    proxy_set_header Expires "Sun, 19 Nov 1978 05:00:00 GMT";
    root "/var/www";
    }

  location / {
    proxy_set_header X-Real-IP $remote_addr;
    proxy_set_header Host $host;
```

(suite sur la page suivante)

(suite de la page précédente)

```
    proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
    proxy_set_header X-Forwarded-Proto $scheme;

  location ~ ^/(api/v3/upload_packages|api/v3/upload_hosts/|upload_waptsetup) {
    proxy_pass http://127.0.0.1:8080;
    client_max_body_size 4096m;
    client_body_timeout 1800;
    }

  location /wapt-host/Packages {
    return 403;
    }

  location /wapt-host/add_host_kerberos {
    return 403;
    }

  location / {
    proxy_pass http://127.0.0.1:8080;
    }

  location /socket.io {
    proxy_http_version 1.1;
    proxy_buffering off;
    proxy_set_header Upgrade $http_upgrade;
    proxy_set_header Connection "Upgrade";
    proxy_pass http://127.0.0.1:8080/socket.io;
    }
  }
}
```

38.3 Configuration du serveur WAPT pour les grands déploiements

Les paramètres par défaut du système d'exploitation, de Nginx et de Postgresql sont adaptés pour environ 400 agents WAPT. Si vous avez plus de 400 clients, il est nécessaire de modifier quelques paramètres au niveau du système ainsi que la base de données PostgreSQL, le serveur web Nginx et le serveur python WAPT Server.

Dans le futur, le script **postconf.sh** pourra prendre en charge cette configuration en fonction du nombre d'ordinateurs clients attendus.

Avec les paramètres suivants, un serveur WAPT devrait pouvoir fonctionner avec environ 5000 clients actifs simultanés. Vous pouvez avoir plus de clients dans la base de données s'ils ne fonctionnent pas tous en même temps. Si vous avez plus de 5000 clients, il est recommandé d'avoir plus d'un serveur WAPT.

La limite du nombre de clients finaux est due à un goulot d'étranglement dans le code python et le backend PostgreSQL. Les performances de WAPT s'améliorent avec le temps et, à l'avenir, le serveur WAPT pourrait supporter une large base sur un seul serveur. Cependant, la partie Nginx s'adapte très bien et peut tirer pleinement parti d'une connexion 10Gbps pour les déploiements de paquets à forte charge.

Note: Les paramètres à modifier ci-dessous sont liés entre eux et doivent être modifiés globalement et non individuellement.

38.3.1 Configuration de Nginx

Table2: `nginx.conf` emplacement du fichier de configuration

Type d'OS	Localisation des fichiers
Debian / Ubuntu	`/etc/nginx/nginx.conf`
Redhat et dérivés	`/etc/nginx/nginx.conf`
Windows	`C:\wapt\waptserver\nginx\conf\nginx.conf`

Dans le fichier `nginx.conf`, modifiez le paramètre `worker_connections`. La valeur doit être d'environ 2,5 fois le nombre de clients WAPT (n connexions pour les websockets et n connexions pour les téléchargements de packages et le téléchargement d'inventaire + une certaine marge).

```
events {
  worker_connections 4096;
}
```

Mettez ensuite à niveau le nombre de *filedescriptors* dans le fichier `/etc/nginx/nginx.conf` (pour Windows `C:\wapt\waptserver\nginx\conf\nginx.conf`) :

```
worker_rlimit_nofile 32768;
```

En fonction du partitionnement de votre serveur WAPT, vous devrez peut-être faire attention au répertoire de téléchargement de fichiers temporaires de Nginx. Nginx agit comme un proxy inverse pour le moteur Python de WAPTServer et fait une mise en cache des paquets téléchargés lors du téléchargement d'un nouveau paquet depuis la console.

Les paquets sont stockés dans le répertoire `/var/lib/nginx/proxy`. Vous devez vous assurer que la partition qui héberge ce répertoire est suffisamment grande. Vous pouvez modifier l'emplacement de ce répertoire en utilisant le paramètre de configuration suivant de Nginx.

```
$client_body_temp_path
```

38.3.2 Configuration du système Linux

Augmenter le nombre de *filedescriptors*. Le fichier d'unité du système demande une augmentation du nombre autorisé de *filedescriptors* (LimitNOFILE=32768). Nous devrions avoir la même chose pour Nginx. Il y a quelques limites à modifier.

Tout d'abord, nous modifions au niveau du système le nombre de *filedescriptors* autorisés pour Nginx et WAPT.

- Créer `/etc/security/limits.d/wapt.conf`.

```
cat > /etc/security/limits.d/wapt.conf <<EOF
wapt         hard    nofile      32768
wapt         soft    nofile      32768
www-data     hard    nofile      32768
www-data     soft    nofile      32768
EOF
```

Nginx sert de proxy inverse et établit un grand nombre de connexions. Chaque client WAPT maintient une connexion *websocket* en permanence afin de répondre aux actions du serveur WAPT.

Le noyau Linux a une protection contre le fait d'avoir trop de connexions TCP ouvertes en même temps et on peut obtenir le message *SYN flooding on port* dans le journal de Nginx. Afin d'éviter ces messages, il est nécessaire de modifier les deux paramètres suivants. Il doit être environ 1,5 fois le nombre de clients WAPT.

```
cat > /etc/sysctl.d/wapt.conf <<EOF
net.ipv4.tcp_max_syn_backlog=4096
net.core.somaxconn=4096
EOF

sysctl --system
```

38.3.3 Configuration de la base de données PostgreSQL

Table3: `postgresql.conf` emplacement du fichier de configuration

Type d'OS	Localisation des fichiers
Debian / Ubuntu	`/etc/postgresql/{version}/main/postgresql.conf`
Redhat et dérivés	`/var/lib/pgsql/{version}/data/postgresql.conf`
Windows	`C:\wapt\waptserver\pgsql{version}_data\postgresql.conf`

Un plus grand nombre de clients nécessite un plus grand nombre de connexions à la base de données PostgreSQL. Dans le fichier `postgresql.conf` file (file:*/etc/postgresql/11/main/postgresql.conf* sur debian 10 par exemple ou pour Windows `C:\wapt\waptserver\pgsql9.6_data\postgresql.conf`), vous devez augmenter le paramètre suivant pour atteindre approximativement 1/4 du nombre d'agents WAPT actifs.

```
max_connections = 1000
```

Dans le fichier `/opt/wapt/conf/waptserver.ini` (pour Windows `C:\wapt\conf\waptserver.ini`, *db_max_connections* ` doit être égal au *max_connections* ` de PostgreSQL moins 10 (PostgreSQL a besoin de garder quelques connexions pour son ménage). Le paramètre `max_clients` devrait être fixé à environ 1,2 fois le nombre d'agents WAPT :

```
[options]
...
max_clients = 4096
db_max_connections = 990
```

CHAPTER 39

Configuration des dépôts WAPT

39.1 Localisation du dépôt sur le serveur

Système d'exploitation	Valeur
Debian / Ubuntu	/var/www/wapt/
Redhat et dérivés	/var/www/html/wapt/
Windows	C:\wapt\waptserver\repository

39.2 Répliquer un dépôt

39.2.1 Aperçu fonctionnel

Indication: **La méthode expliquée ci-dessous ne concerne que la version Enterprise**.

La méthode Syncthing, est dépréciée et **non supportée**, mais peut être utilisée pour les versions Discovery de WAPT.

Rôle de réplication de l'agent WAPT

La réplication du dépôt peut être activée en utilisant un agent WAPT installé sur une machine existante, une appliance dédiée ou une machine virtuelle.

Le rôle de réplication est déployé par le biais d'un paquet WAPT qui active **le serveur web Nginx** et configure la planification, les types de paquets, la synchronisation des paquets, et bien plus encore.

Cette fonctionnalité permet aux agents WAPT de trouver dynamiquement leur dépôt WAPT disponible le plus proche à partir d'une liste de règles stockées sur le serveur WAPT.

Comportement de réplication

La réplication du dépôt dans WAPT est gérée nativement par les agents WAPT.

Il est basé sur un fichier `sync.json` qui indexe tous les fichiers présents dans ces dossiers :

- `wapt` ;
- `waptwua`;
- `wapt-host` ;
- `wads`.

L'activation de la réplication a les effets suivants :

- Une fois que `enable_remote_repo` est activé sur un agent WAPT, il synchronisera les paquets localement dans le dossier `local_repo_path`.
- Il ajoute l'agent WAPT dans l'onglet *Dépôts secondaires* comme un dépôt distant, permettant de nouvelles actions telles que *Sync tous* ou *Créer l'index*.
- Par défaut, seul le dossier *wapt* est synchronisé, vous pouvez sélectionner le dossier à synchroniser en ajoutant des éléments dans les paramètres `remote_repo_dirs`.
- La période de synchronisation peut être configurée avec les paramètres `local_repo_time_for_sync_start` et `local_repo_time_for_sync_stop`.
- La bande passante allouée à la synchronisation peut être configurée avec `local_repo_limit_bandwidth`.

Tous les paramètres de la synchronisation du dépôt WAPT doivent être définis dans la section `[repo-sync]` du fichier de configuration `wapt-get.ini` de l'agent WAPT.

39.2.2 Configuration de l'agent WAPT

Pour activer la réplication sur un *Agent WAPT existant* (Linux / Windows) vous devez définir dans la section `[repo-sync]` dans le fichier de configuration `wapt-get.ini`.

Indication: Si vous utilisez le DNS, n'oubliez pas de créer une entrée DNS pour votre agent WAPT.

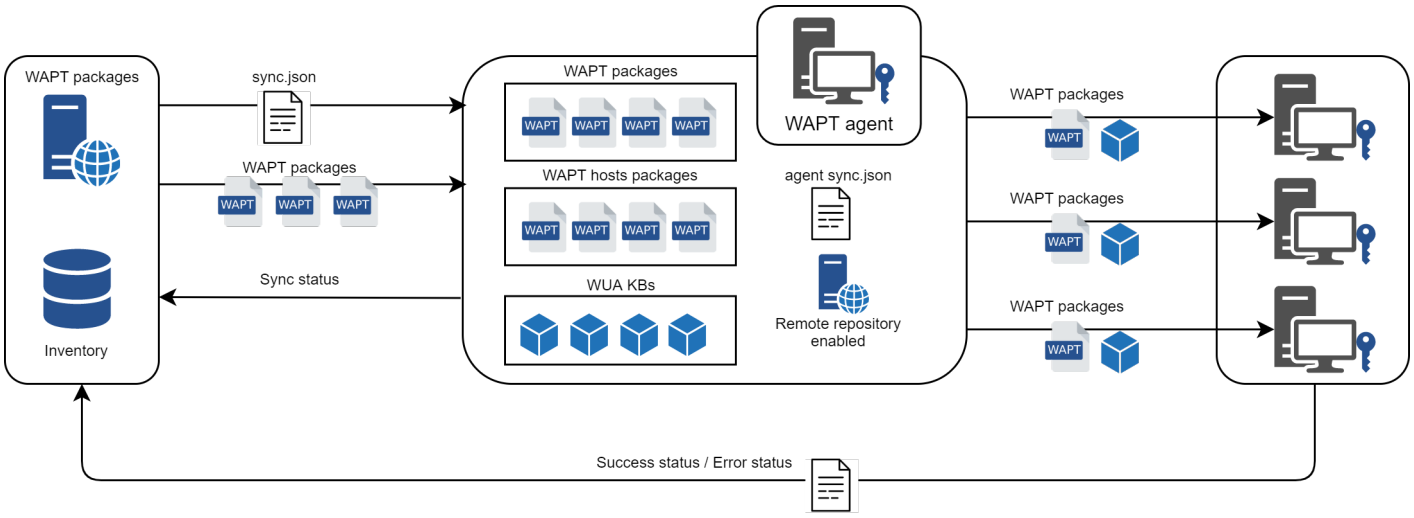


Figure1: Diagramme de flux du comportement de réplication de l'agent WAPT

Table1: Configuration de la réplication de l'agent WAPT

Options / Valeur par défaut	Définition	Exemple
`enable_remote_repo` (défaut `False`)	Permet au dépôt secondaire de se synchroniser avec le dépôt principal.	`enable_remote_repo` (défaut `True`)
`local_repo_path` (défaut``WAPT root dir/repository``)	Définit le chemin vers le répertoire racine du dépôt local pour les paquets WAPT.	`local_repo_path = /var/www/`
`local_repo_time_for_sync_start` (défaut `None`)	Définit l'heure de début de la synchronisation (HH:MM / format 24h).	`local_repo_time_for_sync_start = 22:30`
`local_repo_time_for_sync_end` (défaut `None`)	Définit l'heure d'arrêt de la synchronisation (HH:MM / format 24h).	`local_repo_time_for_sync_end = 05:30`
`local_repo_sync_task_period` (défaut `None`)	Définit la périodicité de la synchronisation (en minutes).	`local_repo_sync_task_period = 25`
`local_repo_limit_bandwidth` (défaut `None`)	Définit la largeur de bande autorisée pour la synchronisation (en Mbits/s).	`local_repo_limit_bandwidth = 2`
`remote_repo_dirs` (défaut `wapt, waptwua`)	Définit les dossiers à synchroniser.	`remote_repo_dirs = wapt, waptwua,wapt-host`
`use_repo_rules` (défaut `False`)	Active l'utilisation des *règles du dépôt*.	`use_repo_rules = True`
`sync_only_forced` (défaut `False`)	Synchronise le dépôt uniquement s'il est forcé.	`sync_only_forced = True`

Avertissement: Si vous modifiez manuellement le fichier `wapt-get.ini` sur le dépôt secondaire, vous devez redémarrer **waptservice**.

Note: Un Paquet WAPT prêt à l'emploi est disponible sur le **store public Tranquil IT** pour permettre la réplication du dépôt sur des agents WAPT basés sur Windows ou Linux.

Ainsi, le bureau de l'accueil d'un bureau distant de n'importe quelle organisation peut devenir un référentiel WAPT pour distribuer des packages WAPT à la flotte d'ordinateurs du bureau distant.

Ce paquet spécial :

- Installe et active le serveur web **Nginx** sur le dépôt seconfaire.
- Configure l'environnement de l'hôte virtuel **Nginx**.
- Active la configuration du dépôt secondaire dans `wapt-get.ini`.

Il est possible de configurer automatiquement les dépôts avec vos propres valeurs en modifiant ce paquet.

Voici un exemple de `wapt-get.ini`.

```
[global]
...
use_repo_rules = True

[repo-sync]
enable_remote_repo = True
local_repo_path = D:\WAPT\
local_repo_time_for_sync_start = 20:30
local_repo_time_for_sync_end = 05:30
local_repo_sync_task_period = 25
local_repo_limit_bandwidth = 4
remote_repo_dirs = wapt, waptwua, wapt-host
```

39.2.3 Configuration du serveur WAPT

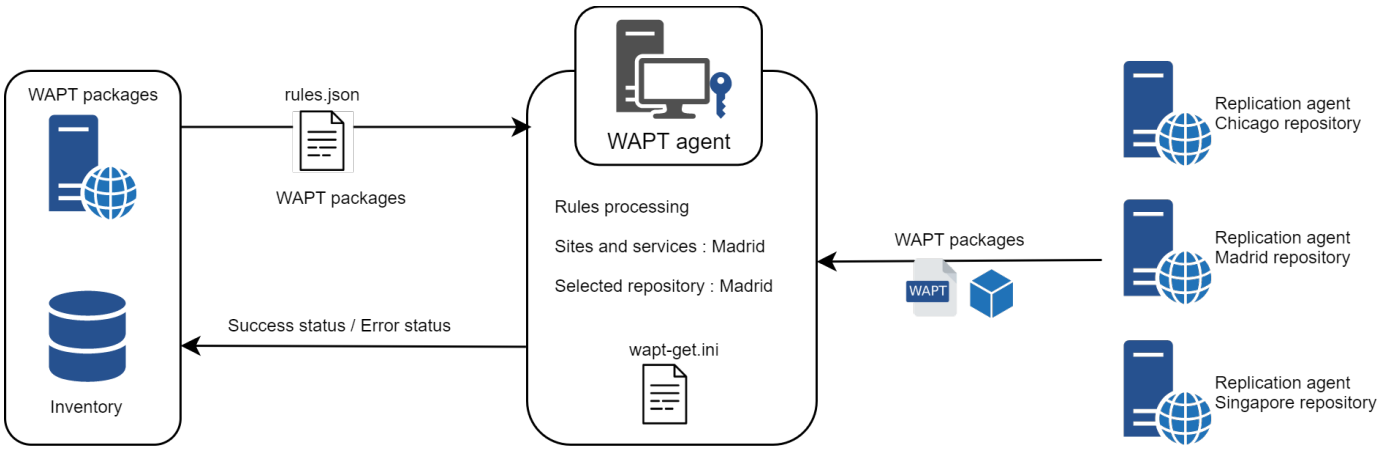
Par défaut, le serveur sait quels agents WAPT sont configurés en tant que dépôt secondaire et il les répertorie dans la console WAPT.

39.2.4 Règles du dépôt

Lorsqu'un agent WAPT a été configuré comme dépôt, il récupère automatiquement son fichier `rules.json` depuis le serveur WAPT.

Le fichier `rules.json` est un fichier *`.json`* signé qui contient une liste de règles triées à appliquer aux agents WAPT distants, afin qu'ils puissent se connecter aux dépôts les plus appropriés.

Si aucune règle ne correspond, l'agent WAPT se rabattra sur le paramètre `repo_url` du serveur WAPT défini dans le fichier de configuration `wapt-get.ini`.



Agent WAPT

Avertissement: Si vous avez configuré des redirections GeoIP sur Nginx, vous devez les désactiver car elles peuvent entrer en conflit avec les règles du dépôt.

Pour activer les règles du dépôt de l'agent WAPT, vous devez activer ce paramètre dans la section `[global]` du fichier de configuration `wapt-get.ini` de l'agent WAPT.

Options / Valeur par défaut	Description	Exemple
`use_repo_rules` (défaut `False`)	Pour l'utilisation de *réplication du dépôt*.	`use_repo_rules = True`

Voici un exemple de `wapt-get.ini`.

```
[global]
...
use_repo_rules = True
```

Note: Il est possible d'activer cette option lors de la *génération d'un agent WAPT*.

Serveur WAPT

Sur le serveur WAPT, la fonctionnalité des dépôts secondaires est automatiquement activée.

Pour le contrôler, éditez `waptserver.ini` et lisez la valeur `remote_repo_support`.

Options / Valeur par défaut	Exemple de valeur	Définition
`remote_repo_support`	`True`	Permet au dépôt secondaire de se synchroniser avec le dépôt principal.

Console WAPT

Les règles du dépôt sont gérées à partir de la console WAPT et sont basés sur plusieurs paramètres:

Table2: Paramètres disponibles pour les règles du dépôt

Options	Exemple de valeur	Description
Agent IP	`192.168.85.0/24`	Règle basée sur le sous-réseau IP de l'agent.
Domaine	`ad.mydomain.lan`	Règle basée sur le nom de domaine Active Directory.
Nom d'hôte	`desktop-04feb1`	Règle basée sur le nom d'hôte de l'agent WAPT.
Public IP	`256.89.299.22/32`	Règle basée sur l'adresse IP publique (hôtes NATés).
Site	`Paris-HQ`	Règle basée sur les sites et services Active Directory.

Création d'une nouvelle règle de dépôt

Pour ajouter une nouvelle règle de dépôt, allez dans l'onglet *Dépôts* de la console WAPT et cliquez sur le bouton *Ajouter une règle*.

Table3: Création d'une nouvelle règle de dépôt

Options	Exemple de valeur	Description
Nom	repo25	Définit le nom de la règle.
Condition	IP de l'agent	Définit la condition à remplir pour que la règle s'applique (voir ci-dessus).
Valeur	192.168.25.0/24	Définit la valeur lorsque la condition s'applique. Si la case « NON » est cochée, la valeur s'applique à l'inverse de la condition.
URL du dépôt	https://repo25.domain.lan	Définit la liste des dépôts secondaires disponibles. La liste inclut `http://download.windowsupdate.com/microsoftupdate/v6/wsusscan/` pour permettre le téléchargement direct des mises à jour de Windows par les dépôts secondaires afin de préserver la bande passante.
Type de paquet	WAPT	Définit quels *types de paquets* sont répliqués.
Autre	Pas de fallback	L'option *Pas de fallback* empêchera de se replier sur le serveur WAPT principal et évitera une congestion indésirable du réseau si le dépôt secondaire devient temporairement indisponible.

- L'option *Proxy* devra être définie si le dépôt secondaire doit se connecter via un proxy.

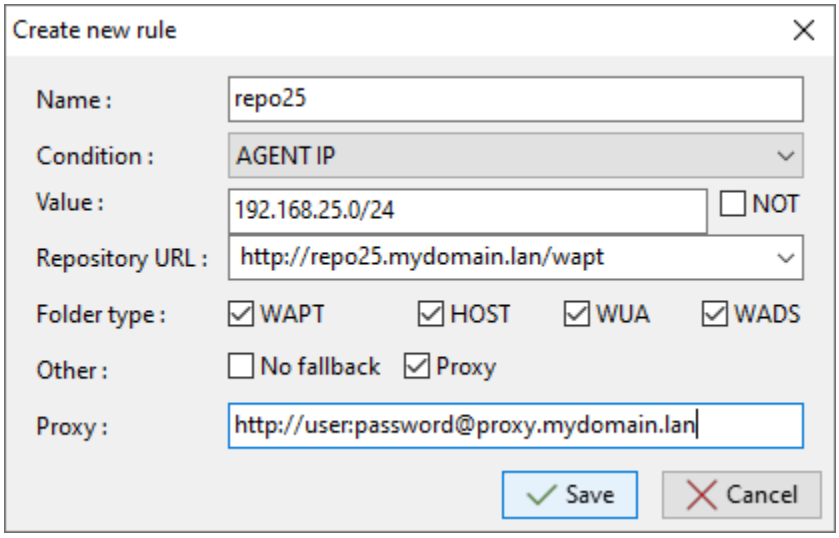


Figure2: Fenêtre pour définir les règles du dépôt dans la console WAPT

Vous pouvez ensuite choisir parmi les différents paramètres ci-dessus et affecter des valeurs à un dépôt secondaire WAPT spécifique.

Avertissement: Les règles sont appliquées de haut en bas.

Les règles sont appliquées de haut en bas. La première règle qui correspond aux conditions prévaut sur toutes les autres règles placées en dessous.

Danger: N'oubliez pas de sauvegarder vos règles de réplication.

39.3 Dépôts multiples

Comme pour les dépôts Debian, il est possible pour l'agent WAPT d'utiliser plusieurs dépôts pour la mise à jour des paquets. Les agents WAPT vérifieront tous les dépôts.

Danger: Si vous utilisez cette fonctionnalité, **SACHEZ CE QUE VOUS FAITES**.

Lorsque vous utilisez des dépôts avec différents signataires, les certificats publics du signataire supplémentaire doivent être ajoutés à `C:\Program Files (x86)\wapt\ssl` sous Windows ou `/opt/wapt/ssl` sous Linux et MacOS, par conséquent, vous **DEVEZ** faire confiance à leur travail et à leur signature.

Vous devez ensuite déployer l'agent WAPT avec les deux clés.

Veuillez vous reporter à la documentation sur *la création de l'agent WAPT* pour ajouter d'autres certificats de confiance.

39.3.1 Configuration de l'agent WAPT

Ces paramètres sont modifiables dans le fichier `wapt-get.ini`.

Description des paramètres disponibles

Table4: Description des options disponibles pour l'utilisation de dépôts multiples

Option	Exemple de valeur	Description
[global]	repositories = wapt-templates,private	Le paramètre *repositories* permet de définir plusieurs options pour les dépôts de paquets, par exemple *wapt-templates* et *private*, où leurs paramètres sont définis dans une `[section]` supplémentaire du fichier.
`[section]`	[wapt-templates] repo_url=https://store.wapt.fr/wapt verify_cert = True [private] repo_url=https://srvwapt.mydomain.lan/wapt verify_cert = False	Tous les paramètres de la synchronisation du dépôt WAPT doivent être définis dans la section `[repo-sync]` du fichier de configuration `wapt-get.ini` de l'agent WAPT.

Table5: Options pour les propriétés du dépôt

Options / Valeur par défaut	Description	Exemple
`http_proxy` (défaut `None`)	Définit l'adresse du proxy HTTP.	http_proxy = http://user:pwd@host_fqdn:port
`repo_url` (défaut `None`)	Définit l'adresse du dépôt WAPT principal.	repo_url = https://srvwapt.mydomain.lan/wapt
`timeout` (défaut `None`)	Définit le délai d'attente lors de la connexion à des dépôts distants (en millisecondes)	timeout = 5000
`use_http_proxy_for_repo` (défaut `False`)	Définit si un proxy doit être défini pour accéder aux dépôts.	use_http_proxy_for_repo = True
`verify_cert` (défaut `None`)	Définit si *Les certificats HTTPS du dépôt doivent être vérifiés*, et si c'est le cas définit le chemin vers le paquet de certificats.	verify_cert = True

Note: L'agent WAPT recherchera les mises à jour dans tous les référentiels définis dans son fichier de configuration `wapt-get.ini` lorsqu'il effectue une **`recherche wapt-get`**.

Plus d'informations sur *l'utilisation de WAPT avec l'interface de ligne de commande*.

39.3.2 Description des options disponibles pour l'utilisation de dépôts multiples

Après avoir configuré l'agent WAPT pour utiliser plusieurs dépôts, nous pouvons faire apparaître les dépôts dans la console WAPT.

Pour cela, modifier le fichier `%appdata%\local\waptconsole\waptconsole.ini`.

Exemple :

```
[wapt-template]
repo_url = https://wapt.tranquil.it/wapt
http_proxy =
verify_cert = True
public_certs_dir =
client_certificate =
client_private_key =
timeout = 5

[private]
repo_url = https://srvwapt.mydomain.lan/wapt
http_proxy =
verify_cert = False
public_certs_dir =
client_certificate =
client_private_key =
timeout = 5
```

Table6: Options pour les référentiels externes dans la Console WAPT

Options / Valeur par défaut	Description	Exemple
`client_certificate` (défaut `None`)	Définit le dossier qui contient les certificats utilisés pour authentifier les paquets externes téléchargés.	client_certificate = `C:\Program Files (x86)\wapt\ssl\server\srvwapt.mydomain.lan.crt (on Windows)`
`client_private_key` = None	Définit le dossier qui contient la clé privée.	client_private_key = C:\Program Files (x86)\wapt\ssl\server\srvwapt.mydomain.lan.pem (on Windows)
`http_proxy` (défaut `None`)	Définit l'adresse du proxy HTTP.	http_proxy = http://user:pwd@srvproxy.mydomain.lan:port
`public_certs_dir` =	Définit le dossier qui contient les certificats utilisés pour authentifier les paquets externes téléchargés.	public_certs_dir = C:\private
`repo_url` (défaut `None`)	Définit l'adresse du dépôt WAPT principal.	repo_url = https://srvwapt.mydomain.lan/wapt
`timeout` (défaut `None`)	Définit le délai d'attente lors de la connexion à des dépôts distants (en millisecondes)	timeout = 5000
`verify_cert` (défaut `None`)	Définit si *Les certificats HTTPS du dépôt doivent être vérifiés*, et si c'est le cas définit le chemin vers le paquet de certificats.	verify_cert = True

CHAPTER 40

Renforcer la sécurité de votre installation WAPT

Par défaut, tous les paquets WAPT sont signés avec votre clé privée, ce qui offre déjà un haut niveau de sécurité. Cependant, vous pouvez améliorer davantage la sécurité de WAPT.

Pour sécuriser complètement votre installation WAPT, vous devez procéder comme suit :

- Activez l'enregistrement authentifié pour filtrer les personnes autorisées à enregistrer le périphérique auprès du serveur WAPT.
- Activez la vérification du certificat https sur les agents et la console pour vous assurer que les agents WAPT et la console WAPT se connectent au bon serveur WAPT.
- Configurez l'authentification Active Directory pour permettre l'accès à la console WAPT uniquement aux administrateurs WAPT autorisés.
- Activez l'authentification par certificat côté client pour n'autoriser que les appareils authentifiés à accéder au serveur WAPT (Remarque : c'est particulièrement important si vous voulez exposer votre serveur WAPT à l'extérieur dans une DMZ (De-Militarized Zone)).
- Si vous utilisez la version **Enterprise** de WAPT et que vous exploitez une grande flotte avec plusieurs administrateurs, vous serez peut-être intéressé de savoir comment configurer et appliquer correctement les ACLs.

40.1 Configuration du pare-feu sur le serveur WAPT

La configuration du pare-feu du serveur WAPT est essentielle et devrait être la première étape pour obtenir une meilleure sécurité dans WAPT.

Comme WAPT vise à être sécurisé dès la conception, seul un ensemble minimal de *ports ouverts* est nécessaire sur le serveur WAPT par rapport aux autres solutions.

Vous trouverez dans la documentation suivante des conseils autour des configurations de pare-feu pour renforcer la sécurité du serveur WAPT.

40.1.1 Configuration du pare-feu pour le serveur WAPT sur Debian / Ubuntu

Par défaut sur Debian Linux, aucune règle de pare-feu ne s'applique.

- Désactivez **ufw** et installez **firewalld** à la place.

```
ufw disable
apt update
apt -y install firewalld
```

- Il suffit d'appliquer cette configuration **firewalld**.

```
systemctl start firewalld
systemctl enable firewalld
firewall-cmd --zone=public --add-port=80/tcp --permanent
firewall-cmd --zone=public --add-port=443/tcp --permanent
systemctl restart firewalld
```

40.1.2 Configuration du pare-feu pour le serveur WAPT sur RedHat / CentOS

- Il suffit d'appliquer cette configuration **firewalld**.

```
systemctl start firewalld
systemctl enable firewalld
firewall-cmd --zone=public --add-port=80/tcp --permanent
firewall-cmd --zone=public --add-port=443/tcp --permanent
systemctl restart firewalld
```

40.2 Configuration de l'authentification kerberos

Note:

- Sans l'authentification kerberos, vous devez soit faire confiance à l'enregistrement initial, soit saisir un mot de passe pour chaque poste de travail lors de l'enregistrement initial.
- Pour plus d'informations, consultez la documentation sur l'enregistrement d'une machine auprès du serveur WAPT et *la signature des mises à jour d'inventaire*.
- L'authentification kerberos sera utilisée uniquement lors de l'enregistrement du dispositif.

40.2.1 Installation des composants kerberos et configuration du fichier krb5.conf

Debian / Ubuntu

```
apt install krb5-user msktutil libnginx-mod-http-auth-spnego
```

CentOS / RedHat

```
yum install krb5-workstation msktutil nginx-mod-http-auth-spnego
```

Note: L'enregistrement avec kerberos n'est pas disponible avec un serveur WAPT fonctionnant sous Windows.

Modifiez le fichier `/etc/krb5.conf` et **remplacez tout le contenu par les 4 lignes suivantes** en remplaçant **MYDOMAIN.LAN** par votre nom de domaine Active Directory (i.e. *<MYDOMAIN.LAN>*).

Attention: La valeur `default_realm` doit être écrit en **MAJUSCULES** ! !!

```
[libdefaults]
  default_realm = MYDOMAIN.LAN
  dns_lookup_kdc = true
  dns_lookup_realm=false
```

Récupérer un keytab de service. Utiliser les commandes **kinit** et **klist**. Vous pouvez utiliser un compte *Administrateur* ou tout autre compte ayant le droit délégué de joindre un ordinateur au domaine dans le conteneur de destination approprié (par défaut *CN=Computers*).

Dans la transcription shell ci-dessous, les commandes sont en noir et le texte renvoyé est commenté en gris clair :

```
sudo kinit administrator
## Password for administrator@MYDOMAIN.LAN:
## Warning: Your password will expire in 277 days on Mon. 17 sept. 2018 10:51:21 CEST
sudo klist
## Ticket cache: FILE:/tmp/krb5cc_0
## Default principal: administrator@MYDOMAIN.LAN
##
## Valid starting       Expires              Service principal
## 01/12/2017 16:49:31  02/12/2017 02:49:31  krbtgt/MYDOMAIN.LAN@MYDOMAIN.LAN
## renew until 02/12/2017 16:49:27
```

Si la demande d'authentification est réussie, vous pouvez alors créer votre Keytab HTTP avec la commande **msktutil**.

Veillez à modifier la chaîne *<DOMAIN_CONTROLER>* avec le nom de votre contrôleur de domaine (par exemple : **srvads.mydomain.lan**).

```
sudo msktutil --server DOMAIN_CONTROLER --precreate --host $(hostname) -b cn=computers --service␣
↪HTTP --description "host account for wapt server" --enctypes 24 -N
sudo msktutil --server DOMAIN_CONTROLER --auto-update --keytab /etc/nginx/http-krb5.keytab --host
↪$(hostname) -N
```

Attention: Assurez-vous d'avoir correctement configuré votre *nom d'hôte* de serveur WAPT avant d'exécuter ces commandes ;

Afin de vérifier votre *nom d'hôte*, vous pouvez exécuter **echo $(hostname)** et il **Doit**renvoyer le nom qui sera utilisé par l'agent WAPT exécuté sur les postes de travail clients. Si votre serveur WAPT est disponible sur Internet, vous devez ajouter un autre servicePrincipalName (SPN) pour qu'il corresponde à l'URL publique WAPT.

- Appliquez les droits d'accès appropriés au fichier `http-krb5.keytab`. Si vous avez un système d'exploitation basé sur Redhat avec selinux, veuillez fixer les droits avec **restorecon**.

Debian / Ubuntu

```
sudo chmod 640 /etc/nginx/http-krb5.keytab
sudo chown root:www-data /etc/nginx/http-krb5.keytab
```

CentOS / RedHat

```
sudo chown root:nginx /etc/nginx/http-krb5.keytab
sudo chmod 640 /etc/nginx/http-krb5.keytab
restorecon -v -R /etc/nginx/http-krb5.keytab
```

40.2.2 Post-configuration de kerberos pour le serveur WAPT

Vous pouvez maintenant utiliser le script de post-configuration pour configurer le serveur WAPT afin d'utiliser kerberos.

Le script de post-configuration va configurer **Nginx** et le serveur WAPT pour utiliser l'authentification kerberos.

Indication: Ce script de post-configuration doit être exécuté en tant que **root**.

```
/opt/wapt/waptserver/scripts/postconf.sh --force-https
```

L'authentification Kerberos sera maintenant configurée.

40.2.3 Cas particuliers d'utilisation

Mon serveur WAPT n'a pas accès à un Active Directory en écriture

- Connectez-vous à votre Active Directory (pas un RODC).
- Créez un compte d'ordinateur *srvwapt*.
- Ajouter un SPN (Service Principal Name) sur le compte *srvwapt$*.

```
setspn -A HTTP/srvwapt.mydomain.lan srvwapt
```

- Créer un keytab pour ce serveur WAPT.

```
ktpass -out C:\http-krb5.keytab -princ HTTP/srvwapt.mydomain.lan@MYDOMAIN.LAN rndpass -minpass 64 -
↪crypto all -pType KRB5_NT_PRINCIPAL /mapuser srvwapt$@MYDOMAIN.LAN
Reset SRVWAPT$'s password [y/n]? y
```

Note: Si l'adresse de votre serveur WAPT est différente de celle de votre domaine Active Directory, remplacez *HTTP/srvwapt.mydomain.lan@MYDOMAIN.LAN* par *HTTP/srvwapt.othername.com@MYDOMAIN.LAN*.

- Transférez ce fichier dans `/etc/nginx/` (avec **`winscp`** par exemple).
- Appliquez les droits d'accès appropriés au fichier `http-krb5.keytab`. Si vous avez un système d'exploitation basé sur Redhat avec selinux, veuillez fixer les droits avec **`restorecon`**.

Debian / Ubuntu

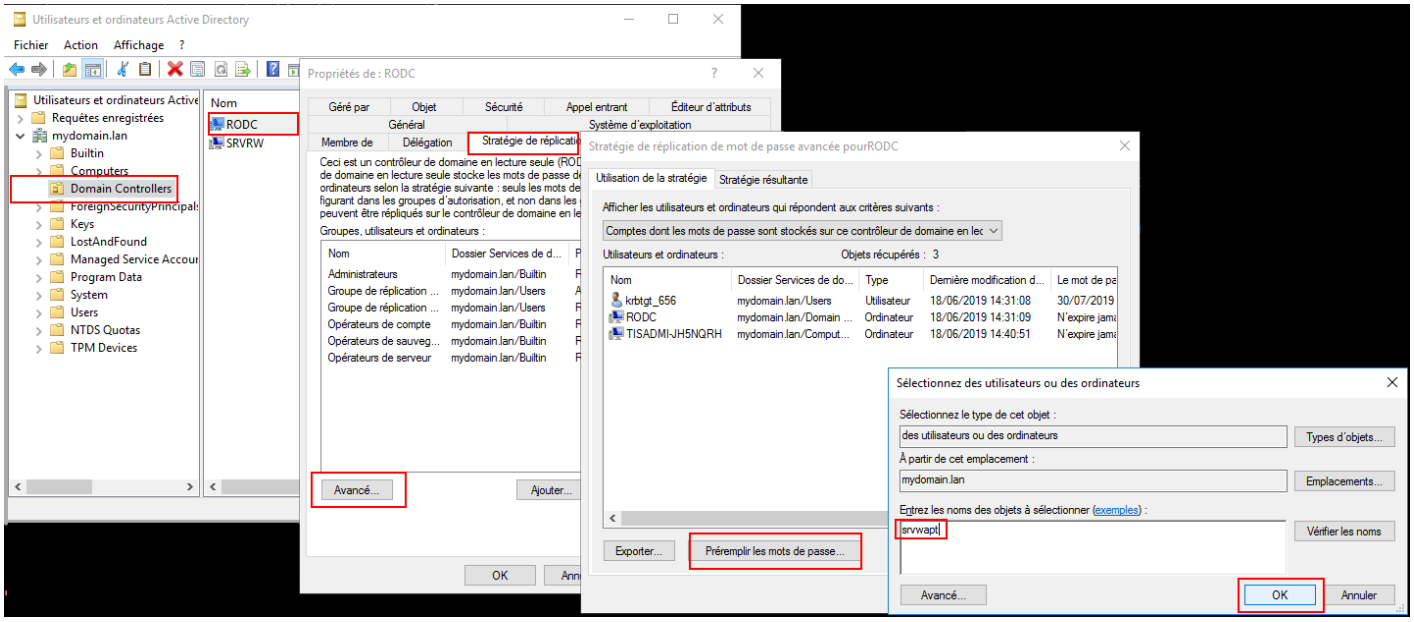
```
sudo chmod 640 /etc/nginx/http-krb5.keytab
sudo chown root:www-data /etc/nginx/http-krb5.keytab
```

CentOS / RedHat

```
sudo chown root:nginx /etc/nginx/http-krb5.keytab
sudo chmod 640 /etc/nginx/http-krb5.keytab
restorecon -v -R /etc/nginx/http-krb5.keytab
```

L'agent WAPT n'a accès qu'à un contrôleur de domaine RODC

- Pour RODC (Read-Only Domain Controller), ajoutez le compte *srvwapt* au groupe de mots de passe autorisés pour la réplication.
- N'oubliez pas de précharger le mot de passe du serveur WAPT avec les différents serveurs RODC.



Vous avez plusieurs domaines Active Directory, avec ou sans relations

Si vous avez plusieurs domaines Active Directory, vous devez créer un `keytab` par domaine en suivant la procédure ci-dessus, ex :

- `http-krb5-domain1.local.keytab`;
- `http-krb5-domain2.local.keytab`;
- `http-krb5-domain3.local.keytab`.

Vous devrez ensuite fusionner tous ces `keytabs` en un unique `keytab` :

```
ktutil
read_kt http-krb5-domain1.local.keytab
read_kt http-krb5-domain2.local.keytab
read_kt http-krb5-domain3.local.keytab
write_kt http-krb5.keytab
```

40.2.4 Débugger les problèmes avec les kerberos

Attention:

- L'adresse du serveur ne peut pas être une IP, Kerberos ne fonctionne bien qu'avec le DNS.
- Dans votre test, l'url utilisée doit être **exactement** la même adresse que celle indiquée dans `C:\Program Files (x86)\waptwapt-get.ini`.

Avez-vous redémarré nginx correctement ?

```
systemctl restart nginx
```

Vérifier les permissions du fichier http-krb5.keytab

```
[root@srvwapt.mydomain.lan]# ls -l /etc/nginx/http-krb5.keytab
-rw-r----- 1 root www-data 921 janv. 4 16:20 /etc/nginx/http-krb5.keytab
```

Le mode kerberos est-il actif sur mon agent ?

Sur la machine Windows :

- Vérifiez dans votre `C:\Program Files (x86)\wapt\wapt-get.ini` que la valeur use_kerberos est `True`.

```
[global]
use_kerberos=True
```

- Si vous modifiez cette valeur, n'oubliez pas de redémarrer le service WAPT.

```
net stop waptservice
net start waptservice
```

Le mode Kerberos est-il actif sur mon serveur ?

Sur la machine linux :

- Vérifiez dans votre `/opt/wapt/conf/waptserver.ini` que la valeur use_kerberos est `True`.

```
[options]
use_kerberos=True
```

- Vérifiez dans votre `/etc/nginx/sites-enabled/wapt.conf` que cette configuration est présente.

```
location ~ ^/.*_kerberos$ {

      proxy_http_version 1.1;
      proxy_request_buffering off;

      proxy_set_header Host $host;
      proxy_set_header X-Real-IP $remote_addr;
      proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;

      # be sure these headers are not forwarded
      proxy_set_header X-Ssl-Client-Dn "";
      proxy_set_header X-Ssl-Authenticated "";

      auth_gss on;
      auth_gss_keytab /etc/nginx/http-krb5.keytab;
      proxy_pass http://127.0.0.1:8080;

  }
```

- Si l'une des deux configurations n'est pas présente, redémarrez la post-configuration et activez kerberos.

Vérification que le fichier keytab contient l'url correcte

```
[root@srvwapt.mydomaine.lan]# KRB5_KTNAME=/etc/nginx/http-krb5.keytab klist -k
Keytab name: FILE:/etc/nginx/http-krb5.keytab
KVNO Principal
---- --------------------------------------------------------------------------
   ...
   3 HTTP/srvwapt.ad.mydomain.lan@AD.MYDOMAIN.LAN
   ...
```

Essayer d'enregistrer l'hôte en utilisant un compte système

Pour passer à un compte système, vous devez utiliser l'outil **psexe** de Microsoft : `psexe`.

- Dans **cmd** en tant qu'administrateur.

```
C:\Users\xxxxxx\Downloads\PSTools\psexec.exe -accepteula -s -i cmd
```

- Dans la nouvelle fenêtre **cmd**, vérifiez que vous êtes identifié comme *System*.

```
C:\WINDOWS\system32>whoami

NT AUTHORITY\System
```

- Exécutez la commande *register*.

```
wapt-get register
```

Tenter une authentification avec le keytab de votre serveur WAPT

- Sur la machine linux.

```
[root@srvwapt.ad.tranq ~]# ktutil
ktutil: read_kt /etc/nginx/http-krb5.keytab
ktutil: list
slot KVNO Principal
---- ---- ---------------------------------------------------------------------
   1    3                   srvwapt$@AD.TRANQUIL.IT
   2    3                   srvwapt$@AD.TRANQUIL.IT
   3    3                   srvwapt$@AD.TRANQUIL.IT
   4    3                   SRVWAPT$@AD.TRANQUIL.IT
   5    3                   SRVWAPT$@AD.TRANQUIL.IT
   6    3                   SRVWAPT$@AD.TRANQUIL.IT
   7    3               host/srvwapt@AD.TRANQUIL.IT
   8    3               host/srvwapt@AD.TRANQUIL.IT
   9    3               host/srvwapt@AD.TRANQUIL.IT
  10    3 HTTP/srvwapt.ad.tranquil.it@AD.TRANQUIL.IT
  11    3 HTTP/srvwapt.ad.tranquil.it@AD.TRANQUIL.IT
  12    3 HTTP/srvwapt.ad.tranquil.it@AD.TRANQUIL.IT
ktutil: quit
[root@srvwapt.ad.tranq ~]# kinit -k -t /etc/nginx/http-krb5.keytab srvwapt\$@AD.TRANQUIL.IT
[root@srvwapt.ad.tranq ~]# klist
Ticket cache: FILE:/tmp/krb5cc_0
Default principal: srvwapt$@AD.TRANQUIL.IT

Valid starting       Expires              Service principal
05/02/2021 19:06:05  06/02/2021 05:06:05  krbtgt/AD.TRANQUIL.IT@AD.TRANQUIL.IT
  renew until 06/02/2021 19:06:05
```

Tentative d'authentification avec curl

- Sur la machine linux.

```
[root@srvwapt.ad.tranq ~]# kdestroy
[root@srvwapt.ad.tranq ~]# kinit sfonteneau
Password for sfonteneau@AD.TRANQUIL.IT:
[root@srvwapt.ad.tranq ~]# klist
Ticket cache: FILE:/tmp/krb5cc_0
Default principal: sfonteneau@AD.TRANQUIL.IT

Valid starting       Expires              Service principal
05/02/2021 19:10:42  06/02/2021 05:10:42  krbtgt/AD.TRANQUIL.IT@AD.TRANQUIL.IT
  renew until 06/02/2021 19:10:39

root@srvwapt.ad.tranq ~]# curl -v --negotiate -u : https://srvwapt.ad.tranquil.it/add_host_kerberos␣
↪-k
* Expire in 0 ms for 6 (transfer 0x563dece09f90)
* Uses proxy env variable no_proxy == 'localhost,127.0.01/8,192.168.0.0/16,10.0.0.0/8,172.16.0.0/12,
↪ad.tranquil.it'
* Expire in 1 ms for 1 (transfer 0x563dece09f90)
 ...
* Expire in 0 ms for 1 (transfer 0x563dece09f90)
* Expire in 0 ms for 1 (transfer 0x563dece09f90)
*   Trying 192.168.149.37...
* TCP_NODELAY set
* Expire in 200 ms for 4 (transfer 0x563dece09f90)
* Connected to srvwapt.ad.tranquil.it (192.168.149.37) port 443 (#0)
* ALPN, offering h2
* ALPN, offering http/1.1
* successfully set certificate verify locations:
*   CAfile: none
  CApath: /etc/ssl/certs
* TLSv1.3 (OUT), TLS handshake, Client hello (1):
* TLSv1.3 (IN), TLS handshake, Server hello (2):
* TLSv1.2 (IN), TLS handshake, Certificate (11):
* TLSv1.2 (IN), TLS handshake, Server key exchange (12):
* TLSv1.2 (IN), TLS handshake, Request CERT (13):
* TLSv1.2 (IN), TLS handshake, Server finished (14):
* TLSv1.2 (OUT), TLS handshake, Certificate (11):
* TLSv1.2 (OUT), TLS handshake, Client key exchange (16):
* TLSv1.2 (OUT), TLS change cipher, Change cipher spec (1):
* TLSv1.2 (OUT), TLS handshake, Finished (20):
* TLSv1.2 (IN), TLS handshake, Finished (20):
* SSL connection using TLSv1.2 / ECDHE-RSA-AES256-GCM-SHA384
* ALPN, server accepted to use http/1.1 Server certificate:
*  subject: C=FR; ST=PDLL; L=Saint Sebastien sur Loire; O=Tranquil IT Systems; OU=TIS; CN=srvwapt.
↪ad.tranquil.it; name=PKI TIS; emailAddress=technique@tranquil.it
*  start date: Aug 3 12:48:03 2017 GMT
*  expire date: Aug 1 12:48:03 2027 GMT
*  issuer: C=FR; ST=PDLL; L=Saint Sebastien sur Loire; O=Tranquil IT Systems; OU=TIS; CN=Tranquil␣
↪IT Systems CA; name=PKI TIS; emailAddress=technique@tranquil.it
```

(suite sur la page suivante)

(suite de la page précédente)

```
*  SSL certificate verify ok.
> GET /add_host_kerberos HTTP/1.1
> Host: srvwapt.ad.tranquil.it
> User-Agent: curl/7.64.0
> Accept: */*
>
< HTTP/1.1 401 Unauthorized
< Server: nginx
< Date: Fri, 05 Feb 2021 18:08:38 GMT
< Content-Type: text/html
< Content-Length: 188
< Connection: keep-alive
< WWW-Authenticate: Negotiate
< WWW-Authenticate: Basic realm=""
<
* Ignoring the response-body
* Connection #0 to host srvwapt.ad.tranquil.it left intact
* Issue another request to this URL: 'https://srvwapt.ad.tranquil.it/add_host_kerberos'
* Uses proxy env variable no_proxy == 'localhost,127.0.01/8,192.168.0.0/16,10.0.0.0/8,172.16.0.0/12,
↪ad.tranquil.it'
* Found bundle for host srvwapt.ad.tranquil.it: 0x563dece07590 [can pipeline]
* Could pipeline, but not asked to!
* Re-using existing connection! (#0) with host srvwapt.ad.tranquil.it
* Connected to srvwapt.ad.tranquil.it (192.168.149.37) port 443 (#0)
* Expire in 0 ms for 6 (transfer 0x563dece09f90)
* Server auth using Negotiate with user ''
> GET /add_host_kerberos HTTP/1.1
> Host: srvwapt.ad.tranquil.it
> Authorization: Negotiate␣
↪IIGagYGKwYBBQUCoIIGXjCCBlqgDTALBgkqhkiG9xIBAgKiggZHBIIGQ2CCBj8GCSqGSIb3EgECAgEAboIGLjCCBiqgAwIBBaEDAgEOogcD
↪+lGapoxK1hCC62Mnye9zF4SzOEKBDLPD77KJp2xbW227lc5ZF/
↪22wGXn8n6Sw1xndf1brq1mSEUo0TzpFUfY1wNoRDaw7WUNhQK2nbTTeMrsiiPACuQtG82W0VrZJZ+4z1Gq3ZFTLYUrlC010S1T8pNRzCLFR
↪OBq9EQd+i/2Mfp8XWy46gGRtezTk8Dya+SH3henhB+L7G4ew0c3MKxFRkv0nXQ65qPAXWyogbivI/
↪ReekU1anHLnGfDyfeBw2QUM8t2kEEcSBmNKfrQ1U/u1jnlRZJ1o067PiziZsH7w/
↪zGpe7uhOa8a4RKYu1LeJEuU+CKrufulQWKuqdiwIBdq9ApoQqduCbNsE9ihH1srL3RYh9XdQ4Unx51eo49nZQA+c2Aj4JvCafbY/
↪jeRBw6SNYCrfGETN1mXytjLRyVBtJlch7djBGUAYaH1HGNfEVt+VnCW4O9OoqgC0M++u6d7Ci5w494ZseNXnF7RBKr01aVEt0231geGglNv
↪arutH6c3CzLb+xPMAOUtCIup0M43SR0DC7gJ/
↪xZ3BZyKHF6b3p3tAWiByat2XNMxfMbgjaiv7oLCNEIAga0IgTg5f0nlahTI9323vfIH8aLNNVYVJeFKNGX1ord0YpJ3RLDshNBnoDTPyCKn
↪QXiFQAezmRut+hfXtoVch8LHC0OIaloDUk1eHlFbAqQ98OaE3SZ8Fx4m8Nw3JgQ0E+zXPt8DJCnNY4YV4j3+9b1093XhsJRyP97qEFazUGF
↪PzqFnxSHtikXjCnjtzfuHLePMWn0HKFbL/hEMmAnfZ15JiBgfBi82OXv3rCui+GKT/ZsjUFsgR8tCUJ58/gXBu9J/
↪gY1R46CvWTnl03+2JHQyomm6k0XnAU+s2hX+n/QcKbIjT7ew/f/
↪UuT0J1YV+bQ8MMTpQqbau4f5sVaembIB7hTVyttpfbBEqCOVk39xZ/r8b9CMpmukShPgeJI0x353i7bO9/
↪mBkchFaeyOc45jA7Z4iJ3IHNiWLaWyYLktH/1N9/dXas8/CoZK0UsKjm9+xlkFFSP18CFHijILlIc1sTdMnAil/
↪jwqQl1W2WRnBlt8r/yE56EDR/i8VkCMHT5XsiiMCHm4LldkmDnIo/+GgHTG+3Z78Pkq939rMati/
↪gzd9geYM8aUuYwJpcb53YsjWvJD1gDEHEwS3K1MYxbyy9eiODCOCgvIeKmVPouNrugXs4TX6PJsCQDtzusSWxmZY4820HxmJkNT1lG5Zkkt
↪b93cbD4oaOWlbPViBpwLZ+TTCkeGAxo3eBicENHSK81EIJYMBfEWTTsjYPEPsl5BK7IFcqArfEWG6HQDw3bOfYAB1ZJb0zSbhyD/
↪rKnRmtSke/
↪eWIAjYaeHDX0qYMruJCuI2lYofHtFwMEKSB1jCB06ADAgESooHLBIHIDVUTdDas6nA0obxBuM2bQiZ0ZUPhAVGMoTniuCmBXU/
↪mFRAsD029zDjfl0nzeFsPdC4UBERcc8Vh4r3YeZIxUxzn5tXCW4oFypYi5kHAdx6Zd4GkZcEpzAhRF7JwSylerZiCF+fnSIiI5wdDG56PMF
> User-Agent: curl/7.64.0
```

(suite sur la page suivante)

(suite de la page précédente)

```
> Accept: */*
>
< HTTP/1.1 200 OK
< Server: nginx
< Date: Fri, 05 Feb 2021 18:08:38 GMT
< Content-Type: text/html; charset=utf-8
< Content-Length: 38
< Connection: keep-alive
< WWW-Authenticate: Negotiate␣
↪oYG3MIG0oAMKAQChCwYJKoZIhvcSAQICooGfBIGcYIGZBgkqhkiG9xIBAgICAG+BiTCBhqADAgEFoQMCAQ+iejB4oAMCARKicQRvQoZWpMI
↪x0oFJX6n4DnhPZxrq/RnjwkoTnik7R8MJkKRuvYncBfTGBIHvTJktq6+j9pHqmBDH5D5L8A
< WWW-Authenticate: Basic realm=""
< Cache-Control: store, no-cache, must-revalidate, post-check=0, pre-check=0
< Pragma: no-cache
<
* Closing connection 0
kerberos connection seems to be working
```

Vérification de la réussite de l'obtention d'un ticket Kerberos

Attention: Exécutez toujours les commandes dans le compte système (voir le point précédent) !

```
klist purge
klist get http/srvwapt.ad.mydomain.lan
```

Vous devez obtenir (dans votre langue) :

```
C:\Windows\System32>klist get http/srvwapt.ad.mydomain.lan

LogonId est 0:0x13794d
Un ticket pour http/srvwapt.ad.mydomain.lan a été récupéré.

Tickets mis en cache : (2)

#0> Client : sfonteneau @ AD.MYDOMAIN.LAN
  Serveur : krbtgt/AD.MYDOMAIN.LAN @ AD.MYDOMAIN.LAN
  Type de chiffrement KerbTicket : AES-256-CTS-HMAC-SHA1-96
  Indicateurs de tickets 0x40e00000 -> forwardable renewable initial pre_authent
  Heure de démarrage : 2/4/2021 15:51:07 (Local)
  Heure de fin :   2/5/2021 1:51:07 (Local)
  Heure de renouvellement : 2/11/2021 15:51:07 (Local)
  Type de clé de session : AES-256-CTS-HMAC-SHA1-96
  Indicateurs de cache : 0x1 -> PRIMARY
  KDC appelé : srvads.AD.MYDOMAIN.LAN

#1> Client : sfonteneau @ AD.MYDOMAIN.LAN
  Serveur : http/srvwapt.AD.MYDOMAIN.LAN @ AD.MYDOMAIN.LAN
```

(suite sur la page suivante)

(suite de la page précédente)

```
Type de chiffrement KerbTicket : AES-256-CTS-HMAC-SHA1-96
Indicateurs de tickets 0x40a80000 -> forwardable renewable pre_authent 0x80000
Heure de démarrage : 2/4/2021 15:51:07 (Local)
Heure de fin : 2/5/2021 1:51:07 (Local)
Heure de renouvellement : 2/11/2021 15:51:07 (Local)
Type de clé de session : AES-256-CTS-HMAC-SHA1-96
Indicateurs de cache : 0
KDC appelé : srvads.AD.MYDOMAIN.LAN
```

Si cela ne fonctionne pas, vérifier dans votre Active Directory que l'attribut `serviceprincipalname` sur le compte de l'ordinateur du serveur WAPT a cette valeur : `HTTP/srvwapt.mydomain.lan`.

Vérifiez qu'il fonctionne avec Firefox

Note: Vous devez d'abord configurer firefox pour l'authentification kerberos.

- Tapez **`about:config`** dans la barre d'URL de votre Firefox.
- Editez `network.negotiate-auth.trusted-uris`, et ajoutez l'url du serveur wapt : `srvwapt.mydomain.lan`.
- Vous pouvez maintenant visiter l'url : https://srvwapt.mydomain.lan/add_host_kerberos.
- Si l'authentification ne fonctionne pas, le serveur renvoie un message d'erreur 403.

En cas d'erreur lors d'un des contrôles précédents

- Supprimez le compte de la machine de l'Active Directory.
- Supprimez le fichier `/etc/nginx/http-krb5.keytab`.
- Redémarrez la machine avec laquelle vous effectuez le test et exécutez à nouveau le processus de création de la keytab.

Note:

- Il est important de redémarrer la machine pour purger les tickets kerberos précédemment obtenus par la machine.
- Pour éviter le redémarrage, vous pouvez également exécuter la commande « klist purge » en tant que SYSTEM.

40.3 Activation de la vérification du certificat SSL / TLS

Lors de l'exécution du script de post-configuration du serveur WAPT, le script générera un certificat auto-signé afin d'activer les communications HTTPS.

L'agent WAPT vérifie le certificat du serveur HTTPS en fonction de la valeur `verify_cert` de la section `[global]` dans `C:\Program Files (x86)\wapt\wapt-get.ini`.

Table1: Options pour `verify_cert`

Options pour `verify_cert`	Fonctionnement de l'agent WAPT
`verify_cert` = 0	l'agent WAPT ne vérifiera pas le certificat HTTPS du serveur WAPT.
`verify_cert` = 1	l'agent WAPT vérifiera le certificat HTTPS du serveur WAPT à l'aide du paquet de certificats `C:\Program Files (x86)\wapt\lib\site-packages\certifi\cacert.pem`
`verify_cert` = C:\Program Files (x86)\wapt\ssl\srvwapt.mydomain.lan.crt	l'agent WAPT vérifiera le certificat HTTPS du serveur WAPT avec le groupe de certificats `C:\Program Files (x86)\wapt\ssl\srvwapt.mydomain.lan.crt`

Indication: Pour activer rapidement et facilement la vérification du certificat https, vous pouvez utiliser la méthode *Pinning*.

40.3.1 Épingler le certificat

L'*épinglage de certificat* consiste à vérifier le certificat SSL/ TLS à l'aide de la définition d'un paquet bien défini et restrictif.

Indication: Cette méthode est la plus simple lorsqu'on utilise un certificat auto-signé.

Pour cela, vous devez lancer les commandes suivantes dans le shell Windows **`cmd.exe`** (avec des privilèges élevés si UAC (User Account Control) est actif).

Si vous avez déjà un shell Windows **`cmd.exe`** ouvert, fermez-le et ouvrez un nouveau shell afin de prendre en compte les variables d'environnement mises à jour :

```
wapt-get enable-check-certificate
wapt-get restart-waptservice
```

Validez le certificat avec **`wapt-get update`**

Lorsque vous avez exécuté la commande **`update`**, assurez-vous que tout s'est bien passé, et en cas de doute, vérifiez *Problème lors du enable-check-certificate*.

Attention: Si *wapt-get enable-check-certificate* renvoie une erreur, supprimez le `.crt` de même nom sur `C:\Program Files (x86)\wapt\ssl\server`

Note:

- La commande **`enable-check-certificate`** télécharge le certificat `srvwapt.mydomain.lan.crt` dans le dossier `C:\Program Files (x86)\WAPT\ssl\server`.
- Il modifie ensuite le fichier `wapt-get.ini` pour spécifier la valeur `verify_cert` = `C:\Program Files (x86)\wapt\ssl\server\srvwapt.mydomain.lan.crt`.
- L'agent WAPT va maintenant vérifier les certificats en utilisant le certificat épinglé.

Attention: Si vous utilisez la méthode d'*épinglage de certificat*, n'oubliez pas de sauvegarder le dossier `/opt/wapt/waptserver/ssl` sur votre serveur WAPT.

Le fichier devra être restauré sur votre serveur si vous migrez ou mettez à niveau votre serveur WAPT, si vous voulez que les agents WAPT puissent continuer à établir des connexions HTTPS de confiance.

40.3.2 Comment utiliser un certificat commercial ou des certificats fournis par votre organisation ?

Si la méthode d'épinglage ne vous convient pas, vous pouvez remplacer le certificat auto-signé généré lors de l'installation de **WAPT**.

Remplacez l'ancien certificat par le nouveau dans le dossier `/opt/wapt/waptserver/ssl/` (linux) ou `c:\wapt\waptserver\ssl\` (windows).

La nouvelle paire de clés doit être au format PEM encodé en Base64.

Note: Cas particulier où votre certificat a été signé par une Autorité de Certification interne

Les certificats émis par une *Autorité de certification* interne doivent avoir la chaîne de certificats complète de l' *Autorité de certification*.

Vous pouvez ajouter manuellement la chaîne de certificats de l'autorité de certification au certificat qui sera utilisé par **Nginx**.

Exemple : `echo srvwapt.mydomain.lan.crt ca.crt > cert.pem`

- Pour les serveurs Linux, il est également nécessaire de réinitialiser les ACLs, si vous êtes avec un OS basé sur Redhat avec selinux, veuillez fixer les droits avec **restorecon** :

Debian / Ubuntu

```
chown root:www-data /opt/wapt/waptserver/ssl/*.pem
```

CentOS / RedHat

```
chown root:nginx /opt/wapt/waptserver/ssl/*.pem
restorecon -v -R /opt/wapt/waptserver/ssl/
```

- Redémarrez **Nginx** pour prendre en compte les nouveaux certificats.

Linux

```
systemctl restart nginx
```

Windows:

```
net stop waptnginx
net start waptnginx
```

Configuration de l'agent WAPT

Pour un certificat commercial, vous pouvez définir `verify_cert = 1` dans `wapt-get.ini`.

Pour un certificat émis par une autorité de certification interne, vous devez placer le certificat dans le dossier `C:\Program Files (x86)\wapt\ssl\server\ca.crt` et spécifier le chemin du certificat avec `verify_cert` dans le fichier :*wapt-get.ini* de l'agent WAPT.

Pour appliquer la nouvelle configuration à l'ensemble de votre flotte :

- Régénérer un agent WAPT avec les paramètres appropriés.
- Utilisation du paquet WAPT pour modifier `wapt-get.ini` et pousser le certificat.

40.3.3 Vérification du certificat dans la console WAPT

Lorsque la console WAPT démarre pour la première fois, elle lit le contenu du fichier `C:\Program Files (x86)WAPT\wapt-get.ini` et elle construit son fichier de configuration `C:\Usersadmin\AppData\Local\waptconsole\waptconsole.ini`.

Ceci définit correctement l'attribut `verify_cert` pour la communication HTTPS entre la console WAPT et le serveur WAPT.

40.4 Configuration de l'authentification des utilisateurs par rapport à l'Active Directory

Par défaut, le serveur WAPT est configuré avec un seul compte *SuperAdmin* dont le mot de passe est défini lors de la post-configuration initiale.

Sur les réseaux étendus et sécurisés, ce compte SuperAdmin ne doit pas être utilisé car il ne peut pas fournir la traçabilité nécessaire aux actions administratives effectuées sur le réseau.

Il est donc nécessaire de configurer l'authentification par rapport à l'Active Directory pour les utilisateurs de la console WAPT ; cela permettra d'utiliser des comptes nommés pour les tâches.

Note:

- L'authentification Active Directory est utilisée pour authentifier l'accès à l'inventaire via la console WAPT.
- Cependant, toutes les actions sur les appareils distants équipés du WAPT sont basées sur des signatures X.509, donc un *Administrateur* aura besoin à la fois d'une connexion Active Directory **ET** d'une clé privée dont le certificat est reconnu par les appareils distants pour gérer sa base installée d'appareils utilisant WAPT.
- Seul le compte *SuperAdmin* et les membres du groupe de sécurité Active Directory **waptadmins** seront autorisés à télécharger des paquets sur le dépôt principal (mode d'authentification par login et mot de passe).

40.4.1 Activation de l'authentification Active Directory

- Pour activer l'authentification du serveur WAPT sur Active Directory, configurez le fichier `waptserver.ini` comme suit.

Note: Le fichier de configuration du serveur WAPT sur les systèmes GNU/ Linux et macOS se trouve dans `/opt/wapt/conf/waptserver.ini` ou dans `/opt/wapt/waptserver/waptserver.ini`.

Le fichier de configuration du serveur WAPT sur les systèmes Windows se trouve dans `C:\wapt\conf\waptserver.ini`.

```
#waptserver.ini

wapt_admin_group=waptadmins
ldap_auth_server=srvads.mydomain.lan
ldap_auth_base_dn=DC=mydomain,DC=lan
ldap_auth_ssl_enabled=False
```

Table2: Options d'authentification disponibles

Options / Valeur par défaut	Description	Exemple
`wapt_admin_group_dn` = (défaut [])	Définit le DN LDAP du groupe d'utilisateurs Active Directory autorisé à se connecter à la console WAPT.	wapt_admin_group_dn = CN=waptadmins,OU=groups,DC=ad
`wapt_admin_group` (défaut [])	Définit le sAMAccountName du groupe d'utilisateurs Active Directory autorisé à se connecter à la console WAPT, c'est une liste qui peut contenir plusieurs groupes. Vous pouvez utiliser cette option plutôt que `wapt_admin_group_dn`, mais **NE PAS UTILISER** les deux attributs en même temps.	wapt_admin_group = waptadmins, wapttech
`ldap_auth_server` (défaut `None`)	Définit le serveur d'authentification LDAP.	ldap_auth_server = srvads.mydomain.lan
`ldap_auth_base_dn` (défaut `None`)	Définit le DN de base de l'authentification LDAP.	ldap_auth_base_dn = dc=domain,dc=lan
`ldap_auth_ssl_enabled` (défaut `True`)	Désactive l'authentification SSL sur les connexions LDAP.	`ldap_auth_ssl_enabled` = False
`verify_cert_ldap` (défaut `True`)	Vérifie le certificat SSL pour les connexions LDAP, à moins que `ldap_auth_ssl_enabled` soit à **False** (sinon il ne fera rien).	verify_cert_ldap = False

- Redémarrer le service **`waptserver`**.

Avertissement: Pour **Microsoft Active Directory**, Microsoft a annoncé que l'authentification *SimpleBind* sur MS-AD sans SSL/TLS sera bloquée par défaut à partir d'avril 2020. Si vous n'avez pas de certificat installé, vous devrez modifier une clé de registre pour que l'authentification fonctionne.

Note: Par défaut **Samba-AD** ne permet pas l'authentification *SimpleBind* sans SSL/TLS. Si vous ne disposez pas d'un certificat valide, vous devrez modifier le paramètre `ldap server require strong auth` dans `/etc/samba/smb.conf`. Pour plus

d'informations, vous pouvez consulter la documentation de Tranquil IT sur https://dev.tranquil.it/samba/en/index.html.

40.4.2 Activer le Single Sign On (SSO) pour la console WAPT et le selfservice

Avertissement: Cette configuration n'est disponible que pour les serveurs sous Linux : CentOS, Debian ou Ubuntu.

You can use Kerberos to authenticate yourself on the **waptconsole** and the **selfservice**. This way, users do not need to enter their password.

Il n'est pas nécessaire d'enregistrer l'agent WAPT en utilisant kerberos pour utiliser le SSO (Single Sign-On) kerberos avec la Console WAPT et avec le Self-Service.

Préparer le serveur pour Kerberos Single Sign On

Attention: Pour activer Kerberos sur le serveur WAPT avec l'option `use_kerberos = True`, lancez le script *WAPT Server postconf*.

```
/opt/wapt/waptserver/scripts/postconf.sh
```

Veuillez consulter *la documentation sur la configuration de kerberos pour l'authentification* au préalable.

Si vous ne souhaitez pas utiliser Kerberos pour l'enregistrement des clients, mettre l'option `allow_unauthenticated_registration` à *True*.

Enfin, redémarrez les services waptserver et wapttasks.

```
systemctl restart waptserver wapttasks
```

Il existe 3 méthodes pour configurer votre serveur WAPT avec Kerberos et l'authentification LDAP.

Pour chacun d'entre eux, vous devrez modifier le fichier *waptserver.ini*.

1. **La première méthode** est la moins sécurisée .

Cette méthode ne vérifie pas le certificat LDAP et n'utilise pas de port sécurisé pour contacter le Serveur WAPT.

```
ldap_auth_ssl_enabled = False
verify_cert_ldap = False
```

En effet, `ldap_auth_ssl_enabled=False` n'essaiera pas de requêter l'Active Directory avec le protocole LDAPS.

L'option `verify_cert_ldap=False` est définie si vous n'utilisez pas *SSL/TLS support*.

Indication: Si votre serveur Active Directory est un Samba-AD et que vous avez cette option dans le `waptserver.ini`, alors le serveur Samba-AD refusera la connexion.

```
ldap_auth_ssl_enabled = False
```

Par défaut **Samba-AD** ne permet pas l'authentification *SimpleBind* sans SSL/TLS.

Si vous n'avez pas de certificat valide, vous devrez modifier le paramètre `ldap server require strong auth` dans `/etc/samba/smb.conf`.

Pour plus d'informations, vous pouvez consulter la documentation de Tranquil IT sur https://dev.tranquil.it/samba/fr/index.html.

2. **La deuxième façon** plus sûre mais pas parfaite.

Cette méthode permet l'authentification SSL sans vérifier le certificat.

```
ldap_auth_ssl_enabled = True
verify_cert_ldap = False
```

Le serveur WAPT essaiera d'utiliser le protocole LDAPS mais sans vérification de certificat pour contacter Active Directory.

3. **La méthode recommandée** est la plus sûre.

```
ldap_auth_ssl_enabled = True
verify_cert_ldap = True
```

- Mais pour faire fonctionner cela, vous aller devoir *activer le support SSL/TLS*.
- Ensuite, vous devrez ajouter ces options dans le fichier `waptserver.ini` :

  ```
  ldap_account_service_login = wapt-ldap@ad.tranquil.it
  ldap_account_service_password = PASSWORD
  ldap_auth_server = srvads.mydomain.lan
  ldap_auth_base_dn = dc=mydomain,dc=lan
  use_kerberos = True
  ```

  Les options `ldap_account_service_login` et `ldap_account_service_password` nécessitent un compte utilisateur dans votre Active Directory.

  Il n'est pas nécessaire que le compte de service ait des droits élevés, juste assez de droits pour lire les groupes et les membres des groupes. En d'autres termes, le serveur WAPT **DOIT** avoir des droits de lecture sur l'attribut `memberof` dans l'Active Directory.

- Puis redémarrez les services sur le serveur :

  ```
  systemctl restart waptserver wapttasks
  ```

Configuration de l'agent WAPT

Du côté du client, vous allez devoir vous assurer que ces 2 options sont définies dans *wapt-get.ini* :

```
service_auth_type = waptserver-ldap
use_kerberos = True
```

Il est possible de faire des changements dans `wapt-get.ini` manuellement ou en déployant un paquet WAPT avec les nouveaux paramètres de configuration.

Un paquet d'exemple est disponible dans le dépôt Tranquil IT.

Avec cette configuration, vous pouvez lancer votre console WAPT ou votre selfservice sans demander de mot de passe.

Pour que cette fonctionnalité fonctionne, l'Active Directory doit être disponible.

Note: La console WAPT continuera à vous demander un login/mot de passe : c'est tout à fait normal, de cette façon vous pouvez utiliser un autre utilisateur que votre utilisateur actuel dans votre session Windows.

Sinon, il vous suffit de mettre votre login et de cliquer sur `OK`.

40.4.3 Activez le support SSL/ TLS pour les connexions LDAP dans le Contrôleur de Domaine Active Directory

Par défaut, l'authentification sur Active Directory repose sur LDAP SSL (port 636 par défaut).

SSL/ TLS n'est pas activé par défaut sur Microsoft Active Directory tant qu'un certificat SSL n'a pas été configuré pour le contrôleur de domaine.

Note: Le serveur WAPT utilise les *paquets* d'autorité de certification du système d'exploitation (CentOS) pour valider la connexion SSL/ TLS à Active Directory.

Si le certificat Active Directory est auto-signé ou a été signé par une autorité de certification interne, vous devrez ajouter ces certificats au magasin de certificats.

Ajouter un *Autorité de Certification* dans le dossier `/etc/pki/ca-trust/source/anchors/` et mettez à jour le magasin des CA.

Debian / Ubuntu

```
cp cainterne.crt /usr/local/share/ca-certificates/cainterne.crt
update-ca-certificates
```

CentOS / RedHat

```
cp cainterne.crt /etc/pki/ca-trust/source/anchors/cainterne.crt
update-ca-trust
```

Windows

```
certutil -addstore -f "ROOT" cainterne.crt
```

- Une fois que vous avez configuré LDAP SSL/ TLS sur votre Active Directory (veuillez vous référer à la documentation de Microsoft pour cela), vous pouvez activer le support de la sécurité SSL/TLS pour AD dans `waptserver.ini`.

```
ldap_auth_ssl_enabled = True
```

- Redémarrer le service **`waptserver`**.

40.5 Configuration de l'authentification par certificat côté client

Si votre entreprise a besoin d'un serveur WAPT ouvert sur Internet, il peut être sécurisé grâce à l'authentification par certificat côté client.

Cette configuration restreint la visibilité du serveur WAPT aux seuls clients enregistrés. Cela se fait en s'appuyant sur la clé privée de l'agent WAPT générée lors de l'enregistrement. Elle fonctionne comme suit :

- L'agent WAPT envoie un CSR (Certificate Signing Request) au serveur WAPT qui le signe et le renvoie à l'agent WAPT.
- Grâce au certificat signé, l'agent peut accéder aux parties protégées du serveur Web **Nginx**.

Note: Nous recommandons fortement d'activer l'enregistrement Kerberos ou par login/mot de passe dans la post-configuration du serveur WAPT.

Avertissement: Toutes les actions sont à mener sur le serveur WAPT

40.5.1 Activation de l'authentification des certificats côté client sur le serveur WAPT

Avertissement: Pour **Linux**, vérifiez si le lien symbolique dans `sites-enabled` existe :

```
cd /etc/nginx/sites-enabled/
find . -maxdepth 1 -type l -ls
```

Le résultat escompté devrait être:

```
269091      0 lrwxrwxrwx   1 root     root           36 juil. 22 15:51 ./wapt.conf -> /etc/nginx/
↪sites-available/wapt.conf
```

Sinon, utilisez la commande suivante:

```
ln -s /etc/nginx/sites-available/wapt.conf ./wapt.conf
```

Pour activer l'authentification, vous devez ajouter ces paramètres dans *le fichier de configuration* du serveur WAPT dans la section option :

```
use_ssl_client_auth = True
```

Relancez le script post-configuration .

Attention: Attention, à la date du 2023-04-17, WAPT ne supporte pas les CRL, ce qui signifie que lorsque vous supprimez une machine dans la console WAPT, la machine aura toujours accès au dépôt WAPT.

WAPTDeploy ne peut pas utiliser le https pour récupérer l'agent WAPT, vous devrez ajouter cette section dans le fichier:

```
server {
  listen                  80;
  listen                  [::]:80;
  server_name             _;
```

```
  location ~ ^/(wapt/waptsetup-tis.exe|wapt/waptagent.exe|wapt/waptdeploy.exe)$ {
      add_header Cache-Control "store, no-cache, must-revalidate, post-check=0, pre-check=0";
      add_header Pragma "no-cache";
      root "/var/www";
  }

  return 301                  https://$host$request_uri;
}
```

40.6 Génération de l'autorité de certification (CA)

Lors de l'installation de WAPT, il vous est demandé de créer une paire *.pem* / *.crt* en cochant les cases *Pour Signature de code* et *Pour usage en tant que CA*.

Cette paire *.pem* / *.crt* permettra de signer les paquets WAPT et les nouveaux certificats.

40.6.1 Generating a new certificate with the Certificate Authority

Construire une nouvelle paire `.pem` / `.crt`.

Note: Le nouveau certificat ne sera pas un certificat auto-signé ;

Ce nouveau certificat sera signé par le CA (la clé générée lors de la première installation de WAPT) ;

Vous devez ensuite remplir la *Clé privée de l'autorité* et le *Certificat de l'autorité*.

Lors de la génération de la nouvelle paire pem/ crt, vous avez la possibilité de choisir si le nouveau certificat sera de type **Pour Signature de code** ou non.

Indication: Pour rappel, un certificat *Pour Signature de code* est réservé aux personnes ayant le rôle *Administrateur* dans le contexte de WAPT et un simple certificat SSL sans l'attribut *Pour Signature de code* est réservé aux personnes ayant le rôle *Déployeur de paquet*.

L'*Administrateurs* sera autorisé à signer les paquets qui **CONTIENNENT** un fichier exécutable `setup.py` (c'est-à-dire les paquets *base*).

Les personnes ayant le rôle *Déployeur de paquet* seront autorisées à signer les paquets qui **NE CONTIENNENT PAS** le fichier exécutable `setup.py` (c'est-à-dire les paquets *host*, *unit* et *group*).

Les clés et les certificats qui ne sont pas **Signature de code** peuvent être distribués aux personnes chargées de déployer les paquets sur la base installée des appareils équipés de WAPT.

Une autre équipe disposant de certificats ayant l'attribut **Pour Signature de code** préparera les paquets WAPT contenant les applications qui devront être configurées conformément aux directives de sécurité de l'*Organisation* et aux personnalisations utilisateur souhaitées par celle-ci.

La génération d'une nouvelle paire .pem / .crt permettra également d'identifier formellement la personne qui a signé un paquet en recherchant l'attribut CN du certificat de paquet WAPT.

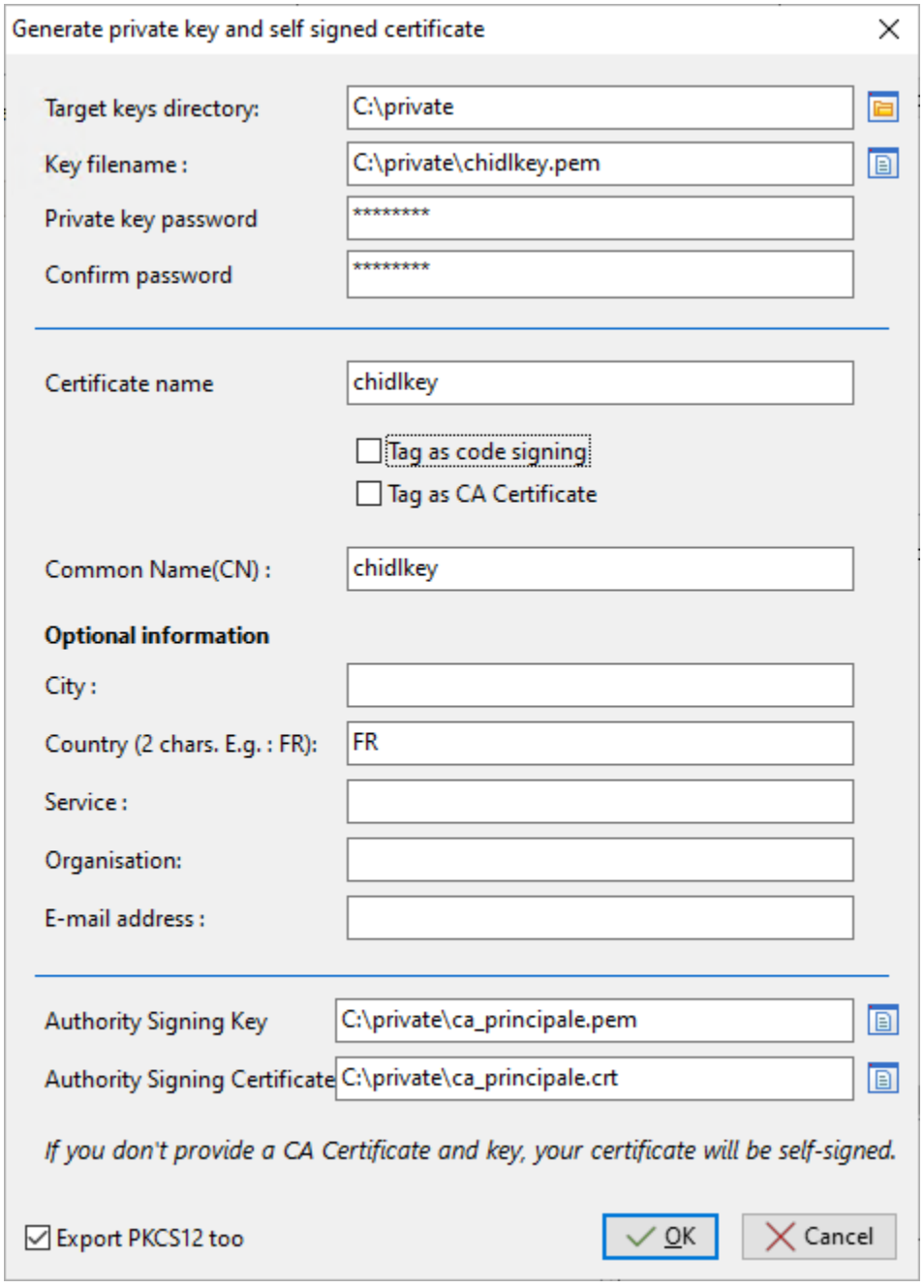


Figure1: Génération d'un certificat sans l'attribut *Pour Signature de code*

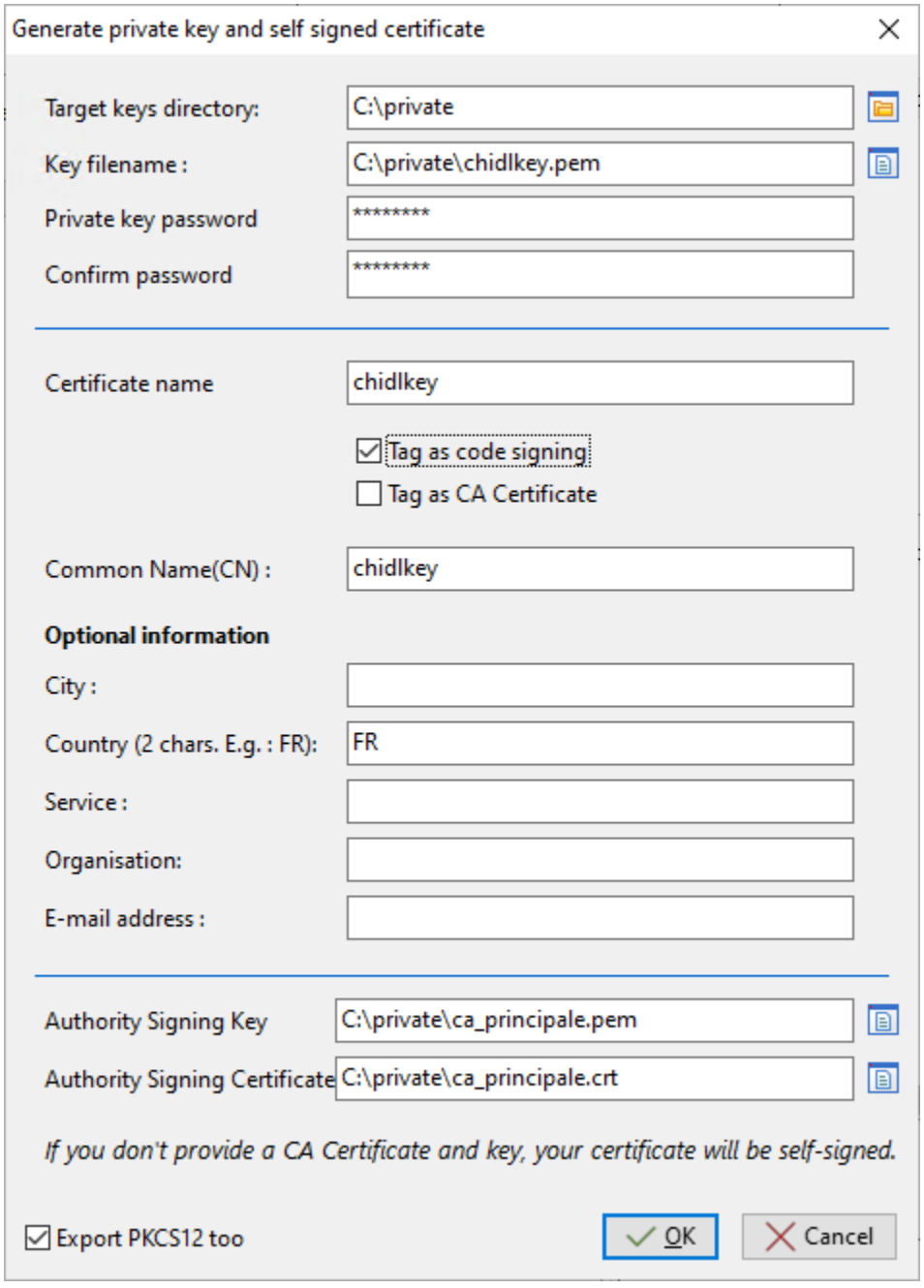


Figure2: Génération d'un certificat avec l'attribut *Pour Signature de code*

Indication: Les nouveaux certificats ne seront pas des *Autorités de Certification*, ce qui signifie qu'ils ne seront pas autorisés à signer d'autres certificats.

En règle générale, il n'y a qu'une seule paire pem / crt d'**Autorité de Certification** par *Organisation*.

Attention: Il n'est pas nécessaire de déployer des certificats enfants avec l'agent WAPT.

Les certificats enfants sont utilisés avec la console WAPT pour autoriser ou restreindre les actions dans la console.

40.7 Déploiement des certificats des administrateurs informatiques locaux sur les clients

Indication: Certaines organisations choisiront de laisser les administrateurs informatiques locaux effectuer des actions sur les appareils équipés de WAPT en leur délivrant des certificats personnels qui fonctionneront sur l'ensemble des appareils dont les administrateurs informatiques locaux sont responsables.

Les administrateurs informatiques du siège déploieront les certificats des administrateurs informatiques locaux sur les ordinateurs que les administrateurs locaux gèrent sur leurs sites respectifs.

Ainsi, les administrateurs informatiques locaux ne pourront pas gérer les ordinateurs situés au siège, mais uniquement sur leurs propres sites.

Il est possible de gérer simplement et de manière plus fine en utilisant *Access Control Lists* avec la version Enterprise de WAPT.

Vous devrez copier les certificats des administrateurs informatiques locaux autorisés sur les clients WAPT dans `C:\program files(x86)\wapt\ssl`.

Indication: Ne pas oublier de redémarrer le service WAPT sur les clients pour qu'ils utilisent leur nouveau certificat. Ouvrez une console en ligne de commande **`cmd.exe`**.

```
net stop waptservice && net start waptservice
```

Si vous voulez déployer les certificats en utilisant WAPT, utilisez un paquet de certificat

40.8 Configuration des listes de contrôle d'accès

Indication: L'utilisateur *SuperAdmin* de WAPT est authentifié par un mot de passe stocké dans `waptserver.ini` comme valeur de l'attribut `wapt_password`. Les autres utilisateurs WAPT peuvent être des utilisateurs locaux `htpasswd_path`) ou des utilisateurs de comptes AD (`ldap_auth_server` / `ldap_auth_base_dn`).

Les ACL définissent les actions autorisées pour tous les types d'utilisateurs dans le contexte WAPT.

Note: Les ACLs par défaut au niveau utilisateur sont définies par `default_ldap_users_acls` dans `waptserver.ini`.

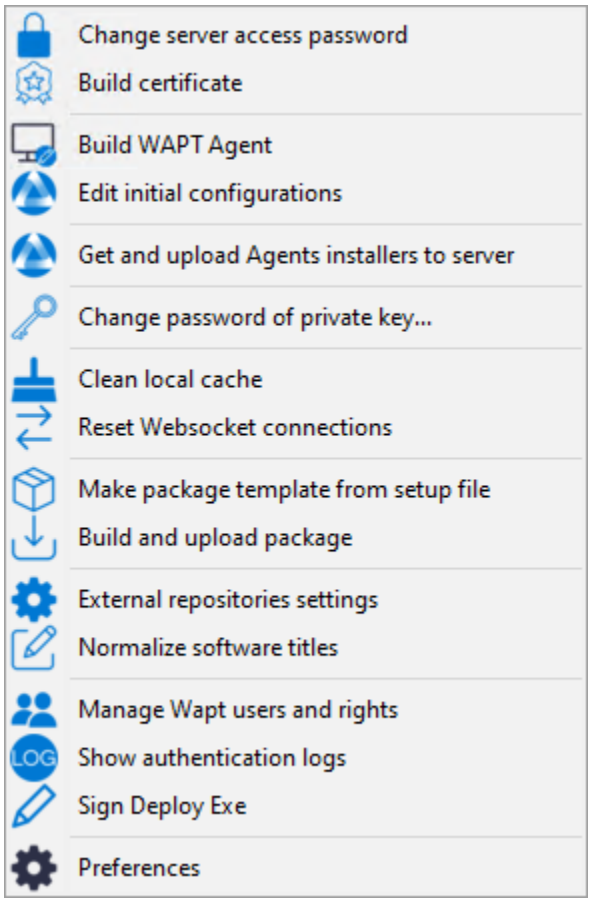
L'ACL par défaut pour un nouvel utilisateur est `vue`.

Attention: La sécurité est définie par le certificat déployé sur les clients, et non par les ACL.

Les ACL limitent simplement les actions que le serveur est autorisé à relayer de la console WAPT aux agents WAPT.

A la date du |date|, les agents WAPT ne vérifient pas les droits ACL.

Pour configurer les ACL dans WAPT, allez dans *Outils* → *Gérer les utilisateurs Wapt et des droits*.



Note: Au premier lancement après l'installation du serveur, seul le compte *SuperAdmin* est présent dans la liste des utilisateurs.

Si le compte *SuperAdmin* n'existe pas ou ne possède pas le droit *admin*, le compte est recréé en redémarrant le service waptserver.

Le compte *SuperAdmin* est authentifié en utilisant la valeur de `wapt_password` dans le fichier de configuration `waptserver.ini`.

Deux types de comptes sont gérables par ACL, *local* et *Active Directory*.

40.8.1 Compte d'utilisateur local

Les utilisateurs locaux sont définis par un fichier *.htpasswd*.

Configuration du serveur WAPT

Pour utiliser un compte utilisateur local, vous devez créer un fichier nommé `waptusers.htpasswd` dans le même *dossier* sur le serveur WAPT contenant le fichier `waptserver.ini`.

Linux:

```
touch /opt/wapt/conf/waptusers.htpasswd
chown wapt /opt/wapt/conf/waptusers.htpasswd
```

Windows

```
cd. > C:\wapt\conf\waptusers.htpasswd
```

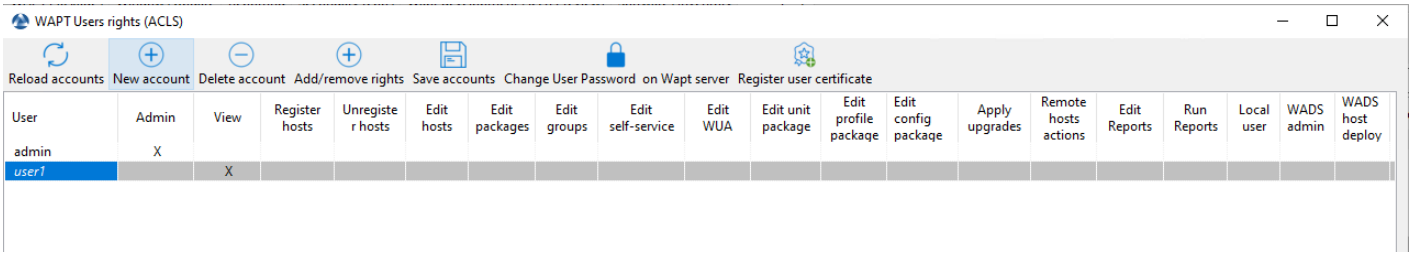
- Sur `waptserver.ini` ajoutez les paramètres `htpasswd_path`.

```
htpasswd_path = password file location
```

Indication: Redémarrer le service **waptserver**

Création du compte utilisateur

- Dans la fenêtre *Droits des utilisateurs WAPT*, cliquez sur *Nouveau compte*.



Il est possible de renommer des comptes en appuyant sur `F2` sur la colonne *User*.

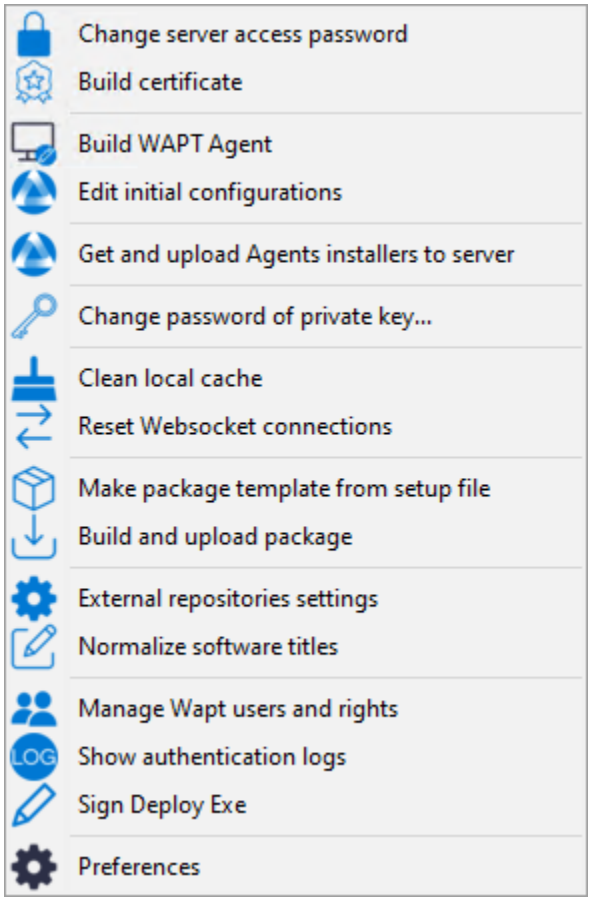
- Sauvegardez en cliquant sur *Enregistrer les comptes*.
- Pour définir un mot de passe, voir le point **Changez le mot de passe**.
- Pour définir les droits, consultez la section *gérer les droits ACL*.

Si l'utilisateur local a un mot de passe dans `waptusers.htpasswd`, alors, le nom d'utilisateur apparaît en **gras** et *Mots de passe* est coché, sinon changez le mot de passe pour cet utilisateur.

Changer le mot de passe de l'utilisateur

Pour changer le mot de passe du compte sélectionné :

- Faites un *clic droit sur le compte* → *Changer le mot de passe utilisateur sur le serveur Wapt*.



- Saisissez le nouveau mot de passe.

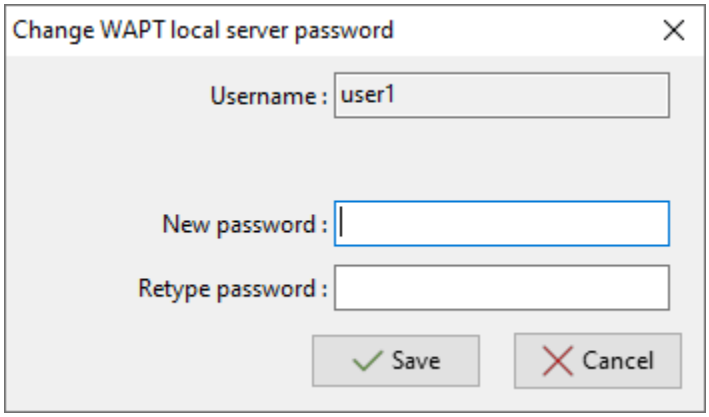


Figure3: Boîte de dialogue permettant de modifier le mot de passe de l'utilisateur dans le fichier htaccess

L'utilisateur local apparaît en *gras* et la case *Mots de passe* est cochée.

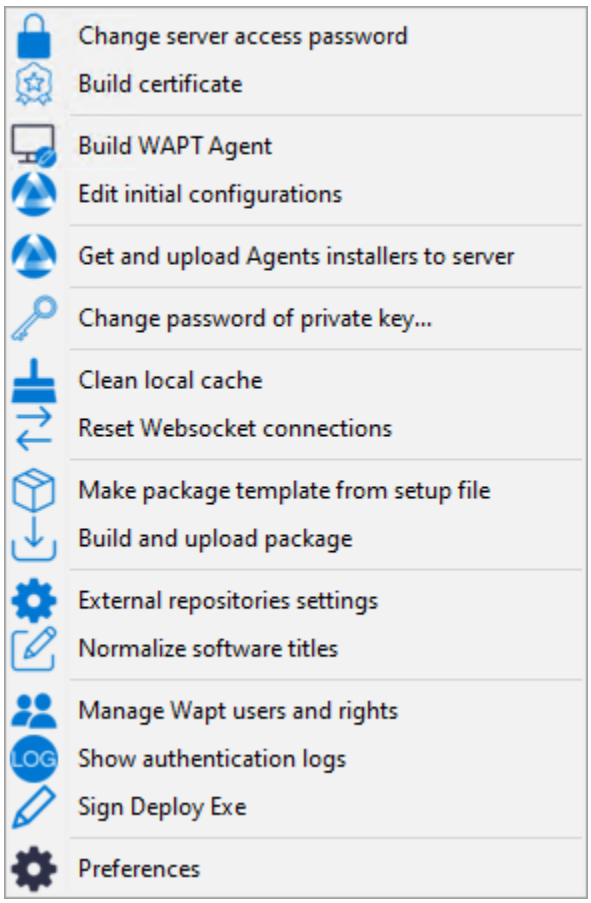
40.8.2 Utilisateurs WAPT définis comme utilisateurs Active Directory

Pour gérer les utilisateurs WAPT avec votre Active Directory, vous devez activer l'"*authentification Active Directory*.

Après une première connexion réussie, le compte AD apparaîtra automatiquement dans la liste des utilisateurs WAPT.

40.8.3 Blocage des comptes d'utilisateurs locaux

Pour désenregistrer les utilisateurs locaux, faites *clic droit sur le compte → Invalider le mot de passe de l'utilisateur sur le serveur WAPT*.



Le compte sera bloqué et ne pourra plus gérer quoi que ce soit dans WAPT.

40.8.4 Liste des droits

De nombreux *droits et restrictions* peuvent être définis pour chaque utilisateur dans la console WAPT.

Table3: Liste des droits des utilisateurs

Droit	Description
Admin	Comme SuperAdmin, tous les droits sont accordés sauf *Mot de passe*.
Voir	Permet de visualiser uniquement les informations sur la console WAPT.
Inscrire machine	Permet d'utiliser les informations d'identification de l'administrateur pour *enregistrer manuellement une machine* avec le Serveur WAPT.
Désinscrire machine	Permet de *supprimer une machine* depuis la console WAPT.
Modif machine	Permet de *modifier le paquet machine* sur la console WAPT.
Modif paquets	Permet de *modifier les paquets de base* qu'elle est autorisée à modifier.
Modif groupes	Permet de *modifier les paquets de groupe* sur la console WAPT.
Modif self-service	Permet de *modifier les règles de self-service* sur la console WAPT.
Modif WUA	Permet de *modifier les règles WUA / WSUS* sur la console WAPT.
Modif paquets AD OU	Permet de *modifier les paquets unit* sur la console WAPT.
Modif paquets Profile	Permet de *modifier les packages profile* sur la console WAPT.
Lancer les instalations	Permet d'appliquer à distance des mises à jour sur son périmètre, si la machine est en statut **PENDING**.
Actions distantes machine	Permet d'utiliser les outils Windows de Gestion de l'ordinateur avec la console WAPT.
Modifier requêtes	Permet de *créer ou modifier des requêtes de rapport*.
Lancer requête	Permet de *exécuter des rapports SQL existants*.
Mot de passe	Définit un utilisateur local

40.8.5 Gestion des droits

Par défaut, le **SuperAdmin** est l'utilisateur du *Certificat CA*.

Pour les autres utilisateurs, il est possible d'associer un certificat qui a été généré à partir de la PKI WAPT ou d'une autre CA.

Ces certificats peuvent ou non être des enfants de l'autorité de certification WAPT.

Attention: Si les certificats ne sont pas émis par l'autorité de certification :

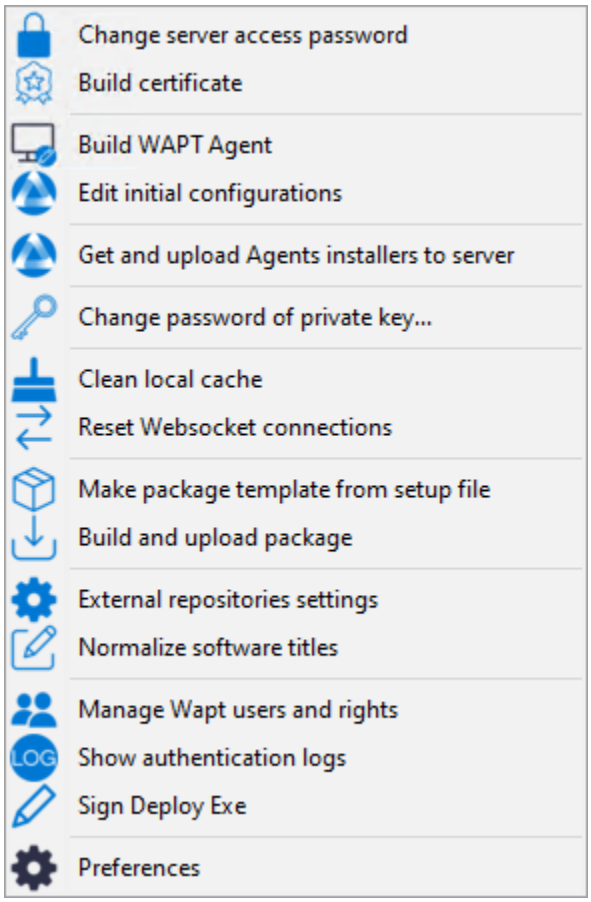
- Les paquets mis à jour sont disponibles uniquement sur les ordinateurs où les certificats sont déployés.
- Les ACL sont valides uniquement sur le périmètre des hôtes où le certificat de l'administrateur est déployé.

Associer un certificat à un utilisateur

Indication: Par défaut, aucun certificat n'est défini pour aucun utilisateur (y compris *SuperAdmin*).

Le compte dans la console WAPT apparaît en *italic* si aucun certificat n'est associé à l'utilisateur.

Pour associer un certificat à un utilisateur, faites *clic droit sur l'utilisateur* → *Associer un certificat à l'utilisateur*.



Ensuite, choisissez le certificat à associer à l'utilisateur.

Ajouter / supprimer des droits

Pour ajouter ou supprimer des droits, sélectionnez la cellule avec *clic gauche* et cochez-la en appuyant sur la `barre d'espace`.

Indication: Il est possible d'effectuer une sélection multiple en utilisant les raccourcis clavier `control+clic gauche` et en appuyant sur la `barre d'espace`.

Restreindre le périmètre des droits accordés à l'utilisateur

Il est possible d'associer un périmètre à un droit donné à un utilisateur.

Vue

Table4: Définition du périmètre autorisé

Périmètre	Description
Tout refuser	Aucun droit de regard n'est autorisé (non coché).
Autoriser sur tout le périmètre	Permet de visualiser à droite tous les agents WAPT.
Autoriser des périmètres spécifiques	La visualisation est autorisée sur le périmètre sélectionné défini comme une liste de certificats.
Autoriser où le certificat d'utilisateur est déployé	La visualisation est autorisée uniquement sur le périmètre où le certificat de l'administrateur est déployé.

Modifier les paquets de groupe

Indication: Tous les paquets de groupe fonctionnent sur le même principe, décrit ci-dessous.

Table5: Définition du périmètre autorisé

Périmètre	Description
Interdire tous les paquets	Aucune édition n'est autorisée pour aucun paquet (non coché).
Autoriser tous les paquets	Le droit de modification est autorisé pour tous les paquets.
Autoriser des noms de paquets spécifiques	Permet le droit d'édition pour les packages WAPT sélectionnés dans la liste.

CHAPTER 41

Simplifier le déploiement de vos postes de travail

De nombreuses entreprises et administrations incluent des logiciels et des configurations dans les images d'OS qu'elles déploient sur leurs flottes de machines.

Mais désormais ce n'est plus la méthode recommandée pour plusieurs raisons :

- Chaque fois que vous créez une nouvelle image, vous perdez beaucoup de temps à installer un logiciel et à le configurer. Vous êtes très limité dans les paramètres que vous pourrez inclure dans votre image.
- Chaque fois que vous créez une nouvelle image, vous devrez suivre les modifications dans un document texte, une feuille de calcul ou un outil de gestion des modifications.
- Les éditeurs de systèmes d'exploitation (notamment Microsoft) conseillent l'utilisation d'images ISO brutes et leur paramétrage en post-installation.
- Enfin, si vous introduisez dans votre image des configurations de sécurité, des configurations réseau ou des configurations pour limiter l'intrusion de la télémétrie, ces configurations peuvent perturber le fonctionnement normal de WAPT, cela compliquera les diagnostics futurs.

Avec WAPT, ce n'est plus nécessaire

41.1 Recommandations

Tranquil IT recommande :

- De réaliser une seule image brute par type d'OS avec MDT, Fog (win10, win2016, etc) ou *WAPT WADS* sans aucune configuration ou logiciel. **Mettez uniquement les pilotes système** dont vous avez besoin pour le déploiement de votre image dans les répertoires MDT ou Fog prévus à cet effet ;
- Pour créer autant d'Unités Organisationnelles que vous avez de types de machines dans l'OU *CN=Computers* (ex : *standard_laptop*, *hardened_laptop*, *workstations*, *servers*, etc) dans votre Active Directory ;

- Pour configurer votre Active Directory afin de distribuer la GPO de l'agent WAPT aux différentes Unités Organisationnelles de machines ; De cette façon, vous pouvez opter pour des configurations fines de votre `waptagent.ini` pour les hôtes rattachés à chaque OU.

Indication: Pour vous faire gagner du temps, vous pouvez baser votre stratégie de configuration de la sécurité sur les paquets WAPT de sécurité déjà disponibles dans le WAPT Store, vous n'aurez qu'à les compléter en fonction des exigences de sécurité spécifiques de votre Organisation.

- Créer dans l'OU *CN=Ordinateurs* autant d'Unités Organisationnelles qu'il y a de types d'utilisation des ordinateurs dans votre organisation (*comptabilité*, *point_de_vente*, *ingénierie*, *vente_sédentaire*, etc).
- Pour créer des paquets WAPT génériques de vos applications logicielles avec leurs configurations associées.

41.1.1 Scénario de déploiement

- Vous recevez ou le responsable informatique du site distant reçoit une nouvelle machine dans sa boîte.
- Vous configurez l'adresse MAC de la machine par DHCP afin qu'elle reçoive la bonne image système et soit placée dans la bonne Unité Organisationnelle à la fin du processus de déploiement.
- L'image système attendue est téléchargée sur la machine en temps masqué, la machine est placée dans la bonne Unité Organisationnelle.
- L'agent WAPT enregistre la machine auprès du serveur WAPT, elle apparaît dans la console WAPT.
- L'agent WAPT détecte qu'il se trouve dans une unité organisationnelle qui nécessite un ensemble de logiciels particulier et une configuration de sécurité particulière.
- L'agent WAPT télécharge et exécute des logiciels et des progiciels de configuration de sécurité en temps masqué ; l'agent WAPT supprime automatiquement les droits délégués qui sont rendus inutiles après avoir rejoint le domaine pour éviter qu'ils ne soient ensuite exploités de manière non autorisée.
- Soit par groupe de machines ou machine par machine, vous finalisez la configuration des machines en leur attribuant des paquets WAPT spécifiques.

Indication: Si vous le souhaitez, vous pouvez même laisser l'étape finale de configuration à vos utilisateurs en configurant le libre-service WAPT pour eux (configuration des imprimantes, besoins logiciels spéciaux, etc).

CHAPTER 42

Déployer vos postes de travail via WADS |enterprise_feature||

WADS pour WAPT Automated Deployment Services a été développé pour fournir une solution simple pour les déploiements de systèmes d'exploitation via WAPT.

Indication: A la date d'aujourd'hui, seul le déploiement des versions Windows est disponible.

42.1 Mode de fonctionnement du WADS

Schématiquement, le déploiement d'un OS implique **3**étapes :

1. Importation des différents supports et fichiers nécessaires au déploiement, tels que les images du système d'exploitation `.iso`, les packs de pilotes et les fichiers de configuration.

1. Création du support de démarrage.

3. Lancement du déploiement sur l'hôte cible en utilisant le réseau ou une clé USB.

42.1.1 Différence entre l'AMED et les autres solutions

- **Solution de déploiement classique**.
- **Solution de déploiement WADS**.

Indication:

- Le mode de fonctionnement de WADS respecte la méthode recommandée par le fournisseur du système d'exploitation.
- Avec WADS, toutes les fonctionnalités sont regroupées sur le même serveur WAPT.
- Il n'est donc pas nécessaire de mettre en place une infrastructure supplémentaire autre que le serveur WAPT.

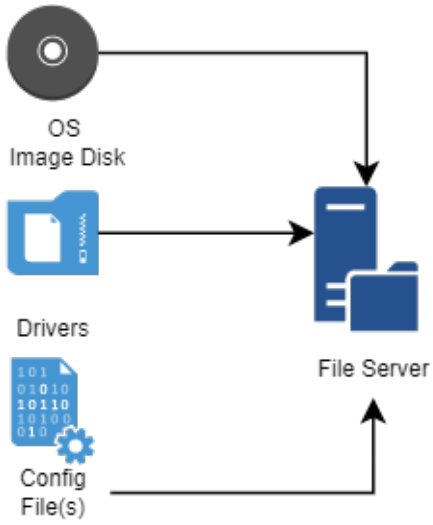


Figure1: Diagramme de flux pour l'importation des fichiers requis pour le déploiement de WADS

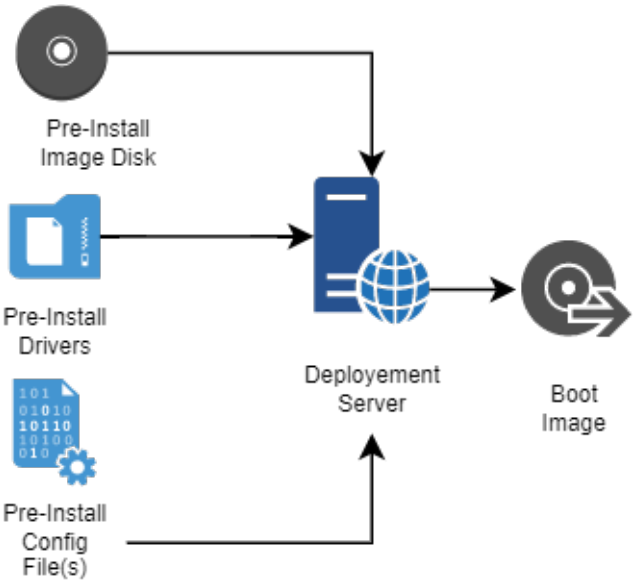


Figure2: Diagramme de flux pour la création du support d'amorçage pour le déploiement WADS

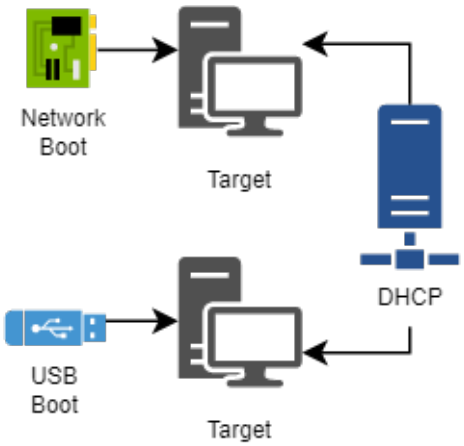


Figure3: Diagramme de flux pour l'utilisation du support de démarrage dans le déploiement WADS

Différences entre les logiciels

Table1: Différences entre la méthode WADS et les autres méthodes

Serveur de déploiement WADS	Méthode PCT standard	Prestations
Utilise iPXE	Utilise le protocole de serveur de fichiers CIFS (Common Internet File System)	Il n'est pas nécessaire de configurer un serveur de fichiers ni d'ouvrir des ports supplémentaires
Aucune configuration de l'image du OS n'est nécessaire	Nécessite de modifier manuellement la configuration d'un fichier de réponses	Simplicité, toutes les configurations sont fournies par WAPT
Utilise HTTPS pour télécharger l'image du système d'exploitation Windows	Utilise CIFS pour télécharger l'image du système d'exploitation Windows	Les hôtes cibles peuvent être déployés sur Internet en utilisant la méthode de la clé USB
La méthode WADS incorpore tous les fichiers nécessaires	La méthode MDT nécessite l'assemblage de fichiers provenant de sources différentes	Le déploiement, la configuration et les mises à jour du système d'exploitation sont regroupés dans un seul packaging logiciel WAPT

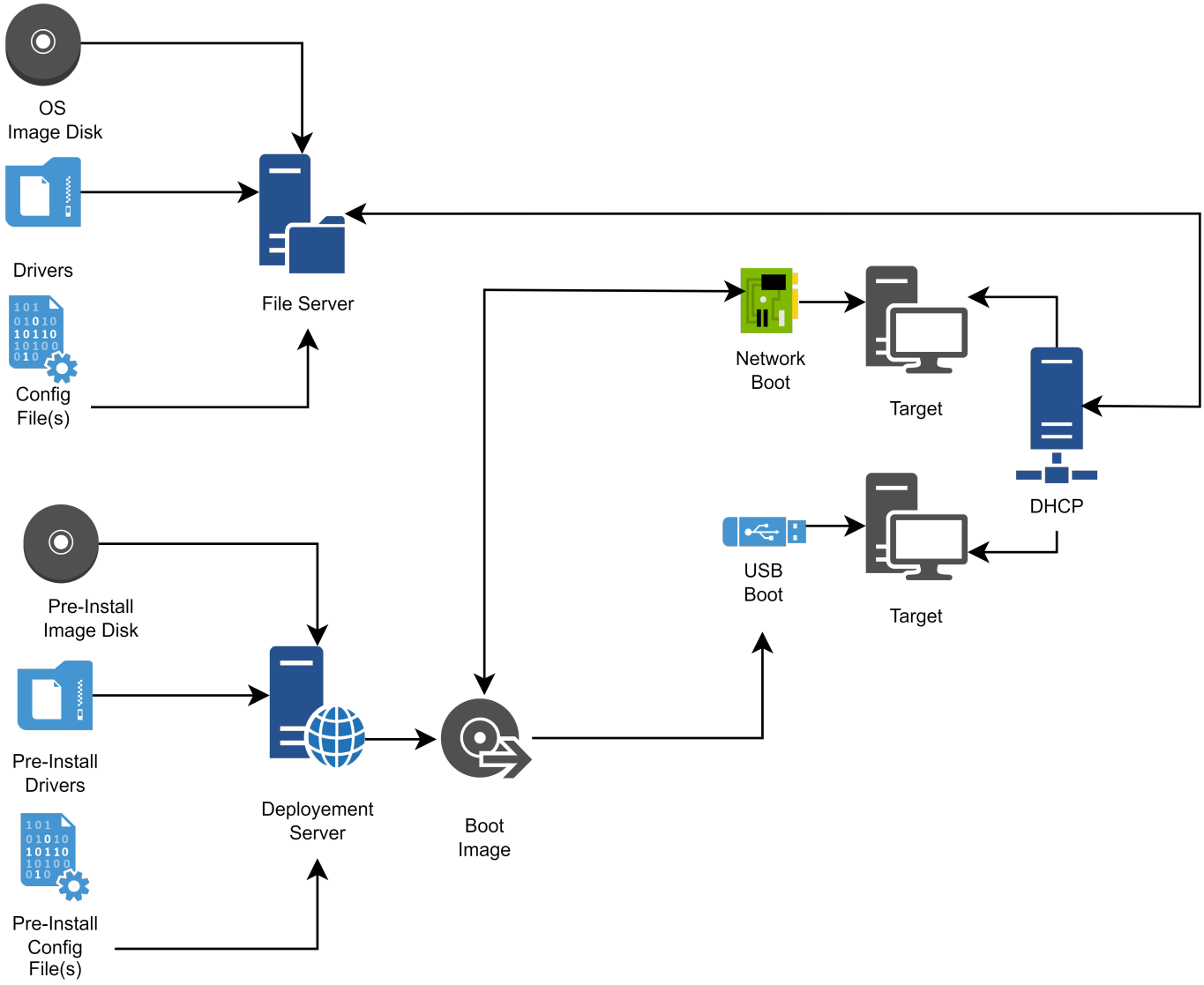


Figure4: Diagramme de flux pour le déploiement d'un OS classique

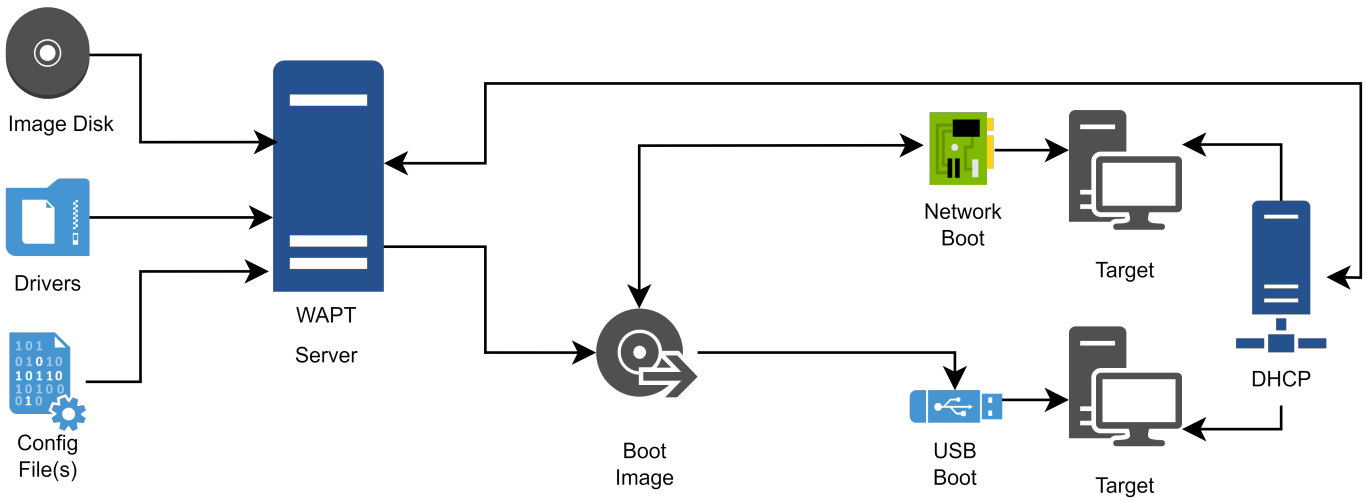


Figure5: Diagramme de flux pour un déploiement WADS

CHAPTER 43

Installation et configuration de TFTP et DHCP pour WADS

43.1 Installation et configuration d'un serveur TFTP

Avertissement: Si vous avez installé un autre serveur tftp sur le serveur WAPT, veuillez d'abord le désinstaller.

Cette documentation est destinée à WAPT 2.2.1 et aux versions ultérieures

Choisissez votre distribution

Linux Debian / Ubuntu / Redhat

- Activer et démarrer le serveur tftpInstaller le serveur TFTP.

```
systemctl enable wapttftpserver
systemctl start wapttftpserver
```

- Vous pouvez tester que le serveur tftp fonctionne correctement en utilisant un client tftp et tester le téléchargement du fichier ipxe.efi. Si vous testez la commande suivante sur une machine basée sur Redhat autre que le waptserver, faites attention au pare-feu sortant local qui bloque les requêtes sortantes du client tftp.

```
cd ~
tftp srvwapt.mydomain.lan
  binary
  get ipxe.efi
  quit
ls -l ipxe.efi
```

Windows

- Lors de l'installation du serveur, cochez la case WADS tftp. Vous pouvez relancer l'installateur si cela n'a pas été fait à ce moment-là. Vous pouvez vérifier que le service est configuré et fonctionne avec la commande

```
sc query wapttftpserver
```

- Si le serveur est installé mais pas démarré, vous pouvez le démarrer avec :

```
net start wapttftpserver
```

43.2 Installation et configuration d'un serveur DHCP

Le démarrage PXE est un processus en deux étapes. D'abord, le chargeur de démarrage UEFI/BIOS téléchargera le binaire iPXE depuis le serveur tftp, puis le binaire iPXE téléchargera le script iPXE et les binaires de démarrage depuis http. C'est pourquoi nous devons avoir une configuration PXE DCHP en deux étapes.

Serveur DHCP

Par exemple :

```
<!-- global options -->
next-server 192.168.1.30;

option ipxe-url code 175 = text;
option client-architecture code 93 = unsigned integer 16;

<!-- subnet mydomain.lan netmask 255.255.255.0 -->

if option client-architecture = 00:00 {
  if exists user-class and option user-class = "iPXE" {
    filename "http://srvwapt.mydomain.lan/api/v3/baseipxe?uefi=false&keymap=fr";
  }
  else{
    filename "undionly.kpxe";
  }

} else {
  if exists user-class and option user-class = "iPXE" {
    option ipxe-url "http://srvwapt.mydomain.lan:80/";
    filename "http://srvwapt.mydomain.lan/api/v3/baseipxe?keymap=fr";
  }
  else{
    filename "ipxe.efi";
  }
}
```

Pour plus d'informations, vous pouvez consulter le site https://ipxe.org/howto/dhcpd

Serveur DNSMASQ

Par exemple :

```
dhcp-match=set:ipxe,175 # iPXE sends a 175 option.
dhcp-boot=tag:!ipxe,undionly.kpxe,IP_WAPTSERVER
dhcp-boot=tag:ipxe,http://srvwapt.mydomain.lan/api/v3/baseipxe?uefi=false
```

Par exemple pour une machine :

```
dhcp-match=set:ipxe,175 # iPXE sends a 175 option.
dhcp-mac=set:waptserver,MAC_ADDRESS_TARGET_COMPUTER
dhcp-boot=tag:!ipxe,undionly.kpxe,waptserver,IP_WAPTSERVER
dhcp-boot=tag:ipxe,http://srvwapt.mydomain.lan/api/v3/baseipxe?uefi=false,waptserver
```

Windows

You can use the following PowerShell command line to configure iPXE booting on your network. Please adapt the *$url_waptserver* and *$waptserver_ipaddress_tftp* depending on your current installation. *keymap* is the keyboard language

```
$waptserver_ipaddress_tftp = "192.168.154.13"
$url_waptserver = "http://srvwapt.mydomain.lan"
$keymap = "fr"

Add-DhcpServerv4Class -Name "legacy_bios" -Type Vendor -Data "PXEClient:Arch:00000"
Add-DhcpServerv4Class -Name "iPXE" -Type User -Data "iPXE"

Set-DhcpServerv4OptionValue -OptionId 66 -Value "$waptserver_ipaddress_tftp"

Add-DhcpServerv4Policy -Name "wapt-ipxe-url-legacy" -Condition AND -UserClass EQ,iPXE -VendorClass␣
↪EQ,legacy_bios*
Set-DhcpServerv4OptionValue -PolicyName "wapt-ipxe-url-legacy" -OptionID 67 -Value "$url_waptserver/
↪api/v3/baseipxe?uefi=false&keymap=$keymap"

Add-DhcpServerv4Policy -Name "wapt-ipxe-url-uefi" -Condition AND -UserClass EQ,iPXE -VendorClass NE,
↪legacy_bios*
Set-DhcpServerv4OptionValue -PolicyName "wapt-ipxe-url-uefi" -OptionID 67 -Value "$url_waptserver/
↪api/v3/baseipxe?keymap=$keymap"

Add-DhcpServerv4Policy -Name "ipxe.efi" -Condition AND -UserClass NE,iPXE -VendorClass NE,legacy_
↪bios*
Set-DhcpServerv4OptionValue -PolicyName "ipxe.efi" -OptionID 67 -Value "ipxe.efi"

Add-DhcpServerv4Policy -Name "undionly.kpxe" -Condition AND -UserClass NE,iPXE -VendorClass EQ,
↪legacy_bios*
Set-DhcpServerv4OptionValue -PolicyName "undionly.kpxe" -OptionID 67 -Value "undionly.kpxe"

For more information, you can refer to https://ipxe.org/howto/msdhcp
```

CHAPTER 44

Déploiement d'un système d'exploitation Windows via WADS

44.1 Processus de déploiement

1. **Utilisation du BIOS/UEFI** :

- l'hôte fait une requête *DHCP* pour obtenir une *IP* et la *configuration PXE* (IP du serveur TFTP et nom du fichier iPXE), ou bien
- l'hôte démarre à partir d'une clé USB qui contient la *configuration PXE*

2. **Utilisation du BIOS/UEFI** :

- l'hôte fait une requête *TFTP* pour obtenir *iPXE* et sa configuration, ou bien
- l'hôte exécute la configuration *iPXE* à partir de la clé USB.

3. Ensuite, en utilisant **iPXE**, l'hôte fait une requête *HTTPS* au serveur WADS pour obtenir le BCD (Boot Configuration Data) et le `fichier WinPE`.
4. Enfin, en utilisant **WinPE**, l'hôte contacte le serveur WADS via *HTTP* pour obtenir le fichier iso du système d'exploitation et ses fichiers de configuration associés.

44.2 Exigences avant de commencer

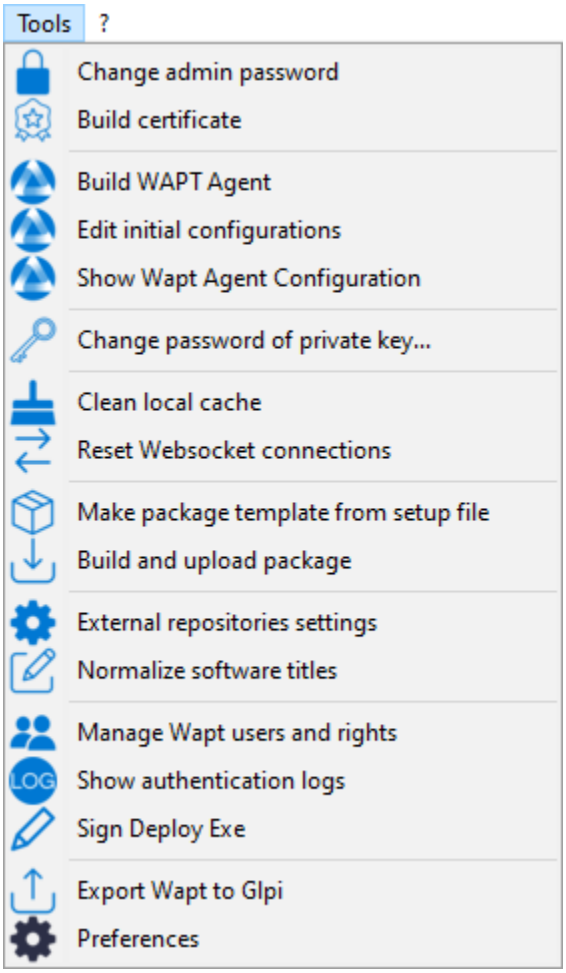
1. Pour utiliser WADS sur votre console WAPT, vous devez installer un packaging spécifique sur votre station de gestion.

Deux packages sont disponibles, un seul est nécessaire. Choisissez en fonction de vos besoins :

- Ce packaging intègre les **exigences minimales**pour créer un fichier WinPE.
- Ce packaging installe **Windows ADK**, tous les outils pour créer et modifier WinPE.

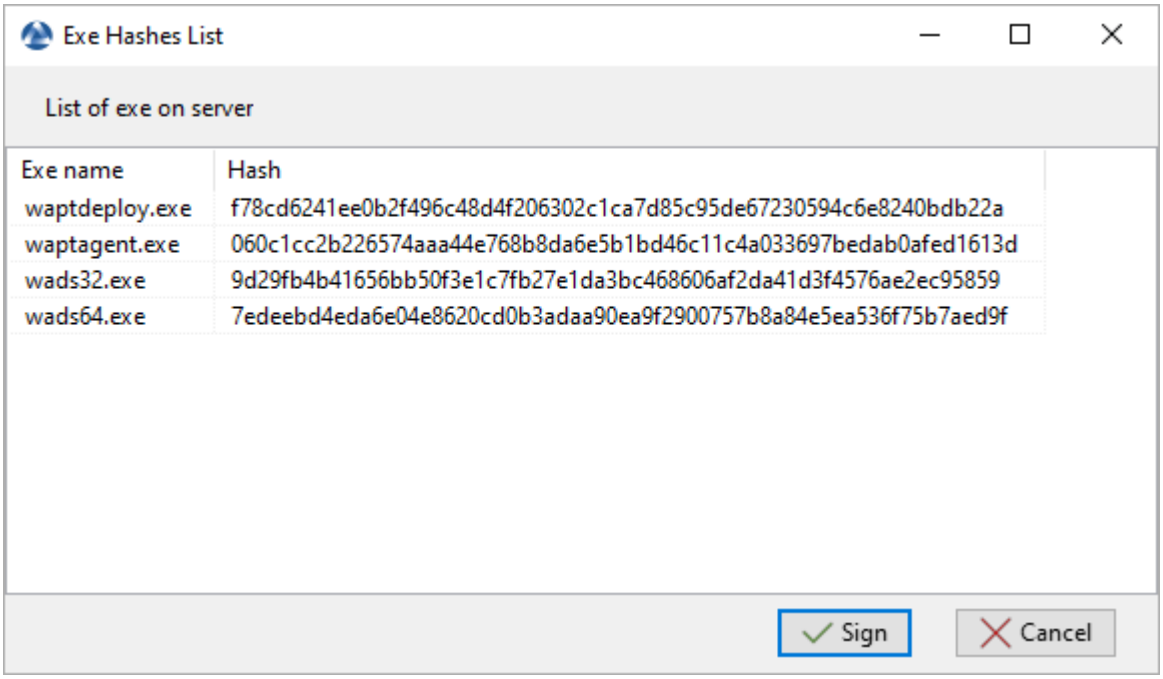
2. A partir de 2023-04-17, le compte utilisateur utilisant la console WADS **Doit**avoir des droits d'administrateur local dans les *Listes de contrôle d'accès WAPT*.
3. Signer WADS avec votre certificat :

- Allez dans le menu *Outils* → *Signer Deploy Exe*.



- Cliquez sur le bouton *Sign* :

4. Allez dans l'onglet *OS Deploy* :



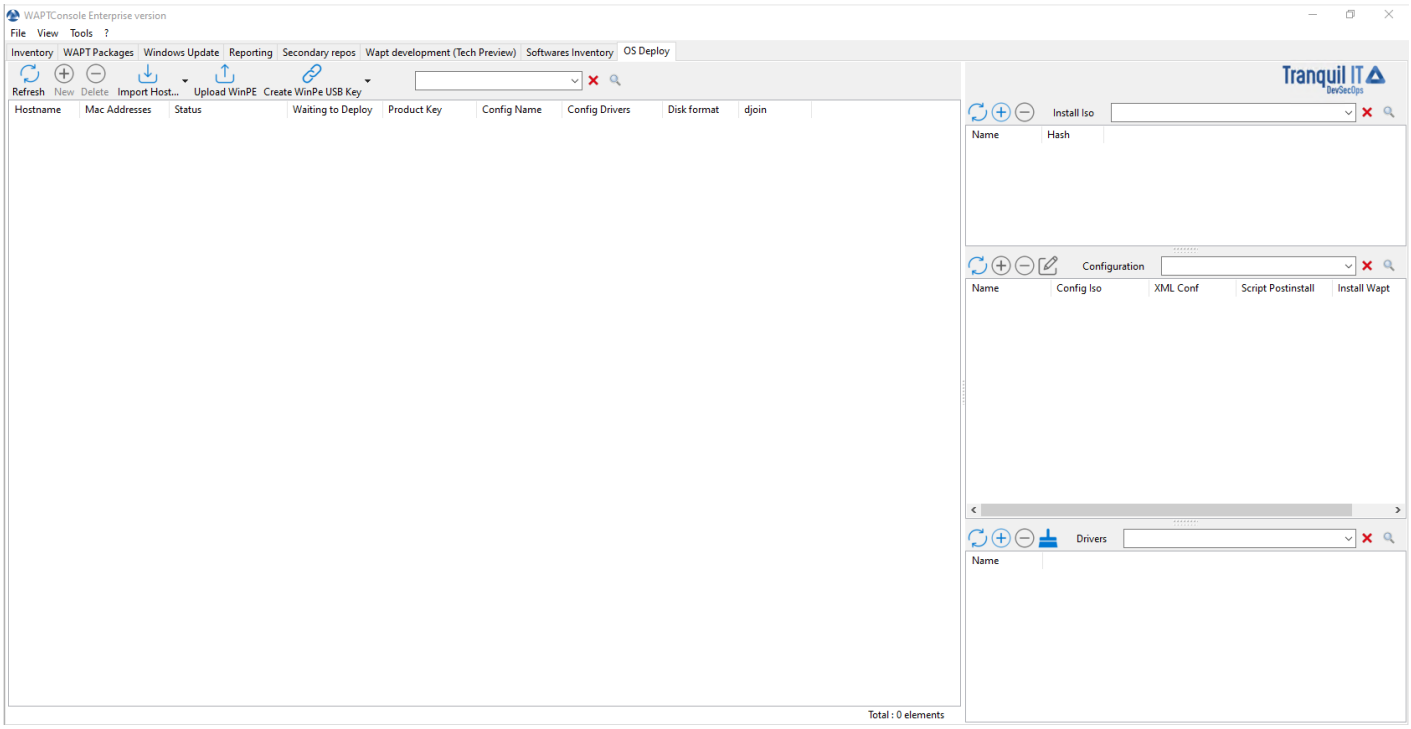


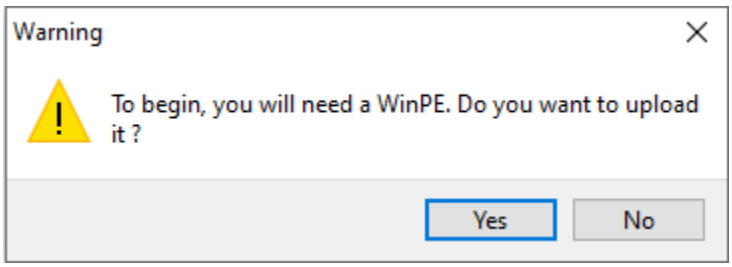
Figure1: Fenêtre principale de la console WADS

44.3 Ajout des fichiers WinPE

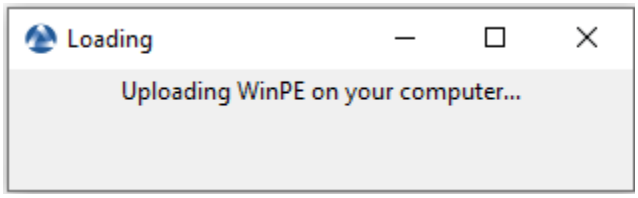
WinPE est un système d'exploitation minimal utilisé pour installer, déployer et réparer Windows.

Sur WADS, WinPE est utilisé pour amorcer le déploiement de Windows.

- Si aucun fichier WinPE n'existe, cette fenêtre contextuelle apparaîtra.



- Cliquez ensuite sur *Upload WinPE*.
- Choisissez la disposition du clavier. **Cette étape est importante car vous allez taper le nom d'hôte dans WinPE en utilisant la disposition de clavier choisie avec cette étape**.
- Sélectionner le certificat avec lequel vous souhaitez signer les fichiers de la clé USB
- Attendez que le fichier `WinPE` soit téléchargé sur l'ordinateur d'administration WAPT :



- Attendez que le fichier `WinPE` soit téléchargé sur le serveur WADS :

Si le fichier `WinPE` a été téléchargé avec succès sur le serveur WADS.

44.4 Ajout du système d'exploitation ISO

L'étape suivante consiste à ajouter le fichier `.iso` du système d'exploitation à utiliser pour déployer Windows.

- Utilisez la dernière version officielle de Windows de Microsoft comme fichier `.iso`.
- Dans la section *Installation ISO* de la console principale de WADS, cliquez sur le bouton + pour télécharger le fichier `.iso` sélectionné.
- Sélectionnez le fichier `.iso` et donnez-lui un nom.
- Lors du téléchargement, le fichier `.iso` est signé avec le certificat sélectionné :
- Une fois l'étape de signature terminée avec succès, le fichier `.iso` est téléchargé sur le serveur WADS :
- Une fois l'étape de téléchargement terminée avec succès, le fichier `.iso` apparaît dans la section *Installation iso* de la console principale de WADS :

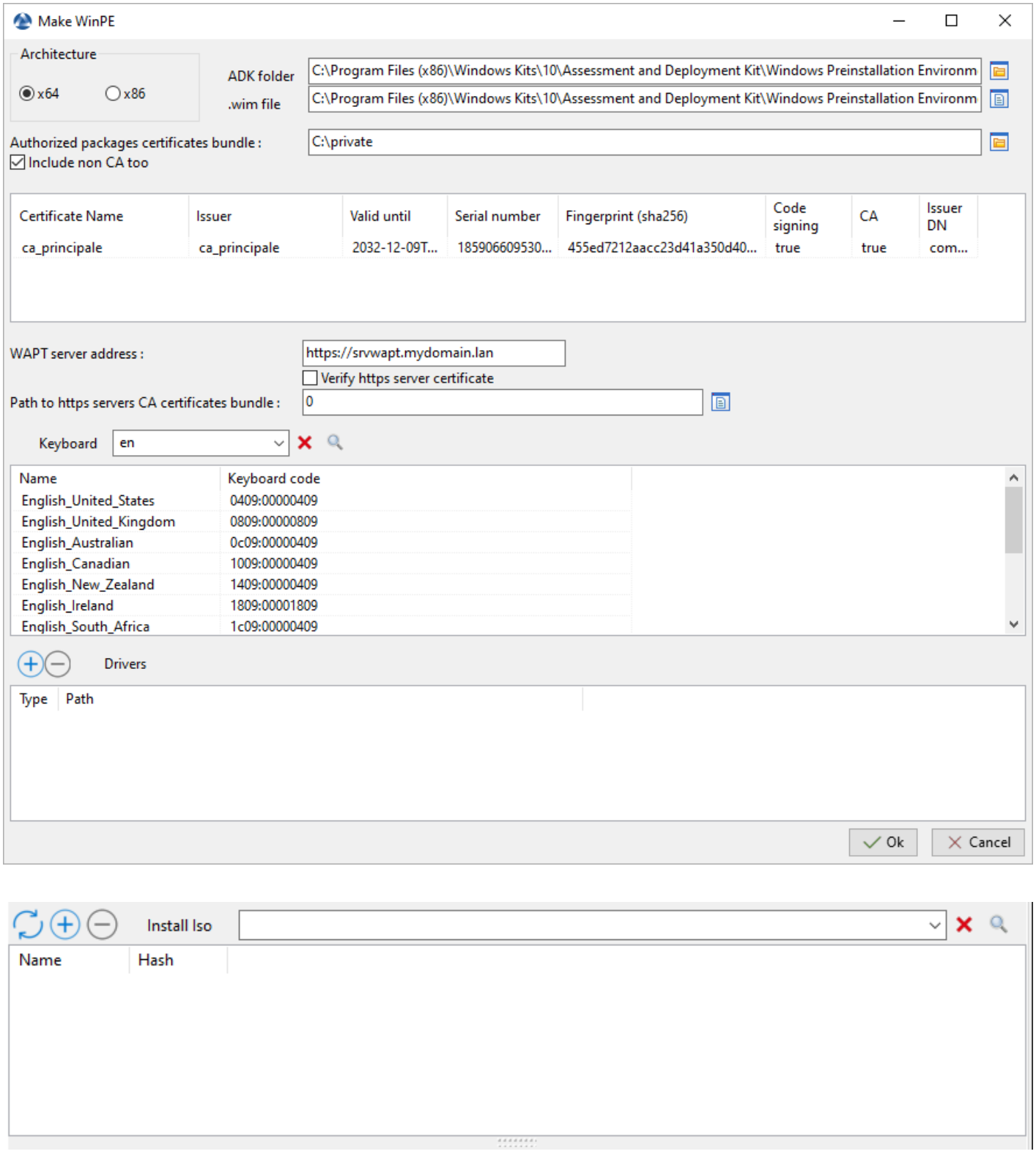


Figure2: Section ISO de la console WADS

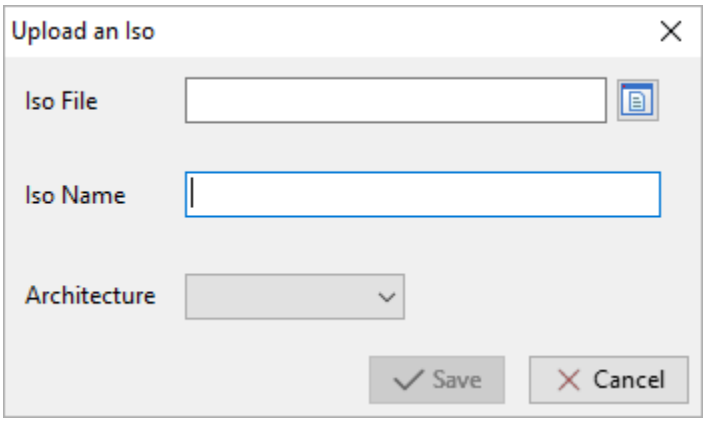


Figure3: Boîte de dialogue permettant de sélectionner le fichier ISO à télécharger sur le serveur WADS

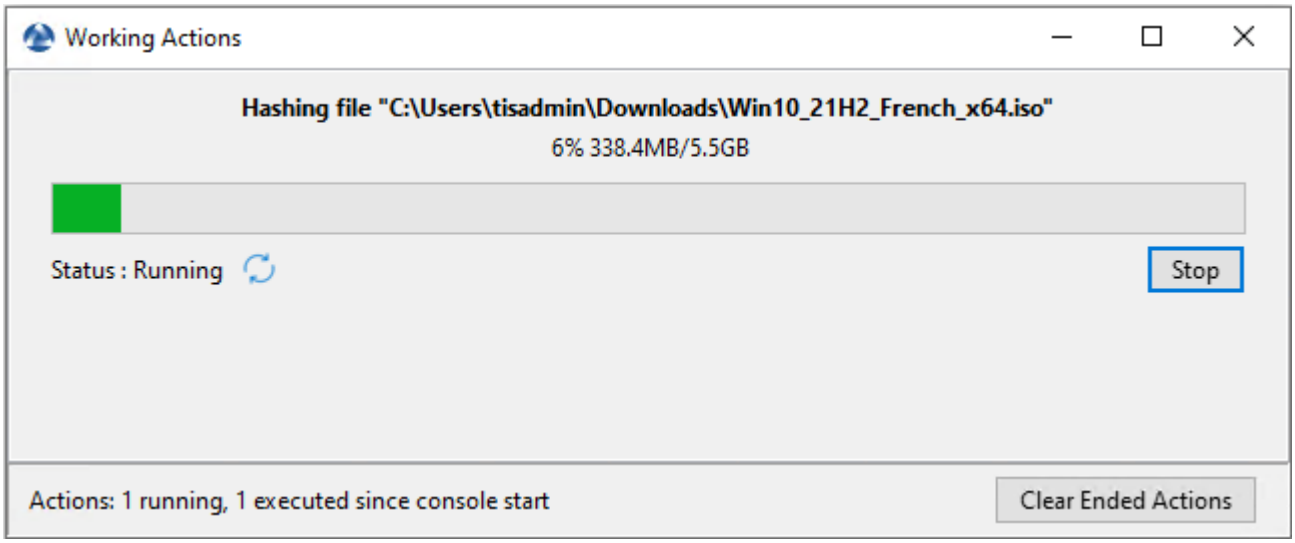


Figure4: Boîte de dialogue informant de la progression de la signature du fichier ISO dans la console WADS

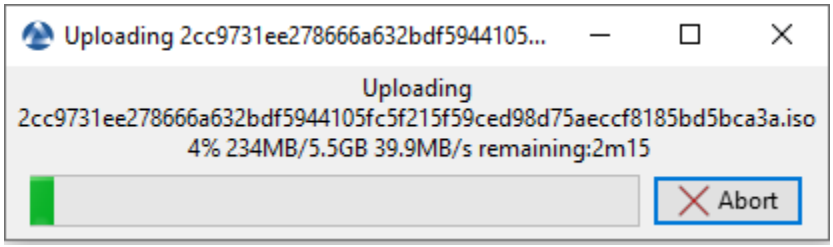


Figure5: Boîte de dialogue informant de la progression du téléchargement du fichier ISO dans la console WADS

Name	Hash
Windows	2cc9731ee278666a632bdf5944105fc5f215f59ced98d75aeccf8185bd5bca3a

Indication: Il est possible de télécharger plusieurs versions `.iso` de Windows pour différents cas d'utilisation.

44.5 Ajouter le fichier de réponse

L'étape suivante consiste à ajouter le fichier de réponse qui sera utilisé pour configurer le déploiement du système d'exploitation Windows.

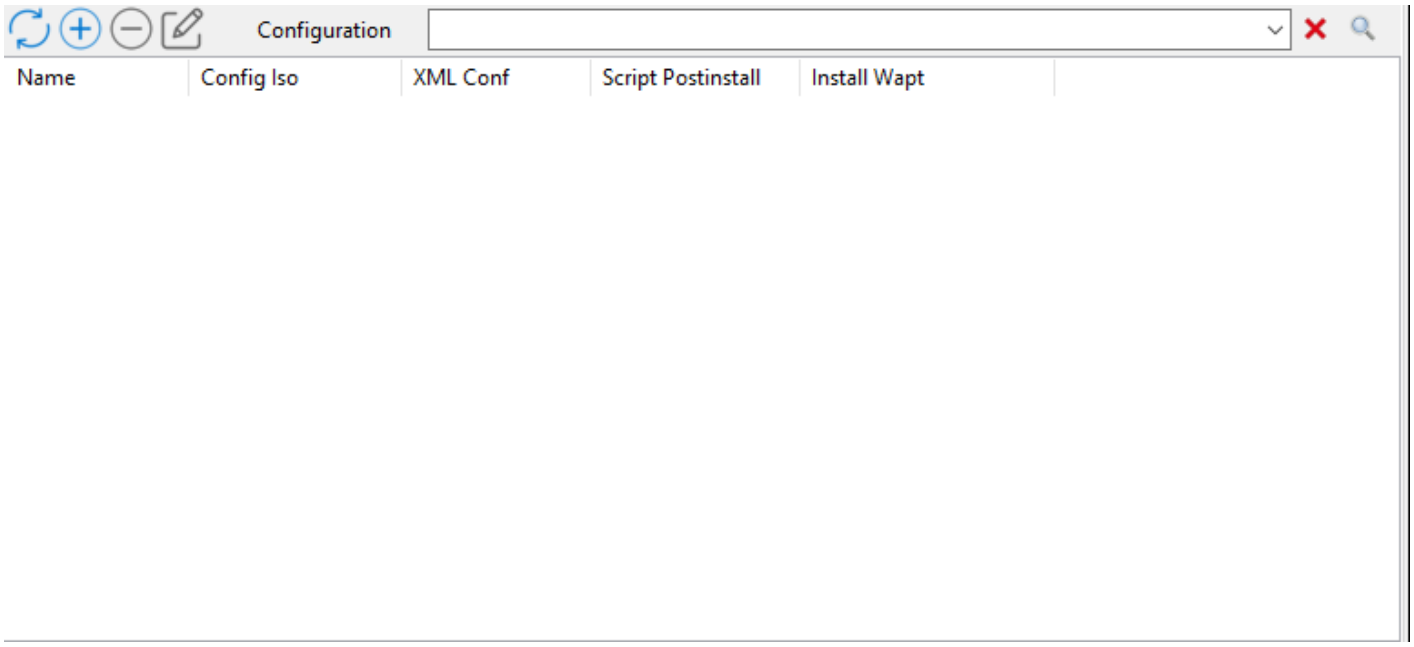


Figure6: Section du fichier de réponses de la console WADS

- Dans la section *Configuration*, cliquez sur le bouton + pour configurer le fichier de réponse.

Table1: Options pour le fichier de réponse dans la Console WADS

Options	Description
Nom de la configuration	Définit le nom du fichier de réponse XML.
Nom de l'ISO	Définit le fichier `.iso` à associer au fichier de réponse XML.
Pour Windows	Définit si vous installez un système d'exploitation Windows ou Linux si cette option n'est pas cochée.
Installer Wapt	Définit s'il faut installer l'agent **WAPT** après l'installation du système d'exploitation.
Fichier de configuration	Définit les fichiers de réponses XML modèles à utiliser pour Windows ou le fichier de configuration pour Linux.
Script de post installation	Définit un script de post-installation `.bat` à exécuter après l'installation du système d'exploitation.

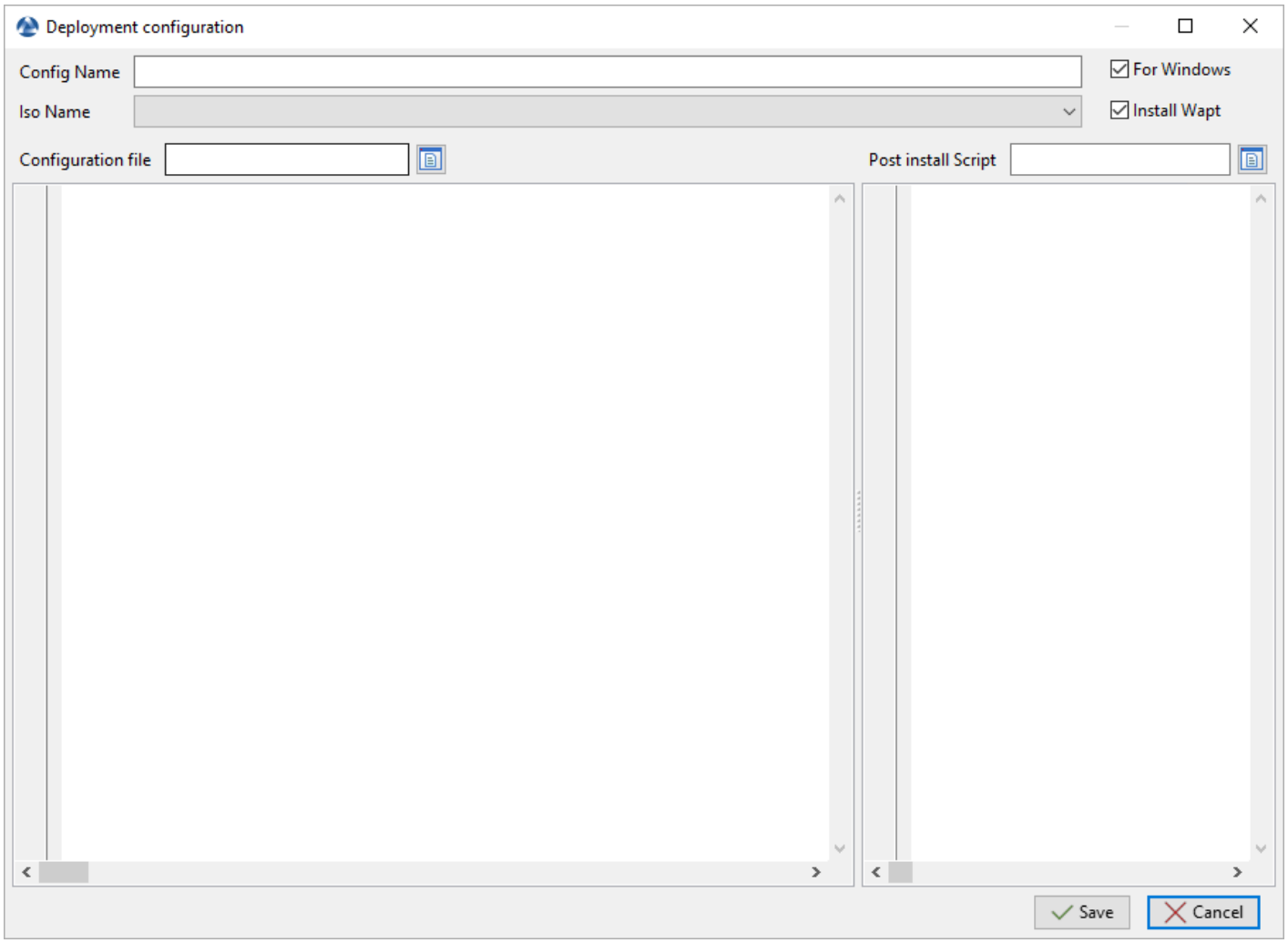


Figure7: Fenêtre de création du fichier de configuration de réponse dans la Console WADS

- Insérez dans le champ *Nom de la configuration* le nom du fichier de réponse.
- Sélectionnez avec la liste déroulante *Nom de l'ISO* le fichier ISO à associer à la configuration de déploiement.
- Cochez ou décochez la case *Installer WAPT* pour installer l'agent WAPT par défaut.
- Cocher ou décocher la case *Pour Windows* pour installer le système d'exploitation Windows.
- Sélectionner le modèle de fichier de réponse à associer à la configuration du déploiement avec le champ *Fichier de configuration* s'il s'agit du système d'exploitation Windows ; sinon, sélectionner le fichier de configuration pour Linux.
- Si nécessaire, définir le script de post-installation dans *Post install Script*, par exemple :

```
"C:\Program Files (x86)\wapt\wapt-get.exe" install tis-firefox-esr
```

- Cliquer sur le bouton *Créer* pour créer le fichier de réponse.
- Lorsque cela est fait, la configuration apparaît dans la section *Configuration*.

Name	Config Iso	XML Conf	Script Postinstall	Install Wapt
Windows 10	Windows	<!--********** ...		True

Figure8: Fichier de réponses ajouté au serveur WADS dans la console WADS

Indication: Il est possible de créer plusieurs configurations de fichiers de réponses pour différentes versions de Windows / Linux et pour différents cas d'utilisation.

44.5.1 Joindre l'hôte à un domaine Active Directory

Vous pouvez utiliser votre propre fichier de réponse avec WADS mais par défaut, WADS intègre *2* types de fichiers de réponse pour Windows :

- **Offline**pour joindre un ordinateur avec la méthode DirectAccess Offline Domain Join (Djoin)
- **Online**pour joindre un ordinateur sur l'AD

Méthode en ligne

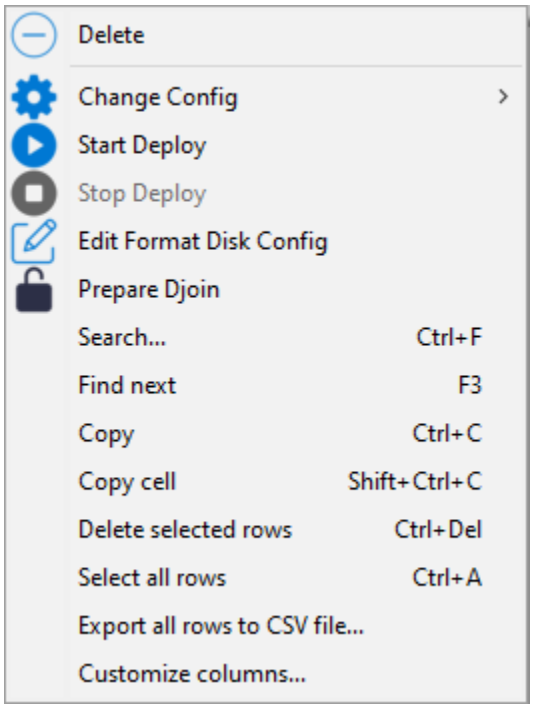
Mettre à jour cette partie avec votre **compte de service d'adhésion**, vous pouvez spécifier une UO si vous le souhaitez. Sinon, supprimez simplement la ligne *MachineObjectOU*.

```
<Identification>
  <Credentials>
    <Domain>mydomain.lan</Domain>
    <Password>password</Password>
    <Username>wadsjoin</Username>
  </Credentials>
  <JoinDomain>mydomain.lan</JoinDomain>
  <MachineObjectOU>OU=MyOu,OU=MyParentOu,DC=MyDomain,DC=lan</MachineObjectOU>
</Identification>
```

Méthode hors ligne

La méthode hors ligne utilise la méthode Djoin.

- Cliquez avec le bouton droit de la souris sur l'hôte pour ouvrir la liste des menus.



- Cliquez sur *Préparation Djoin*.
- Sélectionnez la OU (Organizational Unit)`à laquelle rattacher l'hôte (ou définissez-la manuellement) et cliquez sur :guilabel:`Save.

Vous pouvez cocher *Changer d'utilisateur* si votre utilisateur actuel ne peut ou ne doit pas joindre un ordinateur au domaine. Si cette case est cochée, vous devez indiquer manuellement le **Domaine**, le **Host OU**, le **User** (juste le sAMAccountName, pas l'UPN ni le DOMAINuser) et le **password**.

Vous pouvez cocher la case *Écraser la machine* afin de joindre un nouvel ordinateur.

- Le fichier `Djoin` est prêt à être utilisé pour joindre l'hôte en tant que membre du domaine Active Directory.

44.6 Ajout de conducteurs

L'étape suivante consiste à ajouter les paquets de pilotes qui seront utilisés lors du déploiement du système d'exploitation Windows.

- Dans la section *Drivers*, cliquez sur le bouton + pour ajouter un pack de pilotes au serveur WADS.

Cette fenêtre vous permet de télécharger les paquets de pilotes à associer au déploiement Windows.

Table2: Options pour les paquets de pilotes dans la console WADS

Options	Description
Choisir le chemin	Définit le chemin d'accès au dossier contenant les paquets de pilotes.
Nom	Définit le nom du paquet de pilotes.

- Cliquez sur le bouton *Save*, le téléchargement des bundles de pilotes commence.
- Une fois téléchargé, le pack de pilotes apparaît dans la section *Drivers* de la console WADS.

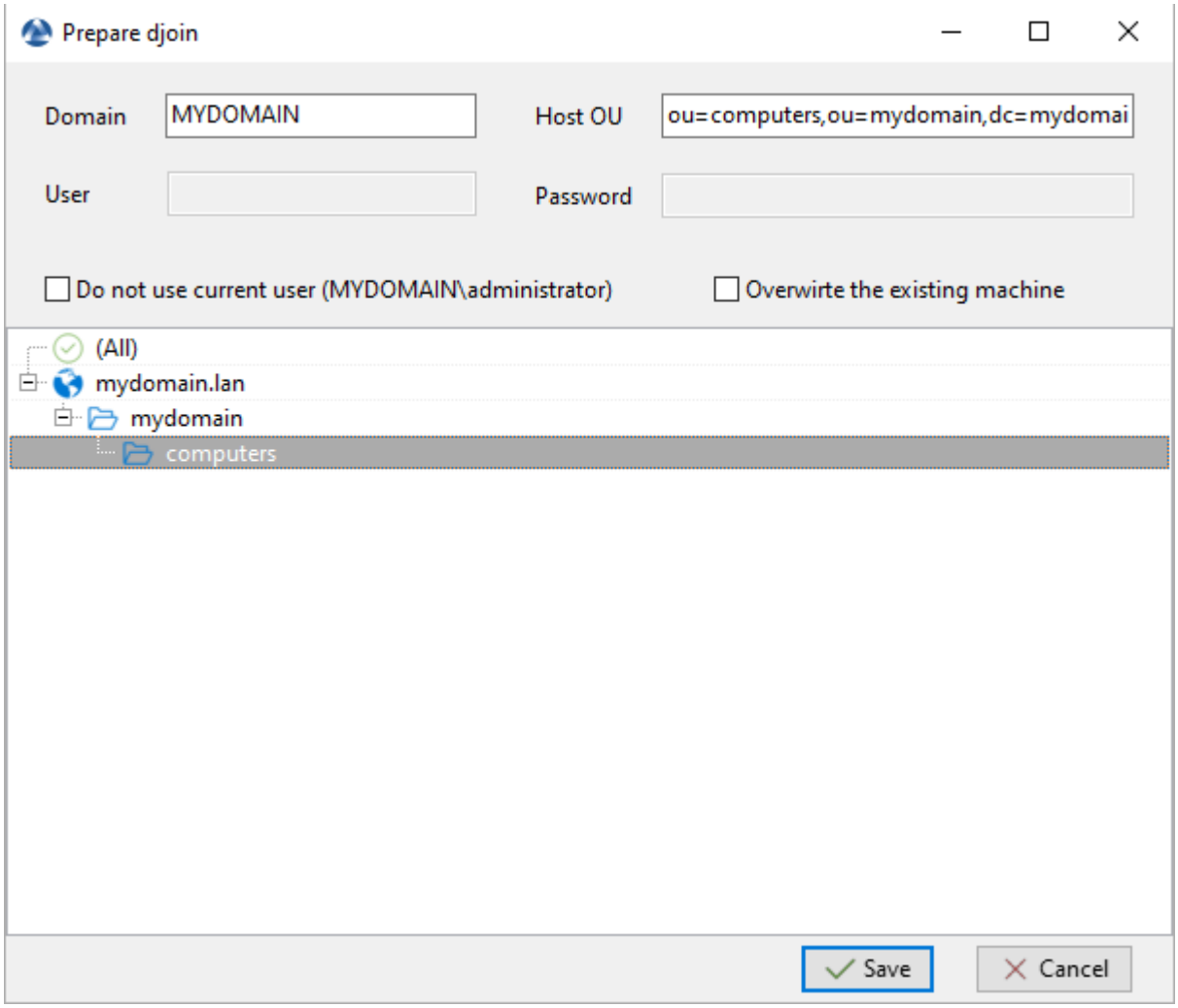


Figure9: Sélection de l'unité organisationnelle à laquelle rattacher automatiquement l'hôte réimagé

Figure10: Section des pilotes de la console WADS

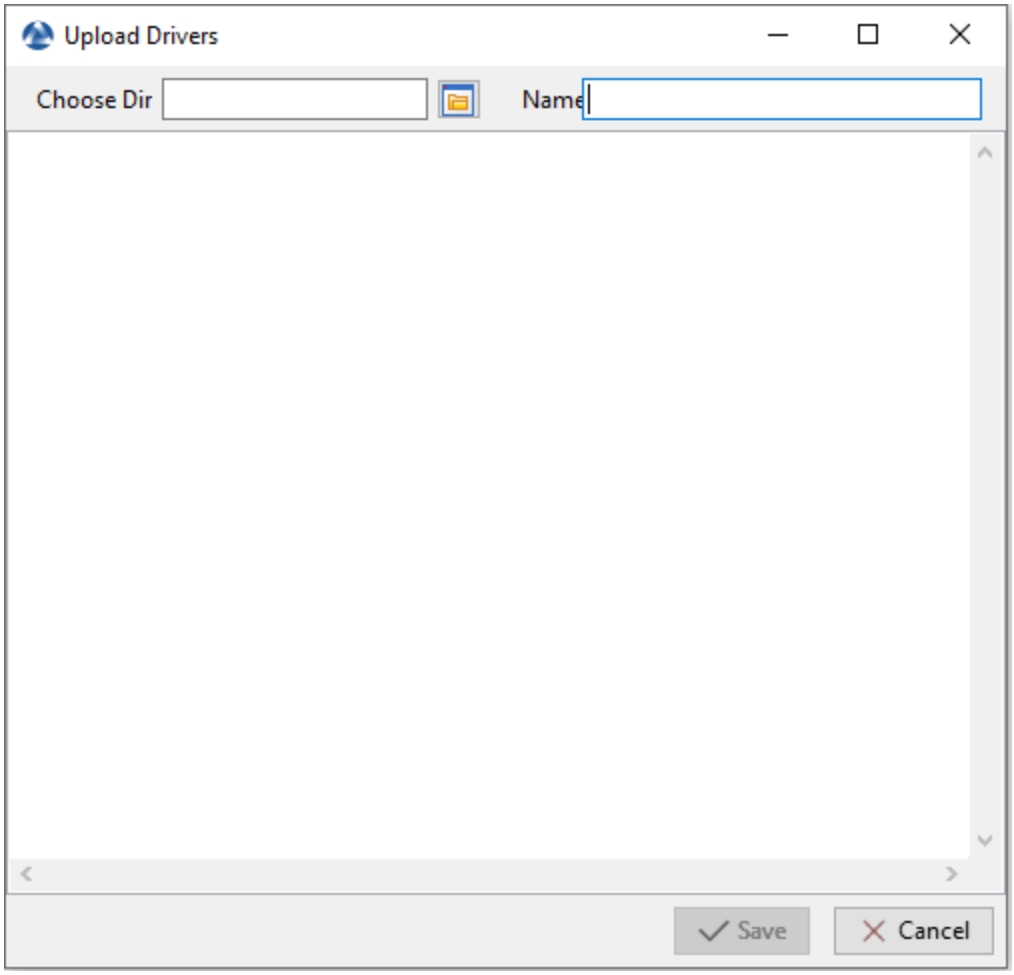


Figure11: Fenêtre pour la création des paquets de pilotes dans la console WADS

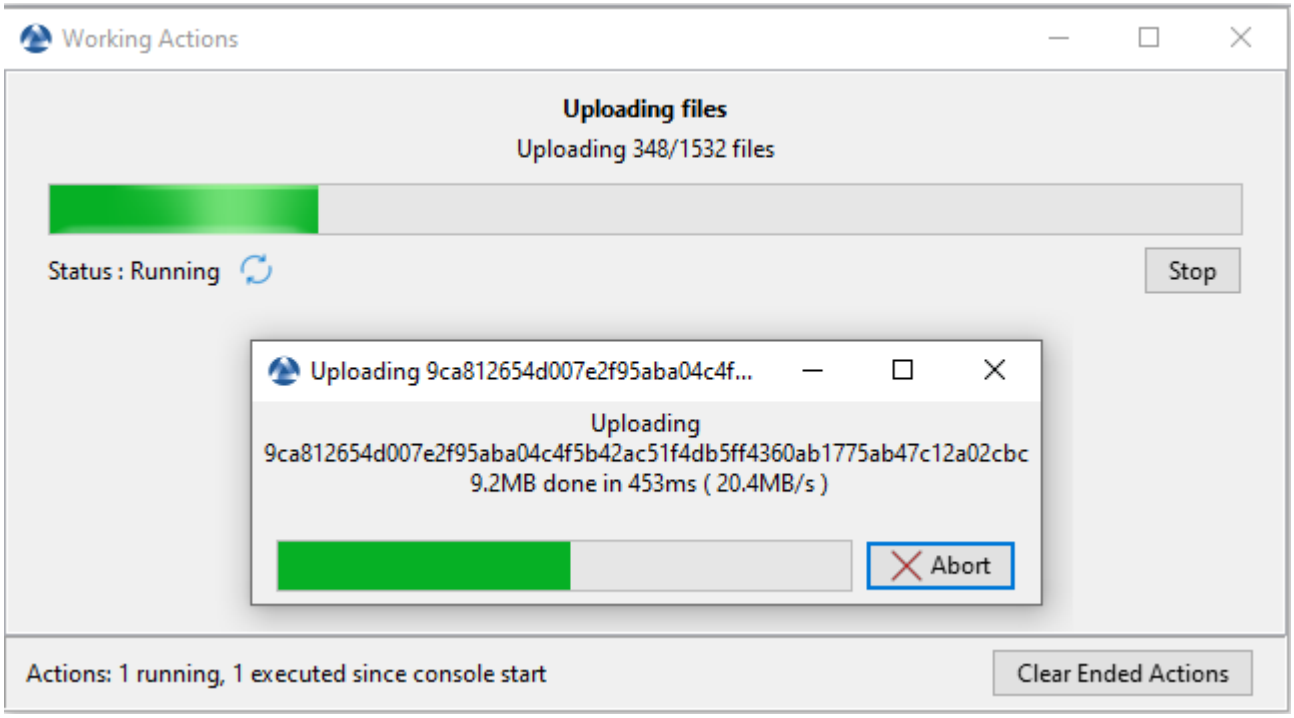


Figure12: Boîte de dialogue informant de la progression du téléchargement des paquets de pilotes dans la console WAPT

Figure13: Le fichier `WinPE` a été téléchargé avec succès sur le serveur WADS

Il est possible de créer plusieurs packs de pilotes pour différentes versions de Windows et pour différents cas d'utilisation.

Il est possible d'utiliser les fichiers `.cab` de OEM (Original Equipement Manufacturers).

Il est également possible d'exporter les pilotes d'un hôte existant qui fonctionne bien en utilisant une commande **`Powershell`**.

```
Export-WindowsDriver -Online -Destination D:\Drivers
```

44.7 Démarrer l'hôte pour réimager avec WADS

WADS permet **2**méthodes de démarrage de l'hôte pour réimager :

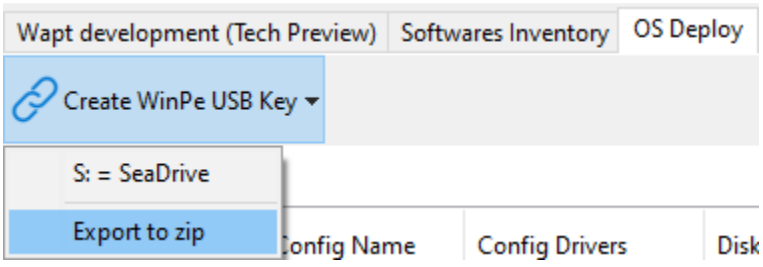
- *En local avec une clé USB*.
- *Via LAN avec un serveur TFTP*

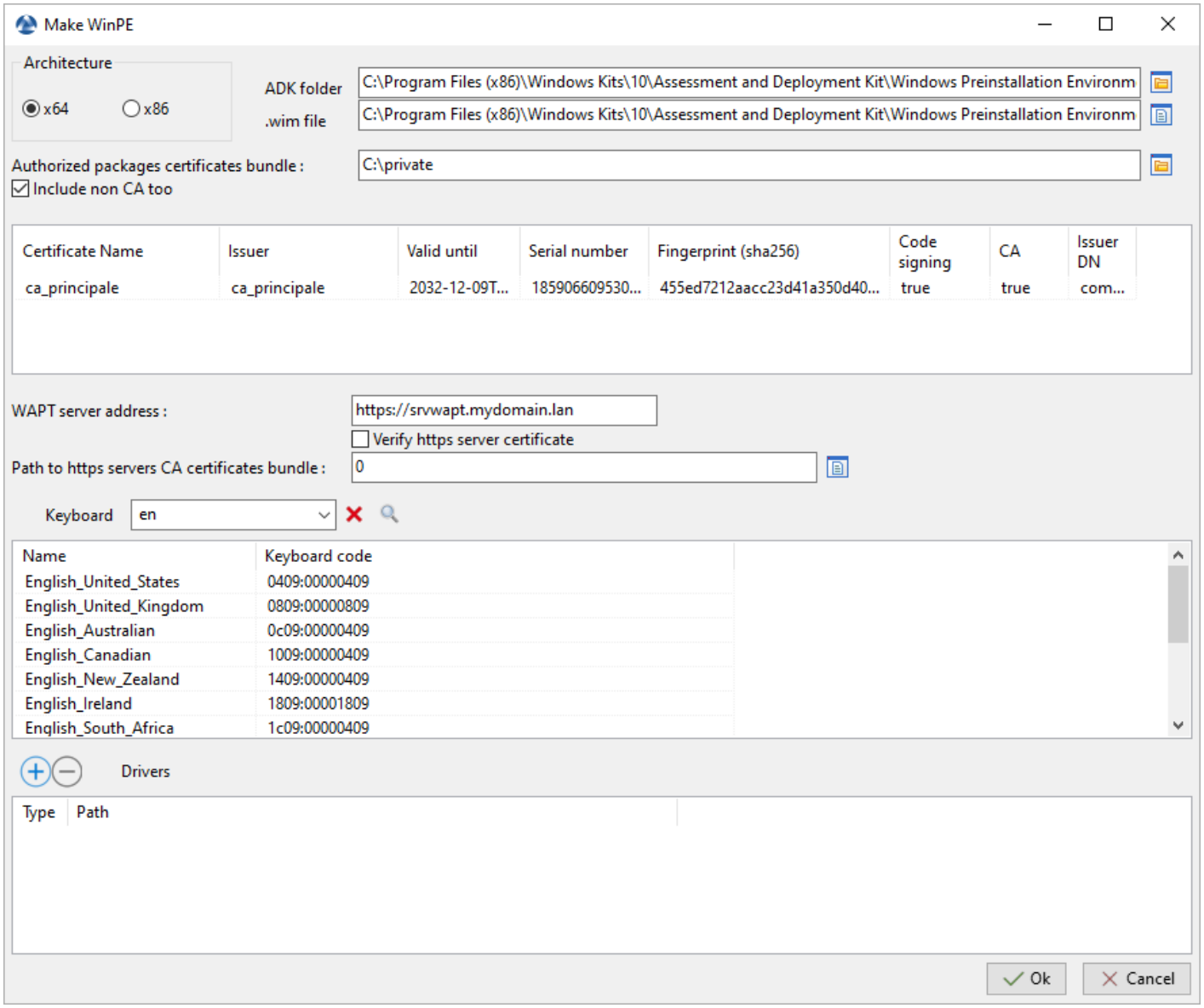
44.7.1 Démarrage de l'hôte avec une clé USB

Note: La clé USB utilisée **Doit** être formatée en FAT32 et vide.

- Insérez la clé USB dans le poste d'administration de WAPT et cliquez sur le bouton *Créer une clé USB WinPE* pour lancer le processus.
- Choisissez la disposition du clavier. **Cette étape est importante car vous allez taper le nom d'hôte dans WinPE en utilisant la disposition de clavier choisie avec cette étape**.
- Sélectionner le certificat avec lequel vous souhaitez signer les fichiers de la clé USB
- Cliquez sur le *Transférer WinPE* pour formater la clé USB et copier le fichier WinPE.
- Démarrez le menu de démarrage de l'ordinateur en utilisant l'option de la clé USB et allez à l'étape *exécuter le déploiement*.

Note: Vous pouvez *Exporter vers un fichier zip* lorsque vous créez une clé USB WinPE si vous ne pouvez pas utiliser une clé USB pour ensuite graver le contenu de la clé USB sur un CD / DVD à la place.




Make WinPE
Architecture
x64
x86
ADK folder
C:\Program Files (x86)\Windows Kits\10\Assessment and Deployment Kit\Windows Preinstallation Environm
.wim file
C:\Program Files (x86)\Windows Kits\10\Assessment and Deployment Kit\Windows Preinstallation Environm
Authorized packages certificates bundle :
C:\private
Include non CA too
Certificate Name
Issuer
Valid until
Serial number
Fingerprint (sha256)
Code signing
CA
Issuer DN
ca_principale
ca_principale
2032-12-09T...
185906609530...
455ed7212aacc23d41a350d40...
true
true
com...
WAPT server address :
https://srvwapt.mydomain.lan
Verify https server certificate
Path to https servers CA certificates bundle :
0
Keyboard
en
Name
Keyboard code
English_United_States
0409:00000409
English_United_Kingdom
0809:00000809
English_Australian
0c09:00000409
English_Canadian
1009:00000409
English_New_Zealand
1409:00000409
English_Ireland
1809:00001809
English_South_Africa
1c09:00000409
Drivers
Type
Path
Ok
Cancel

44.7.2 Démarrage de l'hôte avec le réseau

Le démarrage à partir du LAN nécessite :

- Un serveur *TFTP fonctionnant correctement* ;
- Un serveur *DHCP* qui fonctionne correctement ;
- Avoir le port 69 ouvert sur le serveur WAPT pour le trafic entrant, et avoir tftp conntrack activé sur les pare-feu intermédiaires si vous avez des pare-feu entre le serveur et l'ordinateur client.
- Démarrez le menu de démarrage de l'ordinateur en utilisant l'option LAN et allez à l'étape *exécuter le déploiement*.

44.8 Déploiement de l'image Windows

Il y a **3**choix lors du démarrage avec iPXE :

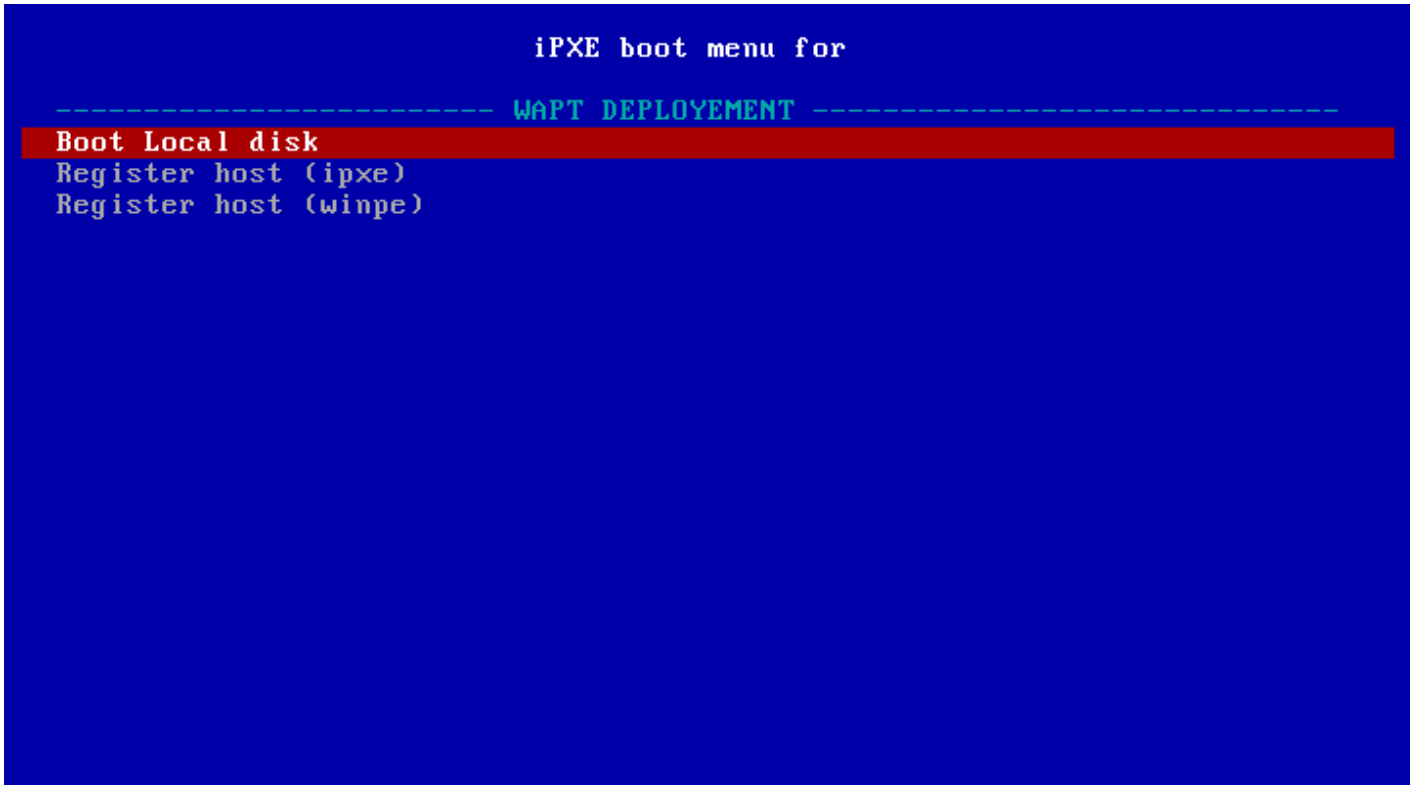


Figure14: fenêtre du menu de démarrage iPXE

- *Disque local de démarrage* pour démarrer normalement à partir du stockage local ;
- *Enregistrer l'hôte (ipxe)* pour enregistrer l'hôte avec le serveur WADS en utilisant la *méthode iPXE* ;
- *Enregistrer l'hôte (winpe)* pour enregistrer l'hôte auprès du serveur WADS en utilisant la méthode *WinPE*.

démarrage iPXE

- Si vous choisissez *Enregistrer un hôte (ipxe)*, définissez un nom d'hôte :

Figure15: Fenêtre de terminal texte demandant un nom d'hôte lors de l'enregistrement par la méthode iPXE

Avertissement: Le clavier est de type qwerty

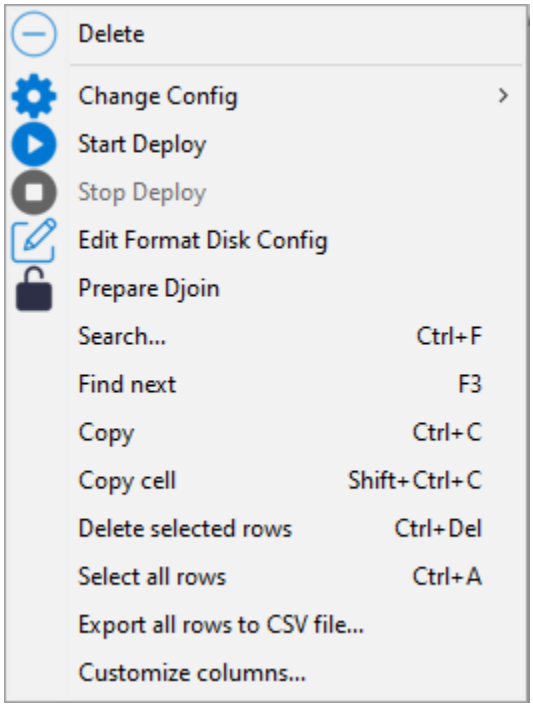
- Rafraîchissez la console WADS avec `F5`, l'hôte apparaît dans l'onglet *OS Deploy*.

Hostname	Mac Addresses	Status	Waiting to Deploy
documentation	08:00:27:19:0D:4B		False

Figure16: Hôte en attente de déploiement

À ce moment, l'état *Waiting to Deploy* de l'hôte est `False`.

- Faites un clic droit sur l'hôte pour ouvrir la liste des menus.



- Allez dans *Change Config* et sélectionnez *un fichier de réponse XML*.
- Cliquez sur *Start Deploy*, le statut *Waiting to Deploy* de l'hôte passe à `True`.

Hostname	Mac Addresses	Status	Waiting to Deploy
documentation	08:00:27:19:0D:4B		True

- Redémarrez l'hôte avec la même option de démarrage que précédemment (USB ou LAN), Windows commencera à s'installer.
- Lorsque l'installation est terminée, l'onglet *OS Deploy*, l'état passe à `Done`.

WinPE

- Si vous choisissez *Enregistrer un hôte (winpe)*, définissez un nom d'hôte :

Le clavier est dans la même disposition que celle définie lors de l'étape *WinPE* de cette documentation.

- Rafraîchissez la console WADS avec `F5`, l'hôte apparaît dans l'onglet *OS Deploy*.

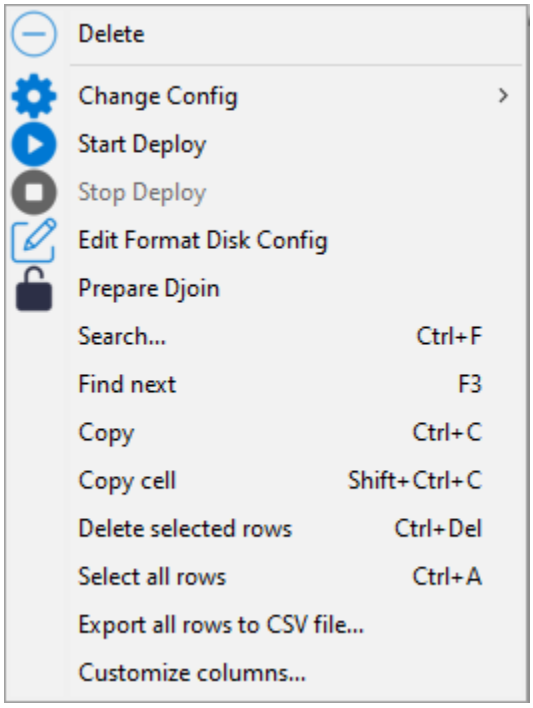
Figure17: Fenêtre de terminal texte demandant un nom d'hôte lors de l'enregistrement à l'aide de la méthode WinPE

Hostname	Mac Addresses	Status	Waiting to Deploy
documentation	08:00:27:19:0D:4B		False

Figure18: Hôte en attente de déploiement

À ce moment, l'état *Waiting to Deploy* de l'hôte est `False`.

- Faites un clic droit sur l'hôte pour ouvrir la liste des menus.



- Allez dans *Change Config* et sélectionnez *un fichier de réponse XML*.
- Cliquez sur *Start Deploy*, le statut *Waiting to Deploy* de l'hôte passe à `True`.

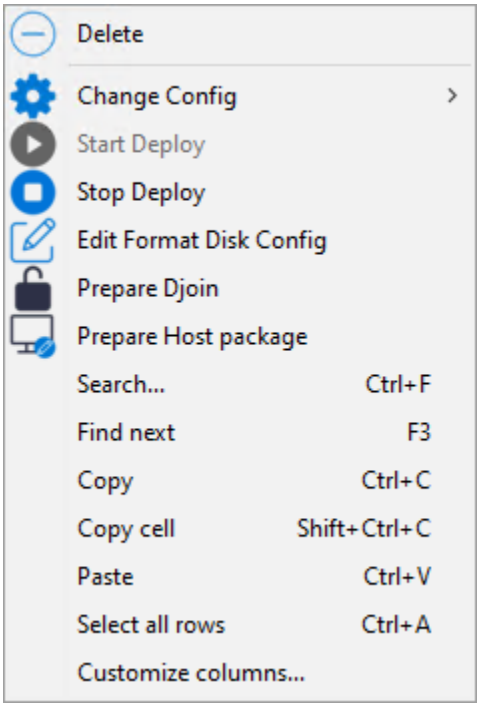
Hostname	Mac Addresses	Status	Waiting to Deploy
documentation	08:00:27:19:0D:4B		True

- Redémarrez l'hôte avec la même option de démarrage que précédemment (USB ou LAN), Windows commencera à s'installer.
- Lorsque l'installation est terminée, l'onglet *OS Deploy*, l'état passe à `Done`.

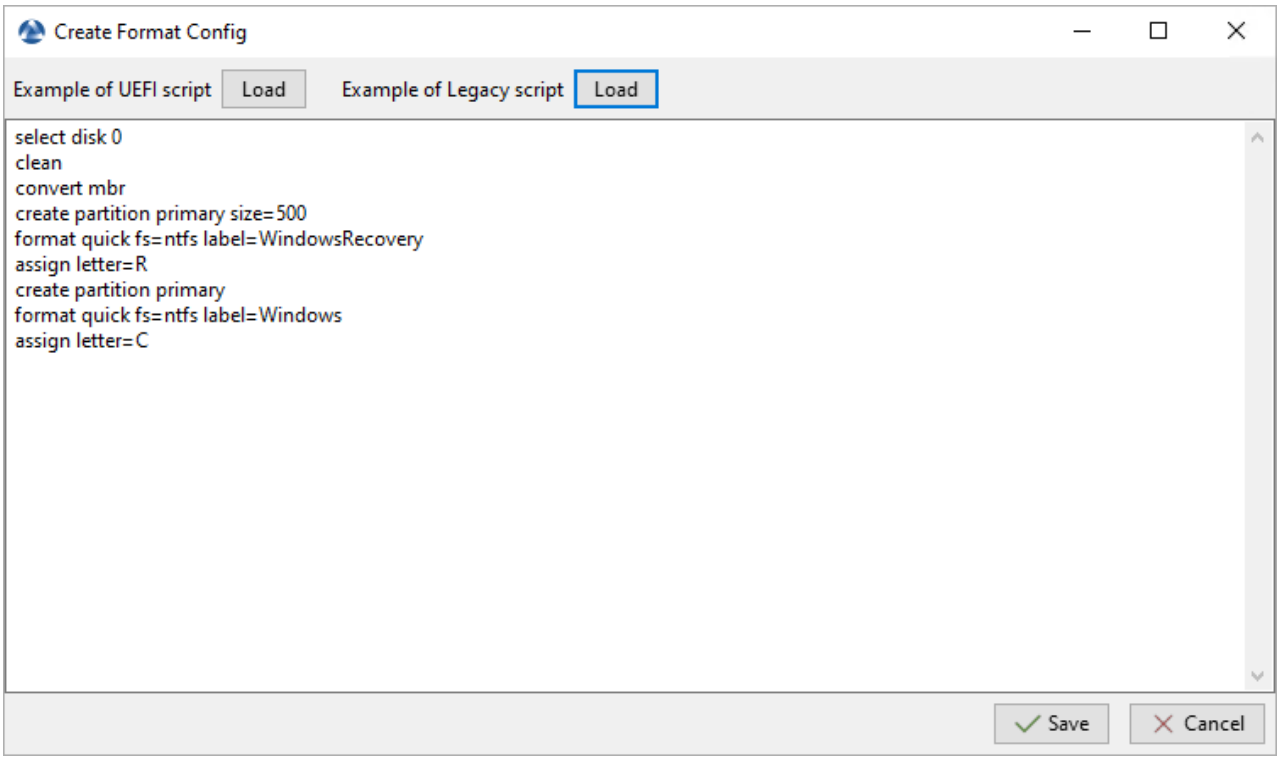
44.9 Formater le disque de la machine

Lorsque votre machine est prête à être redéployée, si nécessaire, vous pouvez formater son disque en utilisant la méthode UEFI ou Legacy.

Pour ce faire, cliquer avec le bouton droit de la souris sur la machine puis *Configurer le formatage du disque*.



Vous pouvez ensuite choisir le script UEFI ou Legacy et personnaliser la configuration du formattage du disque. Voici un exemple avec le script Legacy :


Create Format Config
Example of UEFI script
Load
Example of Legacy script
Load
select disk 0
clean
convert mbr
create partition primary size=500
format quick fs=ntfs label=WindowsRecovery
assign letter=R
create partition primary
format quick fs=ntfs label=Windows
assign letter=C
Save
Cancel

CHAPTER 45

Getting start creating WAPT packages

45.1 Créer son environnement de développement de paquets WAPT

45.1.1 Pré-requis

Attention:

- Il est **impératif** d'être en possession d'un compte *Administrateur Local* de la machine pour cette opération.
- Nous vous conseillons de créer / éditer vos paquets dans un environnement maîtrisé, sain et *jetable*.
- L'utilisation d'une machine virtuelle autonome (type Virtualbox ou équivalent) est vivement recommandée.
- Importer le paquet *tis-pyscripter* dans votre dépôt local et l'installer sur votre machine de développement.

45.1.2 Préconisations concernant l'environnement de test

La méthode préconisée pour tester correctement vos paquets est d'utiliser un échantillon de machines représentatif de votre parc. Donc plus votre parc est hétérogène, plus votre échantillon devra être large.

Cette démarche vise à confronter le paquet WAPT à une multitude de plateformes et d'environments afin qu'il devienne le plus abouti possible en régime de test, avant d'être basculé en production.

45.1.3 Démarche de test

Systèmes d'exploitation et architectures

- Windows XP ;
- Windows 7 ;
- Windows 10 ;
- Windows Server 2008 R2 ;
- Windows Server 2012 ;
- x86 ;
- x64 ;
- Machine physique et virtuelle ;
- Des ordinateurs portables.

Indication: On testera si possible les versions RC / Beta des OS si elles sont disponibles (exemple : Windows 10 Creators Update).

L'état des mises à jour Windows

- **Un poste Microsoft Windows sans aucune mise à jour Windows Update** : l'objectif est de détecter les mises à jour indispensables au bon fonctionnement du logiciel et adapter le paquet en conséquence ;
- **Un poste Microsoft Windows à jour avec les toutes dernières MàJ Windows Update** : l'objectif est de détecter les mise à jour en conflit avec le logiciel et d'adapter le paquet en conséquence ;

Etat des installations des logiciels

- **Un poste avec peu de logiciels déjà installés** : l'objectif est de détecter une dépendance possible à Java ou autre framework applicatif ;
- **Les postes avec beaucoup de logiciels déjà installés** : l'objectif est de détecter un conflit avec une application existante;
- **Installer les anciennes versions du logiciel** : il est possible que l'installeur ne supporte pas l'écrasement d'une installation précédente, dans ce cas il faudra prévoir la désinstallation des anciennes versions avant d'installer la nouvelle version ;

45.2 Les principes de création d'un paquet WAPT à partir d'un modèle depuis la console

---

Indication: Pour créer des paquets à partir de la console, il faut d'abord avoir installé l'environnement de développement WAPT *tis-pyscripter*.

45.2.1 Créer un paquet WAPT depuis la console

Dans cet exemple, l'installeur de 7zip est utilisé au format MSI.

- Télécharger 7-zip MSI x64 .
- Créer le modèle de paquet depuis l'installeur.

  Dans la console WAPT, cliquer sur *Outils* → *Créer un modèle de paquet depuis un installeur*:

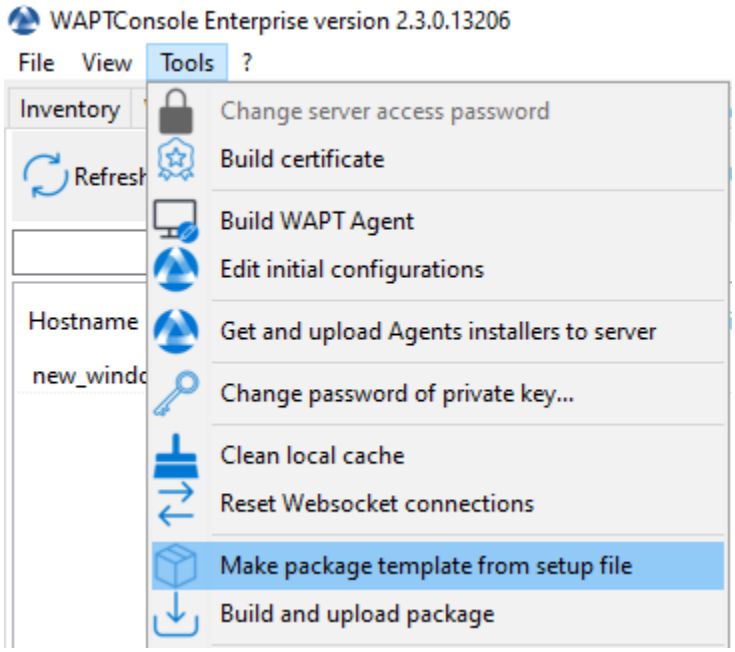


Figure1: Les principes de création d'un paquet WAPT à partir d'un modèle depuis la console

  Sélectionner l'installeur MSI téléchargé et renseigner les différentes informations demandées. Veillez bien à ce que le nom du paquet ne contienne pas de numéro de version.

- Deux solutions sont proposées:
  - Cliquer sur *Créer et éditer* (recommandée) pour lancer la personnalisation du paquet et l'adapter aux besoin spécifique de votre Organisation.
  - Cliquer sur *Créer et Téléverser* pour lancer la création et le chargement direct du paquet sur le serveur WAPT (non recommandé).

> **Attention:** Le bouton *Build and upload* envoi directement le paquet dans le dépôt privé sans tester l'installation.
>
> Cette méthode fonctionne assez bien avec les MSI car leur installation est standard.
>
> Cependant la deuxième méthode qui consiste à tester localement le paquet d'abord puis à l'uploader est la méthode recommandée.

Note: *Une ancienne méthode en ligne de commande est disponible ici.*.

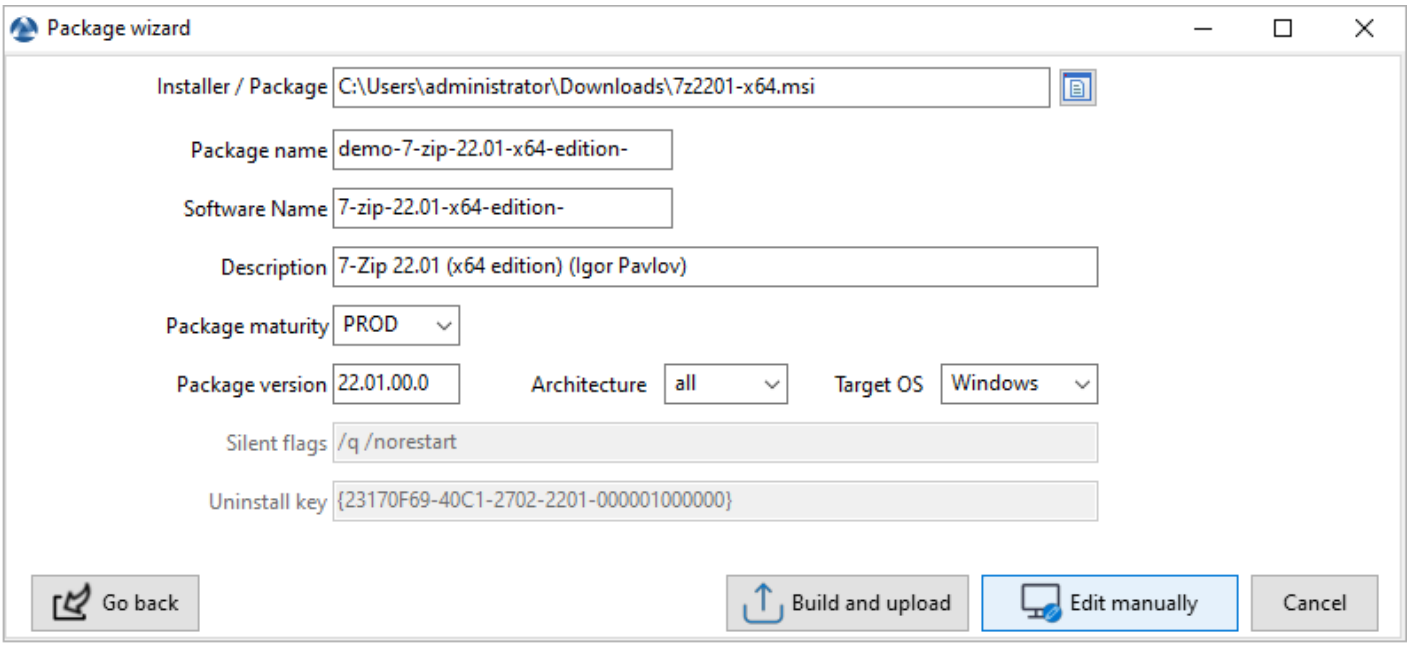


Figure2: Boîte de dialogue demandant des informations lors de la création du packaging WAPT dans la console WAPT

45.2.2 Personnaliser le paquet avant de le téléverser dans votre dépôt

La méthode conseillée avant l'"**upload** d'un paquet est de personnaliser son comportement en l'éditant avec **PyScripter**.

Lors de la création du modèle de paquet, cliquer sur *OK*.

L'IDE **PyScripter** se lance et permet d'éditer les fichiers du paquet.

45.2.3 Présentation de PyScripter

L'explorateur de projets PyScripter

L'explorateur de projets PyScripter liste les différents fichiers dont vous pouvez avoir besoin, notamment le fichier `control` et le fichier `setup.py`.

Run Configurations

Les options de **Run** dans l'explorateur de projets de **PyScripter** vont vous permettre de lancer des actions de votre paquet en cours d'édition.

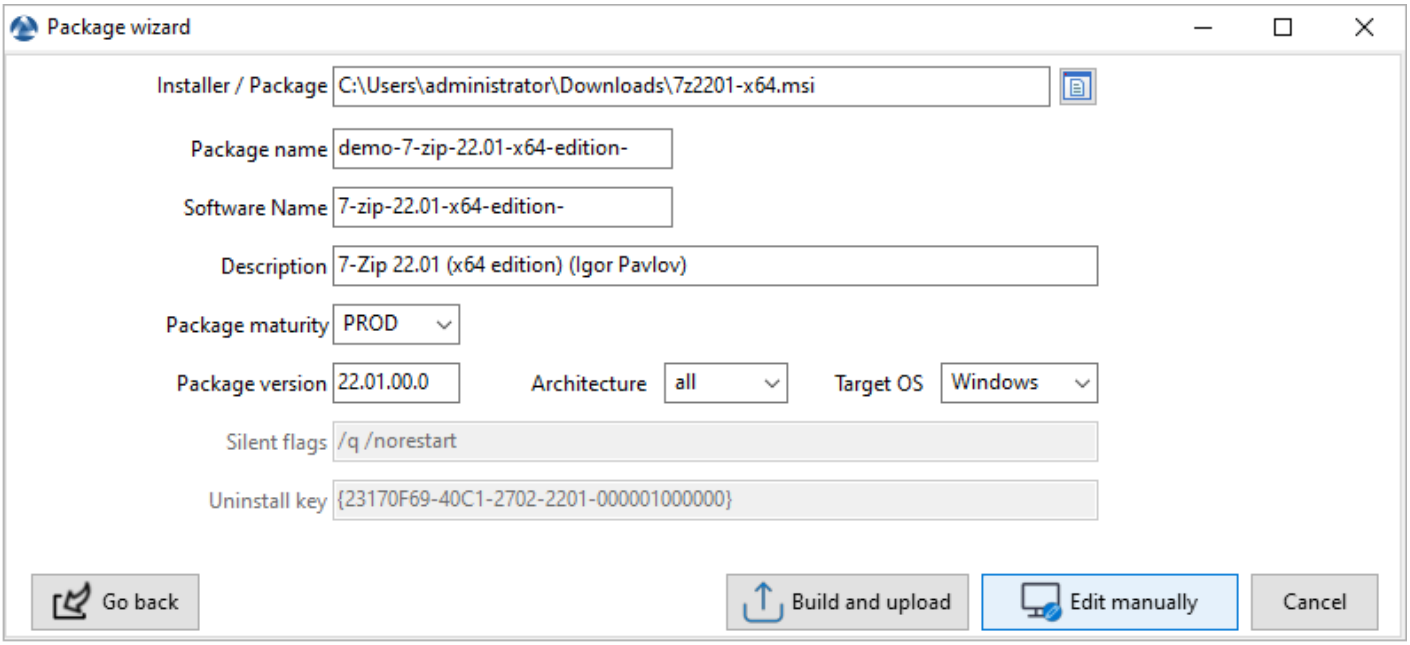


Figure3: Boîte de dialogue mettant en évidence le bouton « Make and edit … » lors de la création du packaging WAPT dans la console WAPT

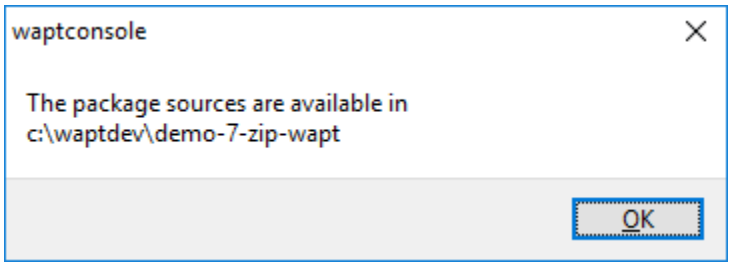


Figure4: Fenêtre de message indiquant dans la console WAPT que le packaging WAPT a été téléchargé dans le référentiel WAPT

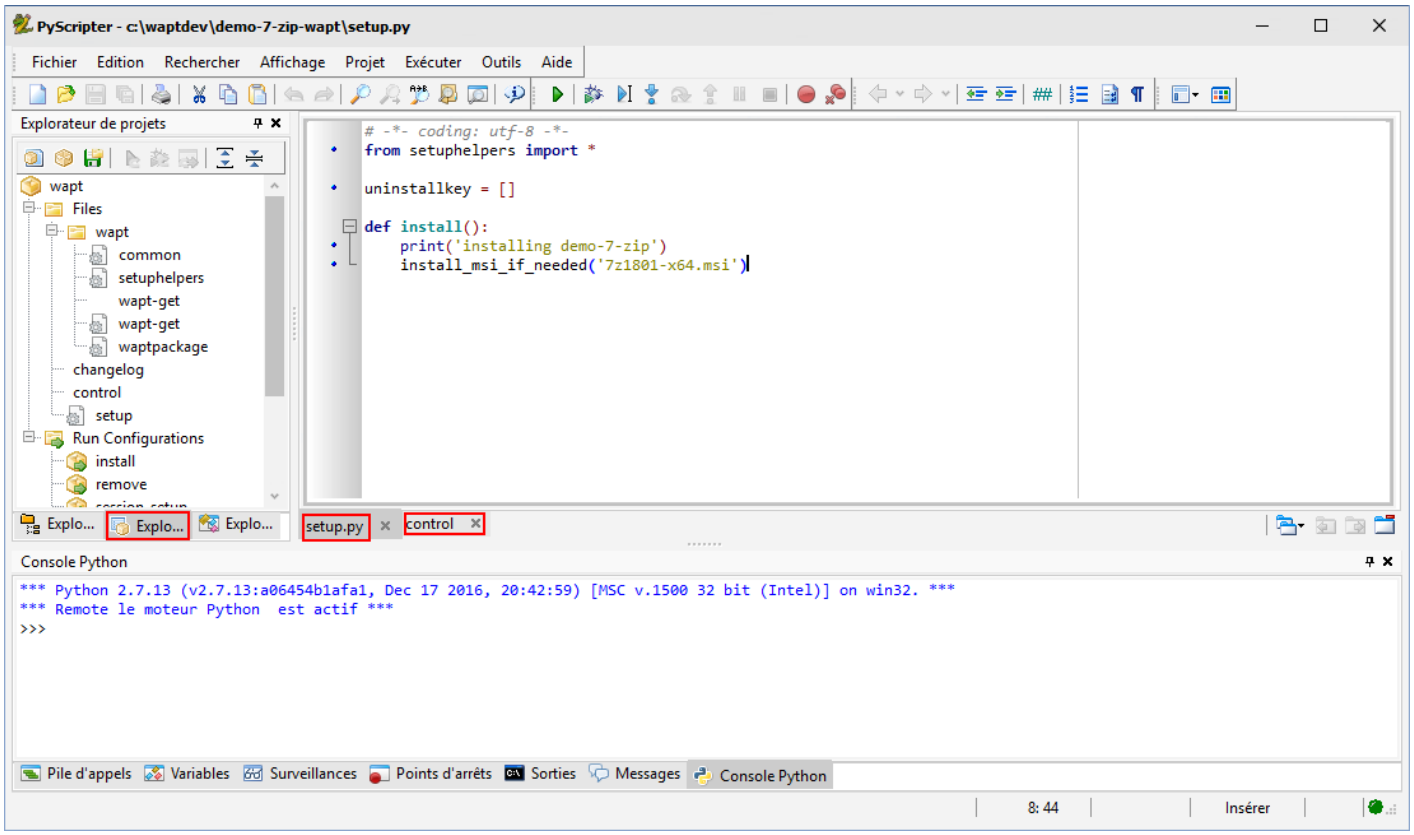


Figure5: PyScripter - Personnalisation du paquet avec PyScripter

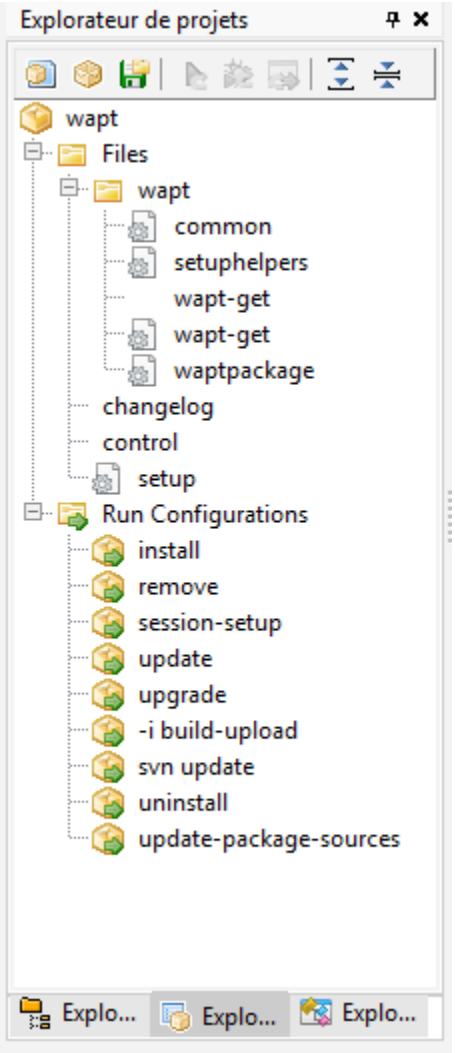


Figure6: PyScripter - Navigation d'un projet dans l'explorateur de fichiers de PyScrypter

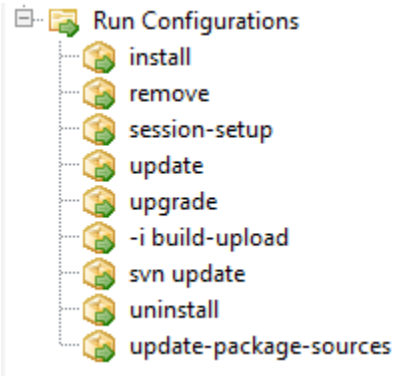


Figure7: PyScripter - Naviguer dans les configurations d'exécution d'un projet dans PyScripter

Zone d'édition

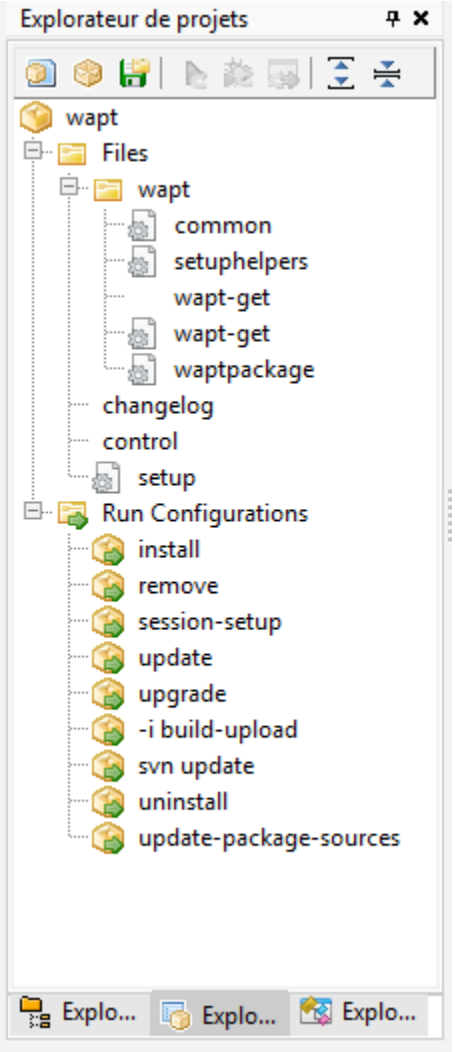


Figure8: PyScripter - Personnalisation du paquet avec PyScripter

La Zone d'édition de **PyScripter** permet d'éditer le fichier `setup.py` ainsi que le fichier `control`.

Console Python

C'est la console python visible dans **PyScripter**, elle va vous permettre d'afficher la sortie python lorsque vous exécuterez des commandes **run**.

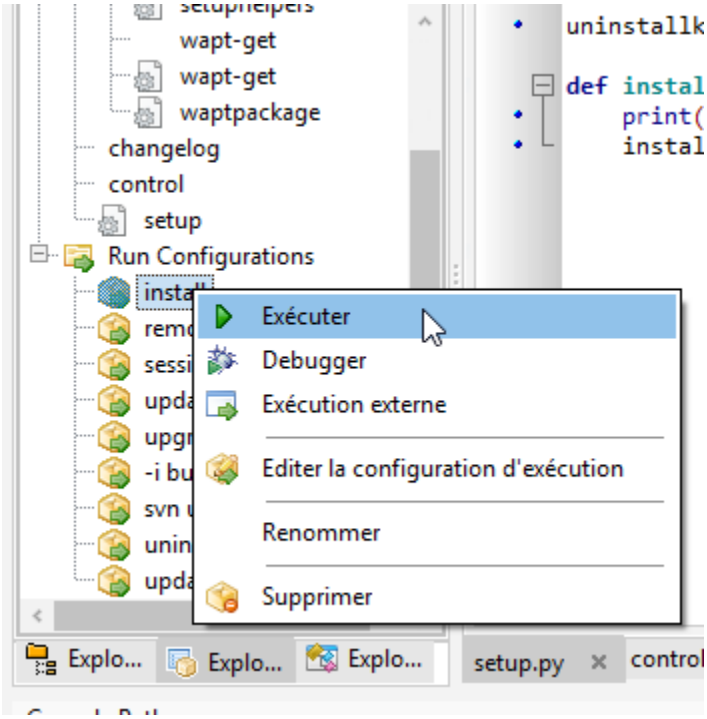
Vous pouvez également l'utiliser pour tester / déboguer des portions de votre script `setup.py`.

Pour en savoir plus sur la composition d'un paquet wapt, visitez la documentation sur la *structure d'un paquet WAPT*.

Figure9: PyScripter - console python de PyScripter

Tester localement l'installation du paquet WAPT

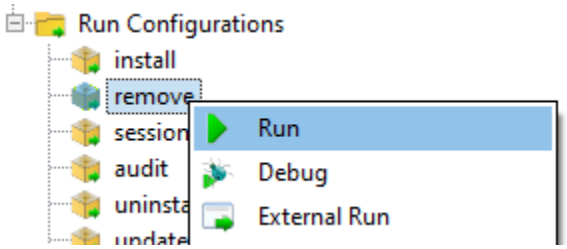
Vous pouvez ensuite tester le lancement d'une installation sur votre station de développement.



La Console PyScripter vous permet de vérifier si l'installation s'est bien déroulée.

Tester localement la désinstallation du paquet WAPT

Vous pouvez ensuite tester le lancement d'une installation sur votre station de développement.



La Console PyScripter vous permet de vérifier si l'installation s'est bien déroulée.

45.3 Creating your first WAPT Packages

45.3.1 Packager des .msi (exemple)

Pour cet exemple, nous prendrons **tightvnc**.

Vous pouvez le télécharger ici : https://www.tightvnc.com/download.php

Maintenant, vous pouvez générer votre modèle de paquet, veuillez vous référer à la *documentation pour la création de paquets à partir de la console WAPT*.

Editer le fichier `control` (`architecture`, `impacted_process`, `target_os`, `description`, `maintainer` ...), veuillez vous référer à la structure du fichier *documentation du fichier control*.

Votre **pyscripter** s'ouvre, allez dans votre `setup.py` :

```
# -*- coding: utf-8 -*-
from setuphelpers import *

uninstallkey = []

def install():
    print('installing tis-tightvnc')
    install_msi_if_needed('tightvnc-2.8.5-setup-64bit.msi')
```

- La fonction testera également si une version du logiciel est déjà installée sur la machine avec la *clé de désinstallation*.
- Si présence il y a, l'installation sera enclenchée uniquement si la version actuellement installée est plus ancienne.
- Après installation, la fonction testera finalement la présence de la clé de déinstallation et sa version pour vérifier que tout s'est bien passé.

Table1: Liste des arguments disponibles avec *install_exe_if_need*

Options (Option par défaut)	Description
`min_os_version`	Définit la version minimale au dessus de laquelle le logiciel se mettra à jour.
`killbefore` (par défaut `None`)	Liste des programmes à tuer avant de lancer l'installation.
`accept_returncodes` (par défaut `[0,3010]`)	Définit les codes de retour autres que 0 ou 3010 acceptés en retour par la fonction.
`timeout` (par défaut `300`)	Définit la durée d'attente maximale d'installation (en secondes).
`propriétés` (par défaut `None`)	Définit les propriétés supplémentaires à passer en argument au MSI pour l'installation.
`get_version` (par défaut `None`)	Définit la valeur passée en paramètre pour le contrôle de version au lieu de celle retournée par la fonction *installed_softwares*.
`remove_old_version`	Supprimme automatiquement une ancienne version d'un logiciel dont la *uninstallkey* est identique.
`force` (par défaut `False`)	Force l'installation du logiciel même si une *uninstall key* avec une version identique est trouvée.

La fonction **`install_msi_if_needed`** récupère la clé de désinstallation depuis le MSI, il n'est pas nécessaire de l'écrire dans le fichier `setup.py`.

Vous n'avez pas non plus à remplir le champ `killbefore` si la valeur indiquée dans le champ `impacted_process` du fichier `control` est correcte.

Note: Le `setup.py` aurait pu ressembler à cela, mais la méthode est moins élégante car elle fait moins de vérifications :

```
# -*- coding: utf-8 -*-
from setuphelpers import *

uninstallkey = ["{8B9896FC-B4F2-44CD-8B6E-78A0B1851B59}"]

def install():
    print('installing tis-tightvnc')
    run('msiexec /norestart /q /i "tightvnc-2.8.5-setup-64bit.msi"')
```

Lancez l'installation et voyez ce qui se passe lorsque le logiciel est déjà installé.

```
wapt-get -ldebug install C:\waptdev\tis-tightvnc-wapt
Installing WAPT file C:\waptdev\tis-tightvnc-wapt
MSI tightvnc-2.8.5-gpl-setup-64bit.msi already installed. Skipping msiexec

Results:

=== install packages ===
C:\waptdev\tis-tightvnc-wapt | tis-tightvnc (2.8.5.0-1)
```

Ajouter des propriétés supplémentaires en argument

Pour ajouter des propriétés supplémentaires on va les stocker dans un élément *dict*.

```
# -*- coding: utf-8 -*-
from setuphelpers import *

uninstallkey = []

properties = {
    'SERVER_REGISTER_AS_SERVICE':0,
    'SERVER_ADD_FIREWALL_EXCEPTION':0,
    }

def install():
    print(u'Installation en cours de TightVNC')
    install_msi_if_needed('tightvnc-2.8.5-setup-64bit.msi', properties = properties )
```

Note: Le `setup.py` aurait pu ressembler à cela, mais la méthode est moins élégante car elle fait moins de vérifications :

```
# -*- coding: utf-8 -*-
from setuphelpers import *

uninstallkey = ["{8B9896FC-B4F2-44CD-8B6E-78A0B1851B59}"]

def install():
    print('installing tis-tightvnc')
    run('msiexec /norestart /q /i "tightvnc-2.8.5-setup-64bit.msi" SERVER_REGISTER_AS_
↪SERVICE=0 SERVER_ADD_FIREWALL_EXCEPTION=0')
```

Vidéo de démonstration

https://youtu.be/Z6wr6emPGCU

45.3.2 Packager des .exe (exemple)

- Télécharger l'installateur `.exe` à partir d'une source fiable.

  Télécharger l'installateur au format exe Firefox ESR x64 sur https://download.mozilla.org/?product=firefox-esr-latest-ssl&os=win64.

- Rechercher la documentation associée pour les flags silencieux:
  - Sur le site de la Fondation Mozilla .
  - Autres méthodes pour récupérer le flag silencieux :
    - Dépôt de paquets WPKG ;
    - Dépôt de paquets Chocolatey ;

    * Recherche Internet avec le terme *Firefox silent install*.

- Puis générer votre modèle de paquet, veuillez vous référer à la *documentation pour la création de paquets à partir de la console WAPT*. **PyScripter** se charge et ouvre le projet de paquet `.exe`.

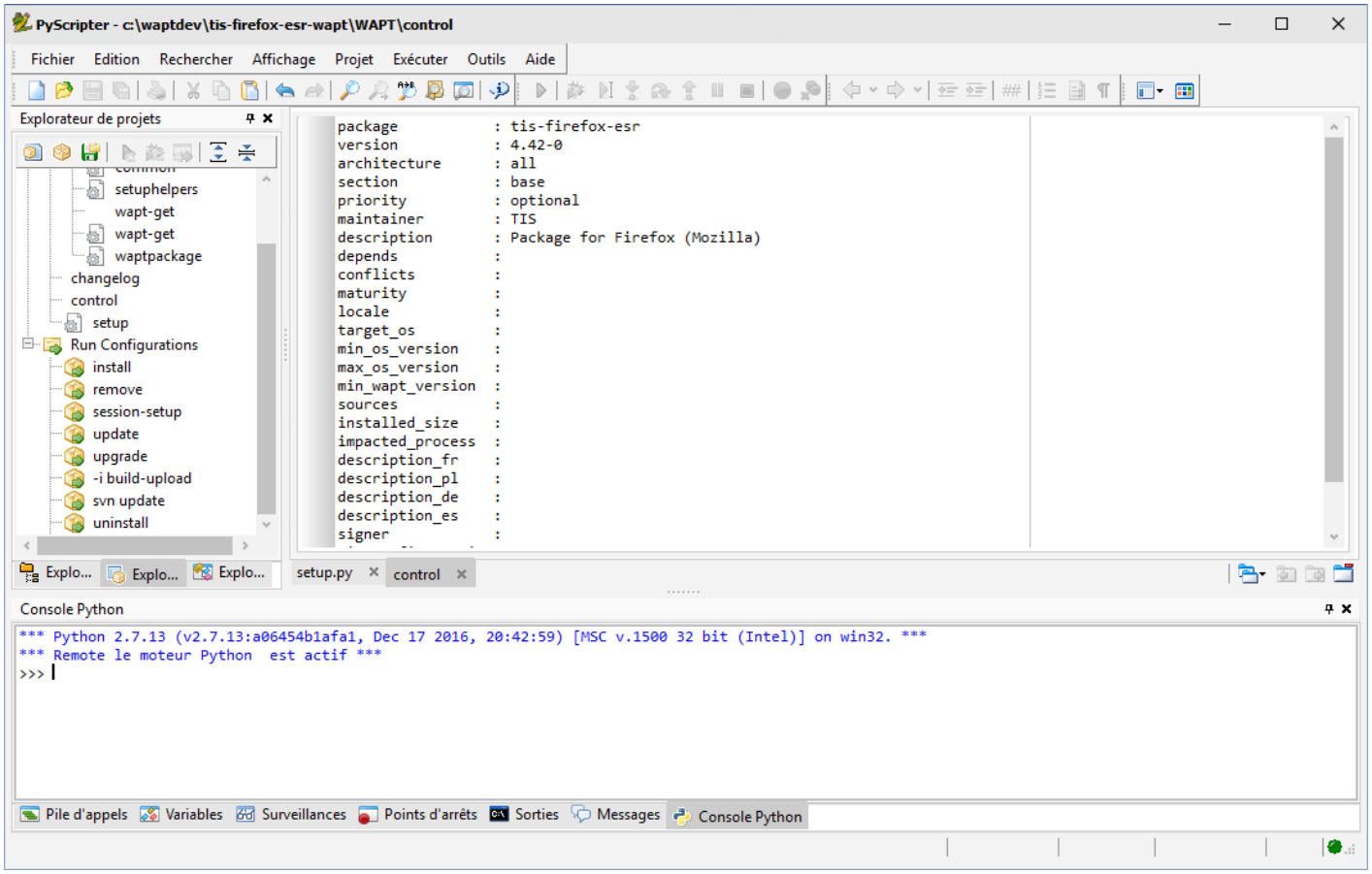


Figure10: PyScripter - Ouverture du packaging WAPT de FirefoxESR

- Editer le fichier `control` (`architecture`, `impacted_process`, `target_os`, `description`, `maintainer` ...), veuillez vous référer à la structure du fichier *documentation du fichier control*.
- Vérifier le contenu du fichier `control`. Mozilla Firefox-ESR ne répond pas aux standards et retourne un numéro de version erroné (il s'agit du numéro de version du logiciel qui créé l'installeur).
  - Fichier `control` d'origine.

```
package           : tis-firefox-esr
version           : 4.42.0.0-0
architecture      : all
section           : base
priority          : optional
maintainer        : user
description       : automatic package for firefox setup 52.6.0esr
impacted_process  :
```

  - Fichier `control` modifié.

```
package           : tis-firefox-esr
version           : 52.6.0-1
architecture      : all
section           : base
priority          : optional
maintainer        : Tranquil-IT Systems
description       : Mozilla Firefox 52.6.0 ESR
impacted_process  : firefox.exe
```

Il est à noter qu'une sous-version *-1* a été ajoutée au numéro de version du logiciel ; il s'agit de la version de packaging du paquet WAPT.

Il permet au développeur de paquets de publier plusieurs versions de paquets WAPT d'un même logiciel, ce qui est très utile pour un développement très rapide et itératif.

Utiliser *install_exe_if_needed*

La fonction est sensiblement la même que celle utilisée pour les installeurs *.msi*, avec quelques différences :

- La fonction nécessite l'ajout des flags silencieux en paramètre.
- La fonction nécessite l'ajout de la clé de désinstallation en paramètre.

```
# -*- coding: utf-8 -*-
from setuphelpers import *

uninstallkey = []

def install():
    print('installing tis-firefox-esr')
    install_exe_if_needed("Firefox Setup 45.5.0esr.exe",silentflags="-ms",key='',min_version="4.42.
↪0.0")
```

Table2: Liste des arguments disponibles avec *install_exe_if_need*

Options (Option par défaut)	Description
`silentflags` (par défaut `None`)	Paramètres silencieux à passer en argument à l'installeur.
`key` (par défaut `None`)	Clé de désinstallation du programme.
`min_os_version`	Définit la version minimale au dessus de laquelle le logiciel se mettra à jour.
`killbefore` (par défaut `None`)	Liste des programmes à tuer avant de lancer l'installation.
`accept_returncodes` (par défaut `[0,3010]`)	Définit les codes de retour autres que 0 ou 3010 acceptés en retour par la fonction.
`timeout` (par défaut `300`)	Définit la durée d'attente maximale d'installation (en secondes).
`get_version` (par défaut `None`)	Définit la valeur passée en paramètre pour le contrôle de version au lieu de celle retournée par la fonction *installed_softwares*.
`remove_old_version`	Supprimme automatiquement une ancienne version d'un logiciel dont la *uninstallkey* est identique.
`force` (par défaut `False`)	Force l'installation du logiciel même si une *uninstall key* avec une version identique est trouvée.

Le paquet aura alors ce comportement :

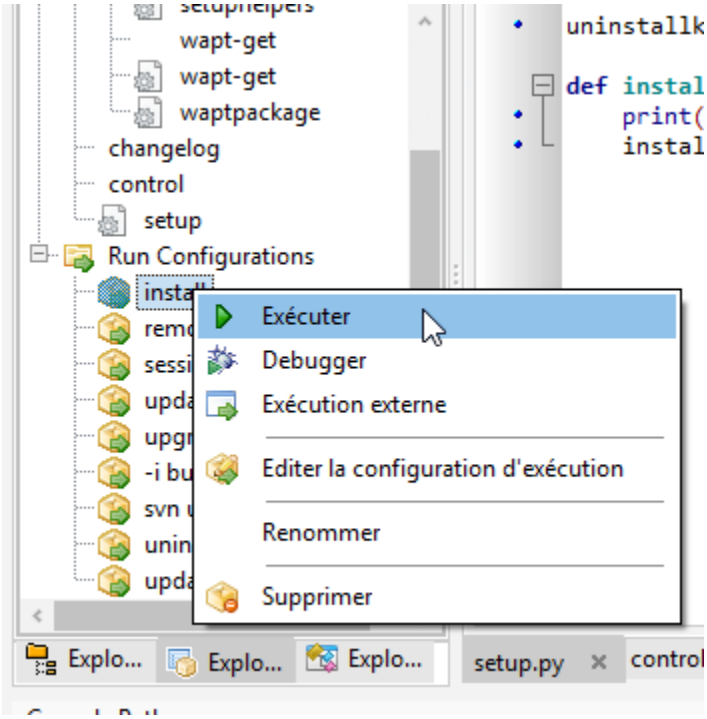
- Le logiciel Firefox s'installera uniquement si le logiciel n'est pas installé et si la version est strictement inférieure à 45.5.0, sauf si l'option `--force` est indiquée lors de l'installation du paquet.
- A l'installation, les processus **`firefox.exe`** en cours d'exécution seront tués (avec la valeur indiquée dans `impacted_process` du fichier `control`).
- La fonction ajoutera elle-même la clé de désinstallation, donc laisser l'argument *uninstallkey* vide.
- A la fin de l'installation, la fonction ira vérifier si l'*uninstallkey* est bien présente sur le poste et si la version est bien égale ou supérieure à 45.5.0, si ce n'est pas le cas, elle basculera le paquet en **ERROR**.

Trouver la clé de désinstallation

Contrairement aux fichiers *`.msi`*, la clé pour désinstaller un *`.exe`* n'est pas dans les propriétés du fichier.

Vous devez donc d'abord installer le logiciel pour connaître la clé de désinstallation.

Vous devez donc démarrer une fois l'installation à partir de **`pyscripter`** avec le *run configuration* et ensuite *install*.



Une fois le logiciel installé, allez à la console WAPT, puis trouvez votre machine de développement.

Dans l'onglet inventaire des logiciels, trouvez votre logiciel et copiez la valeur indiquée dans la colonne clé de désinstallation.

Vous devez également vérifier la valeur de la version avec la valeur indiquée dans `min_version` dans votre `setup.py`.

Modifier votre fichier `setup.py` avec les nouveaux paramètres :

```
# -*- coding: utf-8 -*-
from setuphelpers import *

uninstallkey = []
```

(suite sur la page suivante)

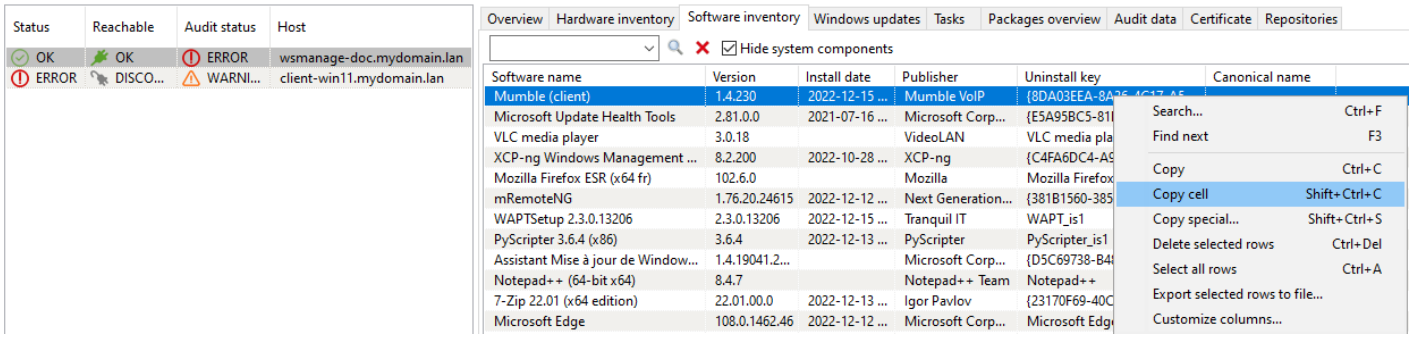
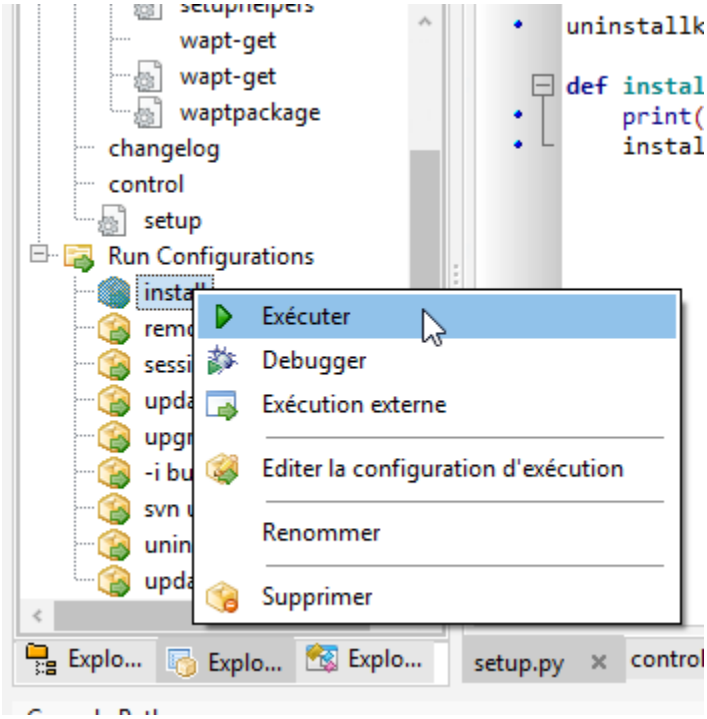


Figure11: Récupérer une clé de désinstallation depuis la console

(suite de la page précédente)

```
def install():
    print('installing tis-firefox-esr')
    install_exe_if_needed("Firefox Setup 45.5.0esr.exe",silentflags="-ms",key='Mozilla Firefox 45.5.
↪0 ESR (x64 fr)',min_version="45.5.0")
```

Pour tester que votre clé fonctionne correctement, vous devez relancer une installation dans **pyscripter**.



WAPT ne tentera pas d'installer le logiciel car il est déjà présent, le message suivant devrait donc s'afficher :

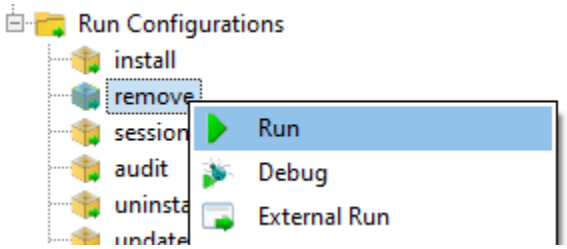
```
>>>
*** Remote Interpreter Reinitialized ***
Command Line : install "c:\waptdev\tis-firefox-esr_x64_PROD_fr-wapt\WAPT\.."
Using config file: C:\Program Files (x86)\wapt\wapt-get.ini
```

(suite sur la page suivante)

(suite de la page précédente)

```
Installing WAPT files c:\waptdev\tis-firefox-esr_x64_PROD_fr-wapt
Exe setup Firefox_Setup_78.7.1esr.exe already installed. Skipping

Results:

=== install packages ===
c:\waptdev\tis-firefox-esr_x64_PROD_fr-wapt | tis-firefox-esr (78.7.1-102)
```

Vous pouvez maintenant tester la désinstallation :



Vous pouvez maintenant construire et envoyer votre paquet, veuillez vous référer à la *documentation pour construire et envoyer des paquets depuis la console WAPT*.

Note: Si vous laissez la clé de désinstallation vide, la désinstallation de votre paquet ne fonctionnera pas.

Cas particulier d'un dé-installeur non-silencieux

Dans certains cas particuliers, un paquet utilisant **install_exe_if_needed** remplit la *clé de désinstallation*, mais la *clé de désinstallation* pointe vers un désinstalleur non silencieux.

Il nous faut contourner le problème en utilisant une fonction qui va supprimer la clé de désinstallation à la fin de l'installation.

```
:emphasize-lines: 13

# -*- coding: utf-8 -*-
from setuphelpers import *

uninstallkey = []

def install():
    install_exe_if_needed("setup.exe",
                          silentflags="/s",
                          key='{D9E87643-0005-447E-9111-78697A9C1595}',
                          min_version="14.0")
    uninstallkey.remove('{D9E87643-0005-447E-9111-78697A9C1595')

def uninstall():
    run(r'"C:\Program Files\Kutl\uninstall.exe" /supersilent')
```

Indication: La fonction de désinstallation peut également être utilisée pour exécuter du code en plus de la désinstallation de logiciels,

ex : supprimer un dossier, supprimer un raccourci ...

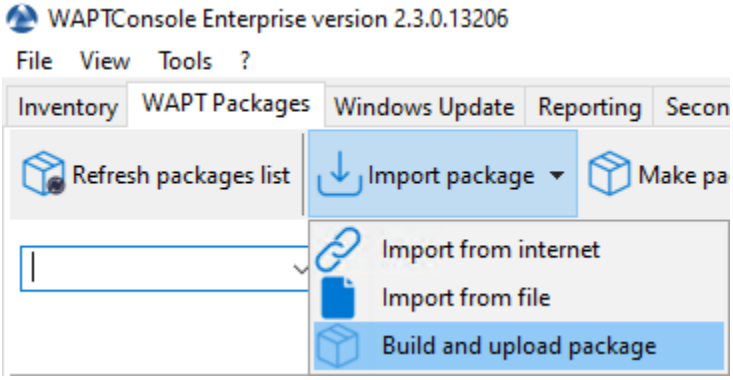
Vidéo de démonstration

https://youtu.be/z_EN2CBCTcY

45.3.3 Packager des paquets linux simples

45.3.4 Construire le paquet et l'envoyer au serveur WAPT

- Une fois que le paquet est prêt, le construire et l'envoyer au serveur WAPT, dans la console WAPT.



- Sélectionner le paquet dans le dossier `c:\waptdev`.
- Confirmer le paquet sélectionné.

Vous venez de charger votre premier paquet wapt.

Note: *Une ancienne méthode en ligne de commande est disponible ici*.

Avertissement: Une fois que votre paquet est téléversé, rafraîchissez la liste des paquets en utilisant le bouton *Actualiser les paquets disponibles* ou en appuyant sur la touche `F5` de votre clavier.

Travailler avec des codes de retour non standard

Les codes de retour sont utilisés pour indiquer si un logiciel a été correctement installé.

Avec Windows, le code standard de retour pour une installation réussie est [0].

Si vous savez que vos paquets WAPT s'installent correctement, mais que vous obtenez quand même un code de retour différent de [0], alors vous pouvez explicitement dire à WAPT d'ignorer le code d'erreur en utilisant le paramètre `accept_returncodes`.

Vous pouvez découvrir comment utiliser le paramètre `accept_returncodes` en explorant le code de ce paquet.

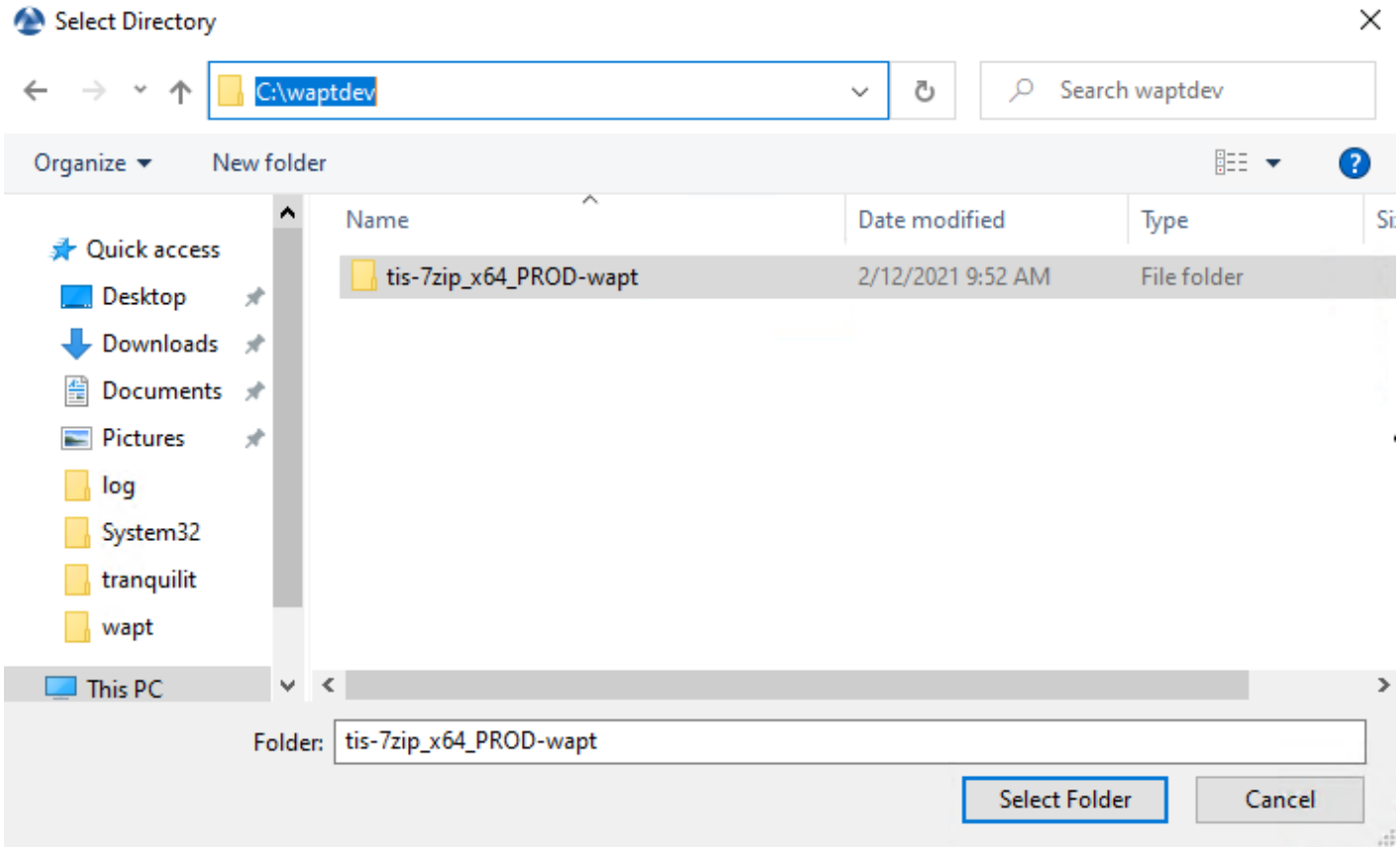


Figure12: Fenêtre du navigateur permettant de sélectionner le packaging WAPT à importer dans le référentiel privé

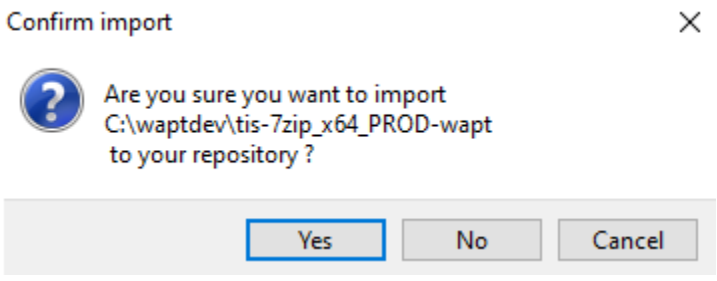


Figure13: Boîte de dialogue de la console WAPT pour confirmer l'importation d'un packaging WAPT dans le référentiel privé

```
# -*- coding: utf-8 -*-
from setuphelpers import *
import re

uninstallkey = []

def is_kb_installed(hotfixid):
    installed_update = installed_windows_updates()
    if [kb for kb in installed_update if kb['HotFixID' ].upper() == hotfixid.upper()]:
        return True
    return False

def waiting_for_reboot():
    # Query WUAU from the registry
    if reg_key_exists(HKEY_LOCAL_MACHINE,r"SOFTWARE\Microsoft\Windows\CurrentVersion\WindowsUpdate\
↪Auto Update\RebootRequired") or \
        reg_key_exists(HKEY_LOCAL_MACHINE,r"SOFTWARE\Microsoft\Windows\CurrentVersion\Component␣
↪Based Servicing\RebootPending") or \
        reg_key_exists(HKEY_LOCAL_MACHINE,r'SOFTWARE\Microsoft\Updates\UpdateExeVolatile'):
        return True
    return False

def install():
    kb_files = [
        'windows10.0-kb4522355-x64_af588d16a8fbb572b70c3b3bb34edee42d6a460b.msu',
        ]
    with EnsureWUAUServRunning():
      for kb_file in kb_files:
          kb_guess = re.findall(r'^.*-(KB.*)-',kb_file)
          if not kb_guess or not is_kb_installed(kb_guess[0]):
              print('Installing {}'.format(kb_file))
              run('wusa.exe "{}" /quiet /norestart'.format(kb_file),accept_returncodes=[0,3010,
↪2359302,-2145124329],timeout=3600)
          else:
              print('{} already installed'.format(kb_file))

      if waiting_for_reboot():
          print('A reboot is needed!')
```

Indication: La liste complète des messages d'erreur de l'installeur Windows peut être consultée sur cette page <https://docs.microsoft.com/en-us/windows/win32/msi/windows-installer-error-messages>`_.

CHAPTER 46

How to code WAPT packages

46.1 Exemples simples de fonctions du setuphelper couramment utilisées

Nous présentons ici quelques fonctions implémentées dans *Setuphelpers* et fréquemment utilisées pour développer des paquets WAPT.

46.1.1 Tests et manipulation de dossiers et fichiers

Créer un chemin avec récursion

... fabrique la variable pour le chemin `C:\Program Files (x86)\Mozilla\Firefox`.

```
makepath(programfiles,'Mozilla','Firefox')
```

Créer et supprimer des raccourcis

... crée le répertoire `C:\test`.

```
mkdirs('C:\\test')
```

... détruit le répertoire `C:\tmp\target`.

```
remove_tree(r'C:\tmp\target')
```

Vérifier si un chemin est un fichier ou un dossier

… vérifie que `C:\Program Files (x86)\software` est un répertoire.

```
isdir(makepath(programfiles32,'software')):
    print('The directory exists')
```

… vérifie que `C:\Program Files (x86)\software\file` est un fichier.

```
isfile(makepath(programfiles32,'software','file')):
    print('file exist')
```

Vérifier si un répertoire est vide

… vérifie que le répertoire `C:\Program Files (x86)\software` est vide.

```
dir_is_empty(makepath(programfiles32,'software')):
    print('dir is empty')
```

Copier un fichier

… copie le fichier `file.txt` dans le répertoire `C:\Program Files (x86)\software`.

```
filecopyto('file.txt',makepath(programfiles32,'software'))
```

Copier un dossier

… copie le dossier `sources` dans le répertoire `C:\projet`.

```
copytree2('sources','C:\\projet')
```

46.1.2 Manipulation de clés de registre

Vérifier l'existence d'une clé de registre

La commande **`registry_readstring`** vérifie si la clé de registre *{8A69D345-D564-463c-AFF1-A69D9E530F96}* existe dans le chemin de registre `SOFTWARE\Google\Update\Clients` de *HKEY_LOCAL_MACHINE*.

```
if registry_readstring(HKEY_LOCAL_MACHINE, "SOFTWARE\\Google\\Update\\Clients\\{8A69D345-D564-463c-
↪AFF1-A69D9E530F96}", 'pv'):
    print('key exist')
```

Afficher la valeur d'une clé de registre

La commande **registry_readstring** lit la valeur *{8A69D345-D564-463c-AFF1-A69D9E530F96}* stockée dans le chemin de registre `SOFTWARE\Google\Update\Clients` de *HKEY_LOCAL_MACHINE*.

```
print(registry_readstring(HKEY_LOCAL_MACHINE, r'SOFTWARE\Google\Update\Clients\{8A69D345-D564-463c-
↪AFF1-A69D9E530F96}', 'pv'))
```

Modifier la valeur d'une clé de registre

La commande **registry_setstring** modifie la valeur de la clé de registre *TOUVersion* stockée dans le chemin de registre `SOFTWARE\Microsoft\Windows Live` de *HKEY_CURRENT_USER*.

```
registry_setstring(HKEY_CURRENT_USER, "SOFTWARE\\Microsoft\\Windows Live\\Common",'TOUVersion','16.
↪0.0.0', type=REG_SZ)
```

46.1.3 Créer et supprimer des raccourcis

Avec WAPT setuphelper, il est possible de créer différents types de raccourcis.

Créer un raccourci du bureau pour tous les utilisateurs

La commande **create_desktop_shortcut** crée le raccourci *Gestion de la console WAPT* dans le répertoire `C:\Users\Public` pointant sur `C:\Program Files (x86)\wapt\waptconsole.exe` ; le raccourci est disponible pour tous les utilisateurs.

```
create_desktop_shortcut(r'WAPT Console Management',target=r'C:\Program Files (x86)\wapt\waptconsole.
↪exe')
```

Removing a desktop shortcut for all users

La commande **remove_desktop_shortcut** supprime le raccourci *WAPT Console Management* du dossier `C:\Users\Public` ; le raccourci est supprimé pour tous les utilisateurs.

```
remove_desktop_shortcut('WAPT Console Management')
```

Firefox place un raccourci sur le bureau de tous les utilisateurs, nous allons le supprimer.

Nous utiliserons la fonction `remove_desktop_shortcut` :

- Modifier le `setup.py` et utiliser la fonction comme ceci.

```
# -*- coding: utf-8 -*-
from *SetupHelpers* import *

uninstallkey = []

def install():
```

(suite sur la page suivante)

(suite de la page précédente)

```
    install_exe_if_needed("Firefox Setup 45.5.0esr.exe",silentflags="-ms",key='Mozilla Firefox␣
↪45.5.0 ESR (x64 fr)',min_version="45.5.0")
    remove_desktop_shortcut('Firefox')
```

- Si vous redémarrez l'installation à partir de **pyscripter**, vous remarquerez que le raccourci de bureau « all users » a disparu.

Créer un raccourci de menu pour une application

La commande **create_programs_menu_shortcut** crée le raccourci *WAPT Console Management* dans le menu démarrer pointant sur `C:\Program Files (x86)\wapt\waptconsole.exe` ; le raccourci est disponible pour tous les utilisateurs.

```
create_desktop_shortcut(r'WAPT Console Management',target=r'C:\Program Files (x86)\wapt\waptconsole.
↪exe')
```

Supprimer un raccourci de menu pour une application

La commande **remove_programs_menu_shortcut** supprime le raccourci *WAPT Console Management* du menu démarrer.

```
remove_programs_menu_shortcut('WAPT Console Management')
```

Créer un raccourci du bureau pour un utilisateur connecté

Indication: Ces fonctions sont utilisées avec le session_setup.

La commande **create_user_desktop_shortcut** crée le raccourci *WAPT Console Management* sur le bureau de l'utilisateur en pointant sur `C:\Program Files (x86)\wapt\waptconsole.exe`.

```
create_user_desktop_shortcut(r'WAPT Console Management',target=r'C:\Program Files (x86)\wapt\
↪waptconsole.exe')
```

Removing a desktop shortcut for a logged in user

La commande **remove_user_desktop_shortcut** supprime le raccourci *WAPT Console Management* du bureau de l'utilisateur connecté.

```
remove_user_desktop_shortcut('WAPT Console Management')
```

Créer un raccourci de menu vers une application pour un utilisateur spécifique

Indication: Ces fonctions sont utilisées avec le session_setup.

La commande **create_user_programs_menu_shortcut** crée le raccourci *WAPT Console Management* dans le menu de démarrage de l'utilisateur en pointant sur `C:\Program Files (x86)\wapt\waptconsole.exe`.

```
create_user_programs_menu_shortcut(r'WAPT Console Management',target=r'C:\Program Files (x86)\wapt\
↪waptconsole.exe')
```

Supprimer un raccourci de menu vers une application pour un utilisateur spécifique

La commande **remove_user_programs_menu_shortcut** supprime le raccourci *WAPT Console Management* du menu de démarrage de l'utilisateur connecté.

```
remove_user_programs_menu_shortcut('WAPT Console Management')
```

46.1.4 Manipulating ini files

Read a value in a section of a ini file

Command **inifile_readstring** will read a value from a key and a section of a ini file.

```
inifile_writestring("file.ini","global","key")
```

Write a value in a section of a ini file

Command **inifile_writestring** will modify a value from a key and a section of a ini file.

```
inifile_writestring("file.ini","global","key","value")
```

Delete a key in a section of a ini file

Command **inifile_deleteoption** will delete a key in a given section of a ini file.

```
inifile_deleteoption("file.ini","global","key")
```

Delete an entire section of a ini file

Command **inifile_deletesection** will delete a section of a ini file and all of its content.

```
inifile_deletesection("file.ini","global")
```

46.1.5 Environnement Windows / Logiciel / Services

Récupérer la version d'un fichier

La commande **get_file_properties** ...

```
get_file_properties(makepath(programfiles32,'InfraRecorder','infrarecorder.exe'))['ProductVersion']
```

Vérifier la version de Windows

La commande **windows_version** vérifie que la version de Windows est strictement inférieure à *6.2.0*.

```
windows_version()<Version('6.2.0'):
```

Indication: For more informations you can visit Microsoft Windows version number.

Vérifier l'architecture 64bits

... vérifie que le processeur de la machine est 64bits.

```
if iswin64():
    print('Pc x64')
else:
    print('Pc not x64')
```

Vérifier la variable Program Files

- programfiles;

  ```
  print(programfiles())
  ```

- programfiles32;

  ```
  print(programfiles32())
  ```

- programfiles64;

  ```
  print(programfiles64())
  ```

Chaque commande renvoie un emplacement de *Program Files* différent.

Par exemple, la commande **programfiles64** renvoie le répertoire natif de Program Files, par exemple `C:\Program Files (x86)` sur l'architecture win64 ou win32 et **programfiles()** renverra le chemin du répertoire Program Files 32bit, par exemple. `Programs Files (x86)` sur une architecture win64, et `Programs Files` sur une architecture win32.

Vérifier la variable AppData

user_appdata / user_local_appdata

Indication: Ces fonctions sont utilisées avec le **session_setup**

… renvoie le profil *appdata* itinérant de l'utilisateur courant (`C:\Users\%username%\AppData\Roaming`).

```
print(user_appdata())
```

… renvoie le profil *appdata* local de l'utilisateur courant (`C:\Users\%username%\AppData\Local`).

```
print(user_local_appdata())
```

Désactiver temporairement la redirection de fichiers wow3264

La commande **disable_file_system_redirection** …

```
with disable_file_system_redirection():
    filecopyto('file.txt',system32())
```

Obtenir l'utilisateur connecté actuel

La commande **get_current_user** affiche le nom d'utilisateur actuellement connecté.

```
print(get_current_user())
```

Obtenir le nom de l'ordinateur

La commande **get_computername** …

```
print(get_computername())
```

Obtenir le domaine AD auquel l'ordinateur est joint

… renvoie le nom de la machine avec le domaine.

```
get_domain_fromregistry()
```

46.1.6 Actions sur les logiciels installés

Vérifier les logiciels installés

… renvoie la liste des logiciels inscrits dans la base de registre machine sous forme de tableau.

```
installed_softwares('winscp')
```

```
[{'install_location': u'C:\\Program Files\\WinSCP\\', 'version': u'5.9.2', 'name': u'WinSCP 5.9.2',
↪'key': u'winscp3_is1', 'uninstall_string': u'"C:\\Program Files\\WinSCP\\unins000.exe"',
↪'publisher': u'Martin Prikryl', 'install_date': u'20161102', 'system_component': 0}]
```

Récupérer la commande de désinstallation avec le registre

… renvoie la commande de désinstallation silencieuse.

```
uninstall_cmd('winscp3_is1')
```

```
"C:\Program Files\WinSCP\unins000.exe" /SILENT
```

Désinstaller des logiciels

```
for soft in installed_softwares('winscp3'):
    if Version(soft['version']) < Version('5.0.2'):
        run(WAPT.uninstall_cmd(soft['key']))
```

- Pour chaque élément de la liste renvoyée par *installed_softwares* contenant le mot-clé *winscp*.
- Si la version dans la liste est plus petite que 5.0.2.
- Alors lancer la désinstallation avec *uninstall_cmd* et en indiquant la *uninstallkey*.

Tuer des tâches

La commande **killalltasks** tue toutes les tâches portant le nom spécifié.

```
killalltasks('firefox')
```

46.1.7 Utiliser les champs du fichier control

Il est possible d'utiliser les informatoins du fichier control dans le `setup.py`

Récupérer la version du paquet

```
def setup():
    print(control['version'])
```

… affiche le champ `version` du fichier `control` du paquet WAPT.

```
def setup():
    print(control['version'].split('-',1)[0])
```

… affiche le numéro de version du fichier `control` sans le numéro de version de packaging WAPT.

Récupérer le nom du logiciel

À faire: documentation à venir

46.1.8 Gérer un paquet WAPT avec un autre paquet WAPT

Installer un paquet

La commande **`install`** …

```
WAPT.install('tis-scratch')
```

… installe *tis-scratch* sur la machine.

Supprimer un paquet

La commande **`remove`** …

```
WAPT.remove('tis-scratch')
```

… désinstalle *tis-scratch* de la machine.

Créer des paquets WAPT

La commande **forget_packages** ...

```
WAPT.forget_packages('tis-scratch')
```

... informe WAPT de ne plus suivre le paquet *tis-scratch* ; WAPT ne connaîtra plus l'existence de ce paquet.

Indication: Si vous voulez supprimer *tis-scratch*, il faudra soit réinstaller le paquet (**wapt-get install "tis-scratch"**), puis le supprimer (**wapt-get remove "tis-scratch"**), ou bien le supprimer manuellement à partir du panneau de configuration Windows *Ajout / Suppression de Programmes*.

46.2 Améliorer mon paquet

46.2.1 Copier un fichier

Il est possible de configurer **Firefox** avec un fichier `policies.json`. Voir https://github.com/mozilla/policy-templates/blob/master/README.md.

Ce fichier doit être placé dans le dossier `distribution` à la racine de Firefox.

Pour vous aider à créer ce fichier `policies.json`, vous pouvez utiliser cette extension : https://addons.mozilla.org/fr/firefox/addon/enterprise-policy-generator/.

Lorsque vous avez généré votre fichier `policies.json`, placez-le dans `c:\waptdev\prefix-firefox-esr-wapt\policies.json`.

Le dossier `distribution` à la racine de Firefox peut ne pas exister, nous allons donc tester son existence et le créer avec la commande **mkdirs** si il n'existe pas :

```
if not isdir(r'C:\Program Files\Mozilla Firefox\distribution'):
    mkdirs(r'C:\Program Files\Mozilla Firefox\distribution')
```

Important: Si vous avez des *backslashes* sur votre chemin, vous devez toujours mettre un **r** devant la chaîne, comme dans l'exemple précédent.

Vous devrez également utiliser la fonction `filecopyto` pour copier le fichier `policies.json` :

```
filecopyto('policies.json',r'C:\Program Files\Mozilla Firefox\distribution')
```

Indication: Il n'est pas nécessaire de mettre le chemin complet du fichier source puisque le fichier `policies.json` est à la racine du paquet WAPT, donc nous utilisons le chemin relatif.

Modifier le `setup.py` :

```
# -*- coding: utf-8 -*-
from setuphelpers import *
```

(suite sur la page suivante)

(suite de la page précédente)

```
uninstallkey = []

def install():
    install_exe_if_needed("Firefox Setup 45.5.0esr.exe",silentflags="-ms",key='Mozilla Firefox 45.5.
↪0 ESR (x64 fr)',min_version="45.5.0")
    remove_desktop_shortcut('Firefox')

    if not isdir(r'C:\Program Files\Mozilla Firefox\distribution'):
        mkdirs(r'C:\Program Files\Mozilla Firefox\distribution')

    filecopyto('policies.json',r'C:\Program Files\Mozilla Firefox\distribution')
```

Votre paquet est maintenant prêt à appliquer une configuration. Vous pouvez lancer une installation avec **pyscripter** et valider que le paquet fonctionne selon votre objectif.

Enfin, lancer votre **Firefox** pour vérifier qu'il fonctionnera pour vos utilisateurs.

46.2.2 Désinstaller des versions non désirées

Indication: À chaque étape de ces exemples, vous pouvez lancer une installation pour tester le résultat.

Dans notre cas, nous voulons désinstaller la version non ESR de **Firefox**.

Nous chercherons les autres logiciels installés sur la machine pour vérifier si une version non-esr de **Firefox** est installée.

Pour reproduire notre exemple, téléchargez et installez la version grand public ici : https://download.mozilla.org/?product=firefox-latest-ssl&os=win :

- Pour rechercher une version non désirée de **Firefox**, nous utiliserons la fonction `installed_softwares`. Cette fonction renvoie un dictionnaire contenant les propriétés du logiciel:

```
print(installed_softwares('Firefox'))

[
   {
 'install_date': '',
   'install_location': 'C:\\Program Files\\Mozilla Firefox',
   'key': 'Mozilla Firefox 78.7.1 ESR (x64 fr)',
   'name': 'Mozilla Firefox 78.7.1 ESR (x64 fr)',
   'publisher': 'Mozilla',
   'system_component': 0,
   'uninstall_string': '"C:\\Program Files\\Mozilla Firefox\\uninstall\\helper.exe"',
   'version': '78.7.1',
   'win64': True
 },

   {
  'install_date': '',
    'install_location': 'C:\Program Files (x86)\\Mozilla Firefox',
```

(suite sur la page suivante)

(suite de la page précédente)

```
    'key': 'Mozilla Firefox 79.0 (x86 fr)',
    'name': 'Mozilla Firefox 79.0 (x86 fr)',
    'publisher': 'Mozilla',
    'system_component': 0,
    'uninstall_string': '"C:\Program Files (x86)\\Mozilla Firefox\\uninstall\\helper.exe"',
    'version': '79.0',
    'win64': False
  }
]
```

- Vérifier le nom de chaque logiciel.

```
for uninstall in installed_softwares('Mozilla Firefox'):
    print(uninstall['name'])
```

- Afficher le nom de chaque logiciel trouvé.

```
for uninstall in installed_softwares('Mozilla Firefox'):
    if not 'ESR' in uninstall['name']
        print(uninstall['name'])
```

- Afficher le nom de chaque logiciel trouvé qui n'inclut pas la chaîne *ESR* dans son nom et sa clé de désinstallation.

```
for uninstall in installed_softwares('Mozilla Firefox'):
    if not 'ESR' in uninstall['name']
        print(uninstall['name'])
        print('Uninstall ' + uninstall['key'])
```

Nous allons maintenant utiliser une astuce WAPT en utilisant la fonction `uninstall_cmd` :

- Install cmd accepte une clé de désinstallation comme argument et enverra la commande à exécuter pour lancer la désinstallation silencieuse.

```
for uninstall in installed_softwares('Mozilla Firefox'):
    if not 'ESR' in uninstall['name']:
        print(uninstall['name'])
        print('Uninstall ' + uninstall['key'])
        silent_uninstall = uninstall_cmd(uninstall['key'])
        print('Run ' + silent_uninstall)
```

- Commencer la désinstallation.

```
for uninstall in installed_softwares('Mozilla Firefox'):
    if not 'ESR' in uninstall['name']:
        print(uninstall['name'])
        print('Uninstall ' + uninstall['key'])
        silent_uninstall = uninstall_cmd(uninstall['key'])
        print('Run ' + silent_uninstall)
        run(silent_uninstall)
```

Nous pouvons également désinstaller le service de maintenance de mozilla:

```
for uninstall in installed_softwares('MozillaMaintenanceService'):
    run(uninstall_cmd(uninstall['key']))
```

- Enfin, modifier votre `setup.py`:

```
# -*- coding: utf-8 -*-
from setuphelpers import *

uninstallkey = []

def install():
    #Install firefox if necessary
    install_exe_if_needed("Firefox Setup 45.5.0esr.exe",silentflags="-ms",key='Mozilla Firefox␣
↪45.5.0 ESR (x64 fr)',min_version="45.5.0")

    #Removal of the firefox shortcut on the all user desktop
    remove_desktop_shortcut('Firefox')

    #Creation of the distribution folder if it does not exist
    if not isdir(r'C:\Program Files\Mozilla Firefox\distribution'):
        mkdirs(r'C:\Program Files\Mozilla Firefox\distribution')

    #Copy of the policies.json file found at the root of the package in the destination of the␣
↪distribution folder
    filecopyto('policies.json',r'C:\Program Files\Mozilla Firefox\distribution')

    #For each Mozilla Firefox installed
    for uninstall in installed_softwares('Mozilla Firefox'):
        #If the software does not have the word ESR in the name
        if not 'ESR' in uninstall['name']:
            print(uninstall['name'])
            print('Uninstall ' + uninstall['key'])

            #Looking for how we can uninstall it silently
            silent_uninstall = uninstall_cmd(uninstall['key'])
            print('Run ' + silent_uninstall)

            #We launch the previous command.
            run(silent_uninstall)

    #Uninstalling mozilla maintenance service
    for uninstall in installed_softwares('MozillaMaintenanceService'):
        run(uninstall_cmd(uninstall['key']))
```

Votre code gère maintenant la désinstallation des versions non désirées de **Firefox**.

46.2.3 Améliorer setup.py pour utiliser des variables

Exemples d'utilisation de variables :

```
version_firefox = "45.0"

uninstallkey = "Mozilla Firefox " + version_firefox + " ESR (x64 fr)"
print(uninstallkey)

uninstallkey = "Mozilla Firefox %s ESR (x64 fr)" % (version_firefox)
print(uninstallkey)

uninstallkey = "Mozilla Firefox {} ESR (x64 fr)".format(version_firefox)
print(uninstallkey)

uninstallkey = f"Mozilla Firefox {version_firefox} ESR (x64 fr)"
print(uninstallkey)
```

Important: Le dernier exemple est le meilleur mais cette opération ne fonctionne qu'avec **Python3**.

Nous pouvons maintenant utiliser des variables dans notre fichier `setup.py` :

```
# -*- coding: utf-8 -*-
from setuphelpers import *

uninstallkey = []

def install():

    version_firefox = "45.5.0"

    #Install firefox if necessary
    install_exe_if_needed("Firefox Setup %sesr.exe" % version_firefox,silentflags="-ms",
↪key='Mozilla Firefox %s ESR (x64 fr)' % version_firefox,min_version=version_firefox)

    #Removal of the firefox shortcut on the all user desktop
    remove_desktop_shortcut('Firefox')

    distribution_folder=r'C:\Program Files\Mozilla Firefox\distribution'

    #Creation of the distribution folder if it does not exist
    if not isdir(distribution_folder):
        mkdirs(distribution_folder)

    ... The rest of the code does not change ...
```

Indication: Vous pouvez récupérer le numéro de version indiqué dans le fichier `control` comme ceci :

```
version_firefox = control.get_software_version()
```

46.2.4 Personnaliser le contexte utilisateur

Il est parfois nécessaire de personnaliser un programme ou un logiciel en contexte utilisateur pour rendre le logiciel immédiatement exploitable par l'utilisateur dans le contexte spécifique de son entreprise ou du service au sein de son entreprise :

- Créer des raccourcis sur le bureau utilisateur avec des arguments spécifiques.
- Modifier de clés registres utilisateurs.
- Modifier des fichiers, une configuration de navigateur.
- Configurer des raccourcis réseaux aux modèles de documents de l'entreprise pour assurer la conformité des documents aux chartes éditoriales en vigueur.
- Paramétrer la messagerie instantannée ou le mail de l'utilisateur à partir d'un référentiel externe (annuaire, base de données, etc) .
- Paramétrer un logiciel bureautique ou métier à partir d'un référentiel externe (annuaire, base de données, etc).

La fonction **session_setup** bénéficie de toute la puissance et de l'étendue des librairies python pour atteindre un niveau d'automatisation élevé.

Les principes du *session_setup*

La fonction WAPT : command:*session_setup* est exécutée pour chaque utilisateur utilisant cette commande:

```
C:\Program Files (x86)\wapt\wapt-get.exe session-setup ALL
```

L'appel à cette fonction permet d'exécuter la partie **session_setup** de chaque paquet WAPT logiciel installé sur la machine.

WAPT enregistre en base locale les instructions de tous les paquets dans le fichier `C:\Program Files (x86)\wapt\waptdb.sqlite`.

Attention: Le **session_setup** de chaque paquet n'est exécuté qu'**une seule fois par paquet ou version de paquet et par profil utilisateur**.

L'agent WAPT stocke dans la base de données locale `%appdata%\wapt\waptsession.sqlite` les instances de **session_setup** qui ont déjà été jouées.

Exemple de sortie de la commande `wapt-get session-setup ALL` :

Note: Le *session_setup* de l'utilisateur courant a déjà été lancé précédemment.

```
wapt-get session-setup ALL

Configuring tis-7zip ... No session-setup. Done
Configuring tis-ccleaner ... Already installed. Done
Configuring tis-vlc ... No session-setup. Done
Configuring tis-tightvnc ... No session-setup. Done
```

(suite sur la page suivante)

(suite de la page précédente)

```
Configuring tis-paint.net ... No session-setup. Done
Configuring wsuser01.mydomain.lan ... No session-setup. Done
```

Utiliser le session-setup

Les scripts `session_setup` sont situés dans la section *def session_setup()* du fichier `setup.py` :

Exemple :

```
def session_setup():
  registry_setstring(HKEY_CURRENT_USER, "SOFTWARE\\Microsoft\\Windows Live\\Common",'TOUVersion',
↪'16.0.0.0', type=REG_SZ)
```

Attention: Avec **session_setup**, il n'est pas possible de faire appel à des fichiers contenus dans le paquet.

Pour appeler des fichiers externes lors de la désinstallation, copier et coller les fichiers nécessaires dans un dossier externe pendant le processus d'installation du paquet (exemple : `c:cachefile`).

Exemple : Créer un raccourci personnalisé sur le bureau

Une des possibilités offertes par *Setuphelpers* est la création de raccourcis individuels sur le bureau utilisateur, à la différence du bureau « Public » commun à tous les utilisateurs.

Nous utiliserons pour ça la fonction `create_user_desktop_shortcut()` pour créer un raccourci contenant le nom de l'utilisateur et qui passera en argument à Firefox le site https://tranquil.it par exemple.

```
# -*- coding: utf-8 -*-
from setuphelpers import *

uninstallkey = []

def install():
    install_exe_if_needed("Firefox Setup 45.5.0esr.exe",silentflags="-ms",key='Mozilla Firefox 45.4.
↪0 ESR (x64 fr)',min_version="45.5.0")

def session_setup():
  create_user_desktop_shortcut("Mozilla Firefox de %s" % get_current_user(),r'C:\Program Files\
↪Mozilla Firefox\firefox.exe',arguments="-url https://tranquil.it")
```

- Maintenant, lancer le `session-setup` directement à partir de **pyscripter**.
- Enfin, vérifier que l'icône est bien présente sur le bureau.

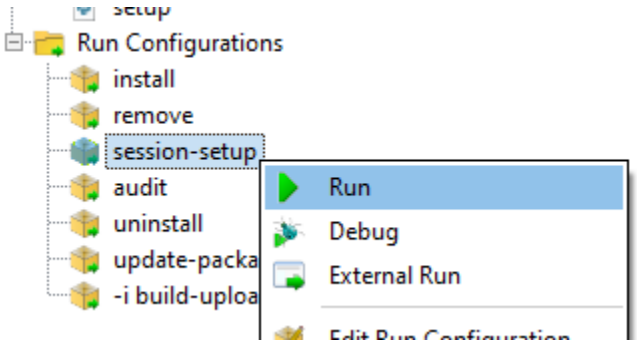


Figure1: Pyscripter - Effectuer un session-setup

46.2.5 Utiliser les fonctions d'audit pour la conformité

Note: Cette fonctionnalité est disponible dans la version **Entreprise**.

L'audit permet d'effectuer des vérifications régulières sur les configurations des postes et de centraliser le résultat des vérifications dans la console WAPT. Ceci permet de vérifier que votre parc est conforme à votre référentiel sur la durée.

Vous pouvez par exemple :

- Vérifier régulièrement la liste des administrateurs locaux des postes.
- Vérifier régulièrement la bonne configuration d'un logiciel critique.
- Vérifier régulièrement la présence de la bonne version d'un logiciel.
- Vérifier régulièrement les configurations de sécurité d'un poste.

La fonction **`audit`** bénéficie de toute la puissance et de l'étendue des librairies python pour atteindre une précision d'audit élevée.

Principe de fonctionnement

Les tâches d'**`audit`** s'exécutent après un **`upgrade`** puis à intervalle régulier défini par la valeur de `audit_schedule`.

Pour executer manuellement un audit vous pouvez également exécuter la commande :

```
wapt-get audit
```

Note: Par défaut, la fonction `audit` ne sera pas lancée si l'audit n'est pas nécessaire.

Pour forcer l'exécution vous pouvez exécuter la commande :

```
wapt-get audit -f
```

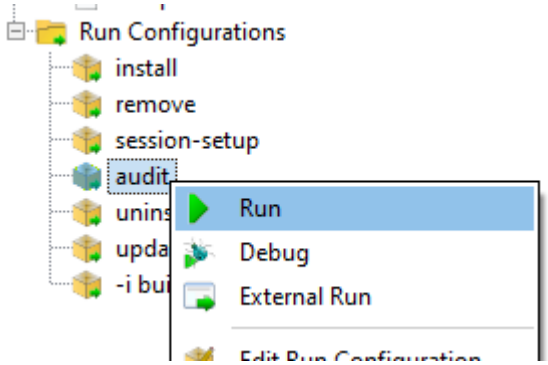
On définit la partie **`audit`** dans le fichier `setup.py` du paquet dans une fonction **`def audit()`** :

Dans cet exemple, nous améliorons le paquet firefox précédemment étudié dans cette documentation.

- Ajouter la fonction `audit` dans le fichier `setup.py`.

```
def audit():
    if isfile(r'C:\Program Files\Mozilla Firefox\distribution\policies.json'):
        print('File policies.json found')
        return "OK"
    else:
        print('File policies.json not found')
        return "ERROR"
```

- Lancer l'audit à partir de **pyscripter**.



- Tester avec le fichier puis supprimer le fichier `C:\Program Files\Mozilla Firefox\distribution\policies.json` et tester à nouveau avec **pyscripter**.

Vous pouvez voir directement l'état de l'audit dans la console (Cliquez sur le paquet puis sur la colonne audit) :

L'audit restitue une des 3 valeurs suivantes :

- **OK** ;
- **WARNING** ;
- **ERREUR**.

Attention: Avec **audit**, il n'est pas possible de faire appel à des fichiers contenus dans le paquet.

Pour utiliser des fichiers lors de l'audit il faut d'abord les copier dans un répertoire temporaire de la machine lors de l'installation du paquet.

Planifier un audit

Les tâches d'"**audit** s'exécutent après un **upgrade** puis à intervalle régulier défini par la valeur de `audit_schedule`.

La valeur est contenue dans le fichier `control` du paquet WAPT.

Par défaut, si `audit_schedule` est vide, la tâche d'audit peut être lancée manuellement depuis la console WAPT ou être lancée automatiquement si vous avez défini l'option `waptaudit_task_period` dans le `wapt-get.ini` de l'agent WAPT. Pour plus d'informations sur la dernière méthode, veuillez consulter *cette documentation*.

Sinon la valeur peut être indiquée de plusieurs manières :

- Un entier (en minutes).
- Un entier suivi d'une lettre (m = minutes , h = heure , d = jour , w = semaine).

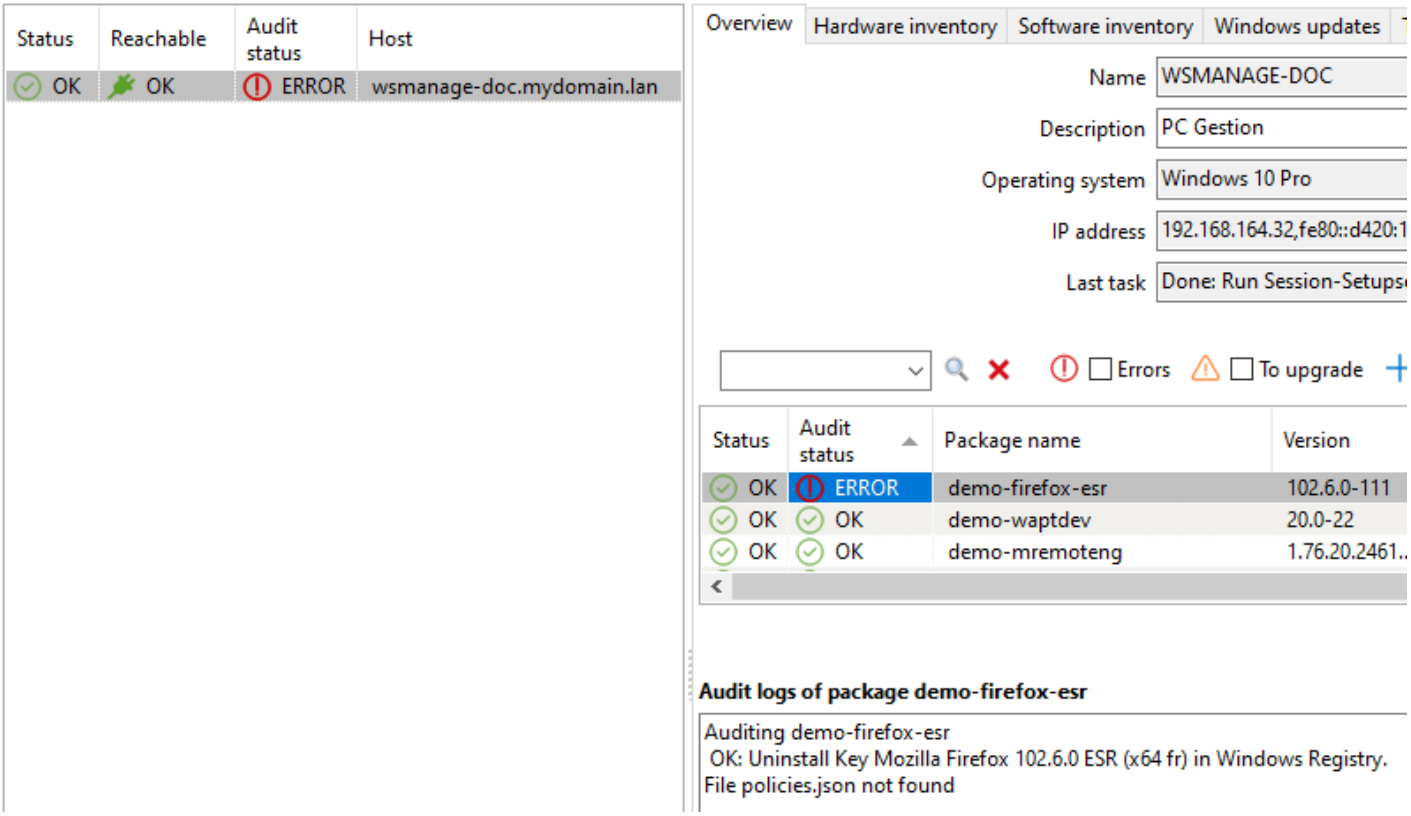


Figure2: Vérifier l'état d'un audit dans la console WAPT

Comportement par défaut de la fonction d'audit

Par défault, si aucun audit n'est déclaré, l'agent WAPT vérifiera la présence des *Uninstallkey* dans le paquet WAPT.

De cette manière WAPT vérifie que le logiciel est toujours présent.

46.2.6 Utiliser les fonctions d'audit pour la conformité

Note: Cette fonctionnalité est disponible dans la version **Entreprise**.

La fonction `audit_data` permet de faire des vérifications régulières des configurations de l'ordinateur et de centraliser les résultats de ces vérifications dans la console WAPT. Il y a une historisation et vous pouvez crypter vos données et les décrypter avec votre certificat WAPT.

Vous pouvez par exemple :

- Modifier un mot de passe d'administrateur, chiffrer des informations et les afficher dans la Console WAPT.
- Vérifier régulièrement les modifications dont votre ordinateur a besoin comme l'inventaire CVE ou l'inventaire GLPI.
- Vérifier régulièrement les configurations de sécurité d'un poste et les historiser.

  La fonction `audit_data` est utilisable uniquement dans la fonction `audit`.

Principe de fonctionnement

Les fonctions `audit_data` sont lancées si elles sont définies dans la section `def audit()` du fichier `setup.py`.

Du côté serveur, les données d'audit sont stockées dans la table HostAuditData. Le contenu de cette table peut être interrogé à l'aide de l'onglet *Reporting* de la console WAPT. Les données sont automatiquement purgées en fonction de leur date d'expiration. Lorsque le `update_status()` est lancé, les données d'audit les plus récentes sont envoyées au serveur WAPT.

Du côté client, les données d'audit sont stockées dans la base de données avec une date d'expiration (date_expiration) et le nombre maximum (max_count) des données stockées est défini dans le code.

Dans cet exemple, nous vérifions l'IP publique de l'ordinateur.

- Ajouter la fonction `audit_data` à l'intérieur de la fonction `audit` dans le `setup.py`.

```
def audit():
  ip = wgets('https://api.ipify.org',verify_cert=False)
  print(f'My public IP address is: {ip}')
  WAPT.write_audit_data_if_changed('Public IP','log for %s' % get_computername(),ip,max_
↪count=5)
  return 'OK'
```

Voici les fonctions liées au `audit_data` :

```
def write_audit_data_if_changed(self, section, key, value, ptype=None, value_date=None,␣
↪expiration_date=None, max_count=2, keep_days=None):
"""Write data only if different from last one
"""
def write_audit_data(self, section, key, value, ptype=None, value_date=None, expiration_
↪date=None, max_count=2, keep_days=None):
```

(suite sur la page suivante)

(suite de la page précédente)

```
"""Stores in database a metrics, removes expired ones

    Args:
        section (str)
        key (str)
        value (any)
        value_date
        expiration_date (str) : expiration date of the new value
        max_count (int) : keep at most max_count value. remove oldest one.
        keep_days (int) : set the expiration date to now + keep_days days. override␣
↪expiration_date arg if not None

    Returns:
        None
"""
def read_audit_data(self, section, key, default=None, ptype=None):
    """Retrieve the latest value associated with section/key from database"""

def read_audit_data_set(self, section, key):
    """Retrieve all the values associated with section/key from database"""

def delete_audit_data(self, section, key):

def read_audit_data_since(self, last_query_date=None):
    """Retrieve all the values since a date from database"""
```

46.2.7 Automatiser la mise à jour d'un paquet logiciel

Note: Cette partie de la documentation est déconseillée aux utilisateurs qui débutent avec WAPT.

Les fonctions `update_package` sont très pratiques, elles permettent de gagner beaucoup de temps lorsqu'il faut mettre à jour un paquet WAPT avec la version la plus récente d'un logiciel.

Principe de fonctionnement

La fonction *update_package* paquet ira :

- Récupérer la dernière version du logiciel en ligne.
- Télécharger la dernière version du binaire.
- Supprimer les anciennes version des binaires.
- Mettre à jour le numéro de version dans le fichier `control`.

Si votre fonction *install* se base sur la version du fichier `control` pour l'installation, alors vous n'avez pas besoin de modifier votre `setup.py`.

Il vous reste maintenant à tester l'installation avant de lancer un **`build-upload`**.

Exemple

Voici l'*update_package* de **firefox-esr** comme exemple :

```
def update_package():
      import re,requests,glob

      #Retrieving the last file name
      url = requests.head('https://download.mozilla.org/?product=firefox-esr-latest&os=win64',
↪proxies={}).headers['Location']
      filename = url.rsplit('/',1)[1].replace('%20',' ')

      #download of it if is not in the package
      if not isfile(filename):
          print('Downloading %s from %s'%(filename,url))
          wget(url,filename)

      #removing old exe with wrong name
      for fn in glob.glob('*.exe'):
          if fn != filename:
              remove_file(fn)

      # updates control version from filename, increment package version.
      control.version = '%s-0'%(re.findall('Firefox Setup (.*)esr\.exe',filename)[0])
      control.save_control_to_wapt()
```

Vous pouvez lancer le *update_package* dans **PyScripter** :

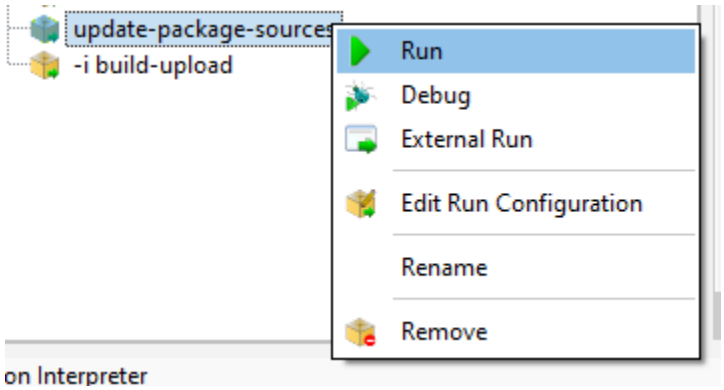


Figure3: Pyscripter - Exécution d'un update-package-source

Vous trouverez de nombreux exemples d'*update_package* qui vous inspireront dans les paquets du store de Tranquil IT.

46.2.8 Exemple: déployer un logiciel portable avec WAPT

Un bon exemple de paquet applicatif WAPT est celui d'un logiciel dit *portable* :

- Créer le répertoire d'installation dans `C:\Program Files (x86)`.
- Copier l'application dans le dossier.
- Créer un raccourci sur le bureau de l'utilisateur.
- Gérer la désinstallation de l'application portable.
- Fermer l'application si elle est en cours d'exécution.

Exemple avec ADWCleaner

Tout d'abord, télécharger Adwcleaner : https://downloads.malwarebytes.com/file/adwcleaner.

Vous pouvez générer votre modèle de paquet, veuillez vous référer à la *documentation pour la création de paquets à partir de la console WAPT*.

Le fichier `C:\waptdev\tis-adwcleaner-wapt` est créé.

Vous trouverez ici un exemple de paquet portable qui prend presque toutes les fonctions WAPT d'un `setup.py` :

```
from setuphelpers import *

uninstallkey = []

exe_name = 'AdwCleaner.exe'
path_adw = makepath(programfiles,'AdwCleaner')
path_exe = makepath(path_adw,exe_name)
nameshortcut = 'AdwCleaner'

def install():
    mkdirs(path_adw)
    filecopyto(exe_name,path_exe)
    create_desktop_shortcut(nameshortcut,path_exe)

def uninstall():
    remove_tree(path_adw)
    remove_desktop_shortcut(nameshortcut,path_exe)

def audit():
    if not isfile(path_exe):
        print('File not found')
        return "OK"
    else:
        print('File Found')
        return "ERROR"

def update_package():
    wget('https://downloads.malwarebytes.com/file/AdwCleaner',exe_name)
    control.version = get_file_properties(exe_name)['FileVersion'] + '-0'
    control.save_control_to_wapt()
```

46.2.9 Créer des paquets WAPT de mises à jour Windows avec des .msu

Indication: Pré-requis : pour construire des paquets WAPT, *l'environnement de développement WAPT doit être installé* ;

Entre les sorties de *Patch Tuesday*, Microsoft peut publier des KB supplémentaires ou des mises à jour critiques qui devront être rapidement poussées sur les machines.

À cette fin, WAPT fournit un modèle de paquet pour les fichiers `.msu`.

Dans cet exemple, nous utilisons la KB4522355 téléchargée du site officiel Microsoft.

- télécharger le paquet MSU KB4522355 depuis le Catalogue du site de Microsoft .
- Créer un modèle de paquet WAPT à partir du fichier `.msu` téléchargé. Dans la console WAPT, cliquez sur *Outils* → *Générer un modèle de paquet*.

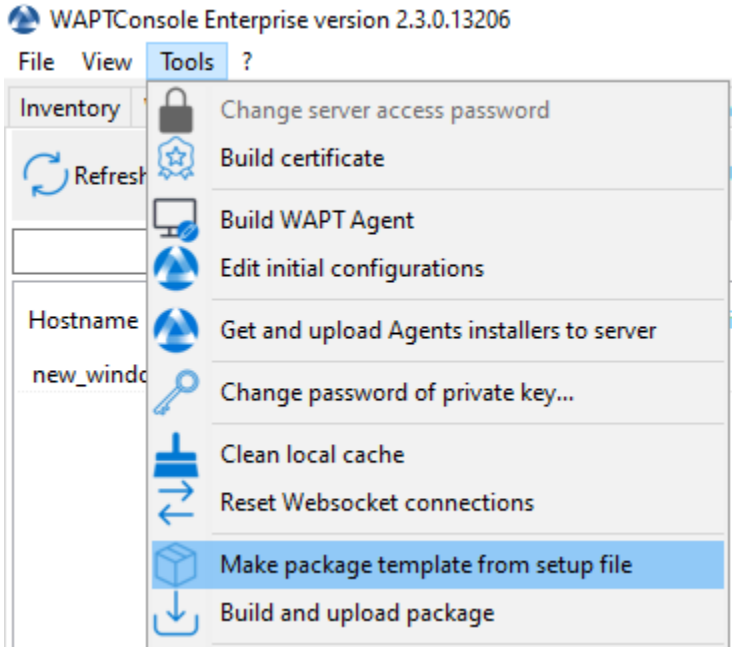


Figure4: PyScripter - Menu pour la création de modèle de paquets depuis la console

- Sélectionner le paquet téléchargé `.msu` et remplir les champs obligatoires.
- Cliquer sur *Créer et éditer* (recommandé) pour lancer la personnalisation du paquet.
- L'IDE du paquet WAPT est lancé en utilisant le code source du modèle prédéfini `.msu`.
- Comme d'habitude avec les paquets WAPT, tester, puis construire, puis signer, puis télécharger et enfin affecter les paquets WAPT souhaités à vos hôtes sélectionnés et c'est fait !
- Si le KB est groupé avec le *Patch Tuesday* suivant, vous pourrez sélectionner les hôtes sur lesquels le paquet a été appliqué et oublier le paquet KB sur les hôtes.

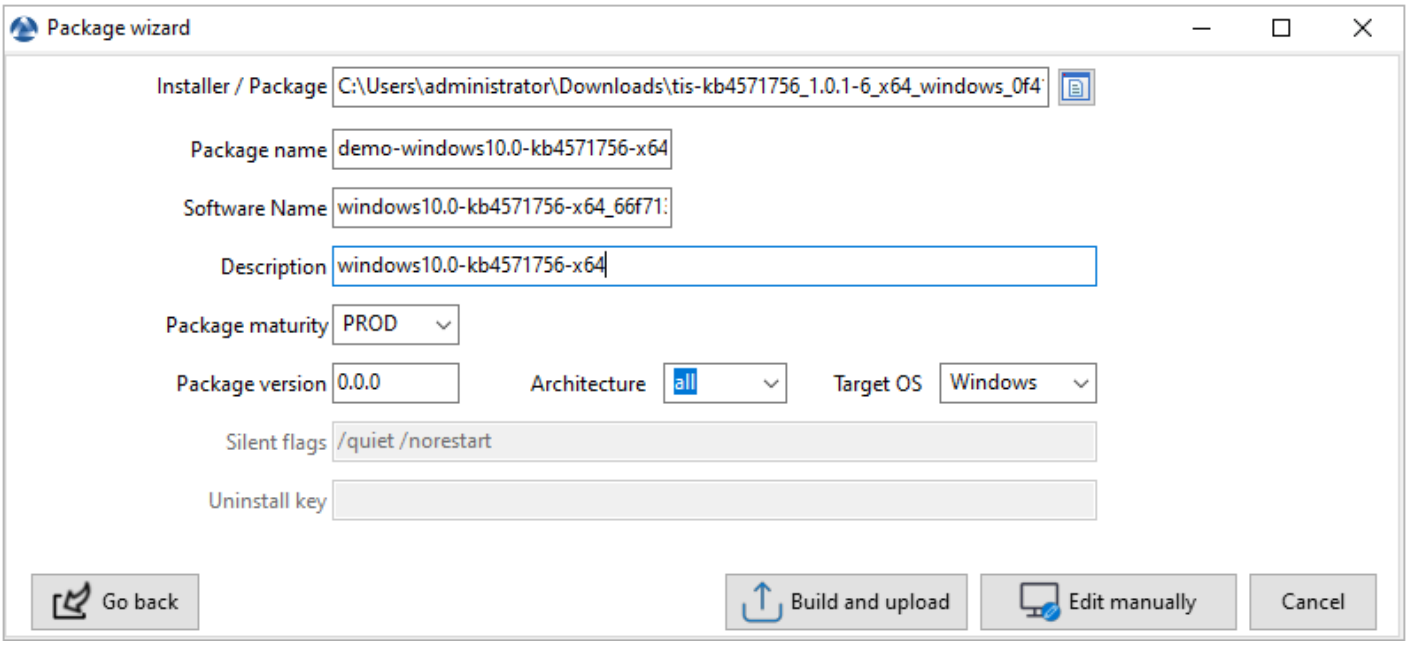


Figure5: Informations requises pour la création du paquet MSU

46.2.10 Packager des paquets linux simples

Avant de commencer, nous supposons plusieurs conditions :

- Vous disposez d'une interface graphique sur votre système Linux que vous utilisez pour développer et tester des paquets.
- Vous avez installé le paquet **vscode** à partir du dépôt de Tranquil IT.
- Votre utilisateur s'appelle *linuxuser* et est membre du groupe *sudoers*.

Créer un modèle de paquet base depuis votre poste linux

- Démarrer un utilitaire de ligne de commande.
- En tant que *linuxuser*, créer un modèle de paquet WAPT.

```
wapt-get make-template <template_name>
```

> **Avertissement:** **Ne pas lancer cette commande en tant que root ou avec sudo.**

Lorsque vous créez un modèle, il y aura plusieurs fichiers dans le dossier `.vscode` à l'intérieur du dossier de développement de paquets :

- `settings.json` ;
- `launch.json`.

Exemple avec **VLC** :

```
wapt-get make-template "tis-vlc"

Using config file: /opt/wapt/wapt-get.ini
Template created. You can build the WAPT package by launching
/opt/wapt//wapt-get.py build-package /home/linuxuser/waptdev/tis-vlc-wapt
You can build and upload the WAPT package by launching
/opt/wapt//wapt-get.py build-upload /home/linuxuser/waptdev/tis-vlc-wapt
```

Indication: Tous les paquets sont stockés dans le répertoire personnel de linuxuser (le home de l'utilisateur acutellement connecté).

- VSCode se charge et ouvre le projet de paquet.

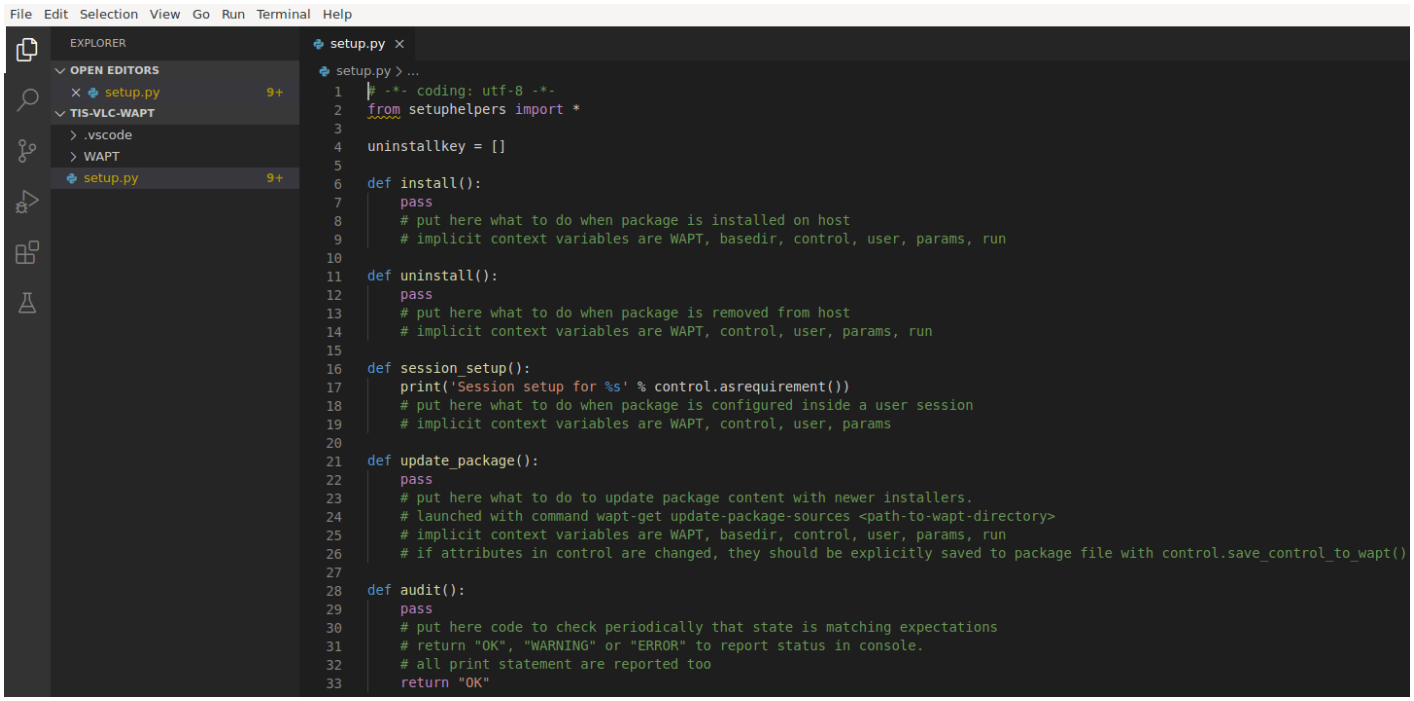


Figure6: Ouverture du VSCode avec le focus sur le fichier *setup*

- Vérifier le contenu du fichier `control`.

  Vous devez donner une `description` à votre paquet, et renseigner le `target_os` et la `version` du paquet.

Indication: `os_target` pour Unix est *linux*.

Avertissement: Le numéro de `version` du logiciel dans votre fichier `control` **DOIT** commencer à 0, et non le numéro de version du titre du logiciel, car le numéro de version peut ne pas être le même que celui affiché dans le dépôt DEB / YUM.

– Fichier `control` d'origine.

```
package            : tis-vlc
version            : 0-0
architecture       : all
section            : base
priority           : optional
maintainer         : user
description        : automatic package for vlc
```

– Fichier `control` modifié.

```
package            : tis-vlc
version            : 0
architecture       : all
section            : base
priority           : optional
maintainer         : Tranquil-IT Systems
description        : VLC for linux
target_os          : linux
min_wapt_version   : 1.8
```

Note: Il est à noter qu'une sous-version *-1* a été ajoutée. Il s'agit de la version de packaging du paquet WAPT.

Il permet au développeur de paquets de publier plusieurs versions de paquets WAPT d'un même logiciel, ce qui est très utile pour un développement très rapide et itératif.

- Changer le code du fichier `setup.py` en conséquence.

```
:emphasize-lines: 8
# -*- coding: utf-8 -*-
from setuphelpers import *

uninstallkey = []

def install():
    apt_install('vlc')
```

- Enregistrer le paquet.

Gérer la désinstallation

- Modifier le fichier `setup.py` avec une procédure de désinstallation.

```
def uninstall():
apt_remove('vlc')
```

- Lancer un *remove* de VSCode *Run Configurations*.
- Vérifier que le logiciel a été correctement supprimé.

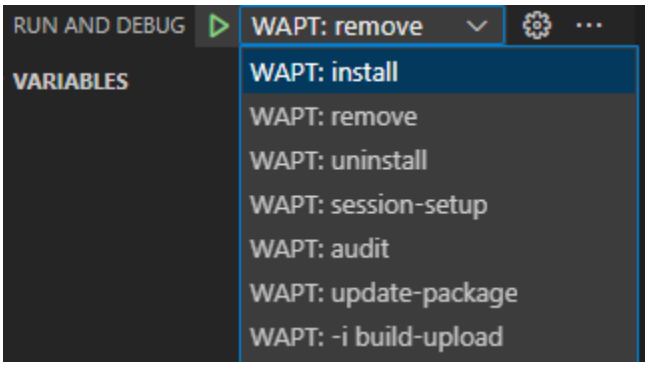


Figure7: À l'issue de la désinstallation, le programme est désinstallé

```
dpkg -l | grep vlc
```

Indication: Dans la fonction **`uninstall()`**, on ne peut pas appeler des fichiers contenus dans le paquet WAPT. Pour les appeler, il faudra avoir copié les fichiers dans un répertoire local de la machine lors de l'installation du paquet.

Gérer le session-setup

- Modifier le fichier `setup.py` avec un `session-setup` ;

  Dans cet exemple, nous allons créer un fichier :fichier : *vlcrc* par défaut dans le profil de l'utilisateur.

```
def session_setup():
  vlcrc_content="""[qt] # Qt interface
qt-notification=0
qt-privacy-ask=0
metadata-network-access=0
"""

  vlcdir = os.path.join(os.environ['HOME'], '.config', 'vlc')
  path_vlrc = makepath(vlcdir,'vlcrc')
  ensure_dir(vlcdir)
  if not isfile(path_vlrc):
      with open(makepath(vlcdir,'vlcrc')) as f:
          f.write(vlcrc_content)
```

- Lancez un *session-setup* à partir de VSCode *Run Configurations*.

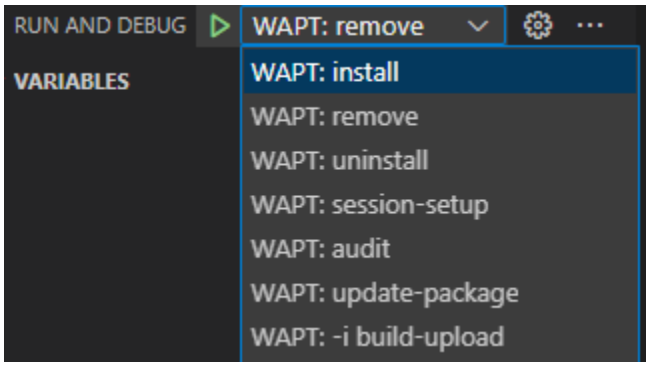


Figure8: À l'issue de la désinstallation, le programme est désinstallé

Construire et téléverser le paquet

Vous trouverez votre paquet dans le répertoire `~/waptdev`.

Vous devez transférer le dossier du paquet sur la machine Windows qui possède la clé privée.

Ensuite, se référer à la *documentation pour la construire et charger le paquet depuis la console WAPT*.

46.2.11 Chiffrer des données sensibles contenues dans un paquet WAPT

Note: Cette partie de la documentation est déconseillée aux utilisateurs qui débutent avec WAPT.

Cette fonctionnalité est disponible uniquement dans la version **Entreprise**.

Quel est l'intérêt de faire cela ?

Dans le fonctionnement de WAPT, l'intégrité du paquet est assurée. Un paquet dont le contenu a été modifié sans avoir été re-signé sera systématiquement refusé par le client WAPT.

En revanche le contenu d'un paquet WAPT n'est pas chiffré et sera lisible de tous. Ce modèle technique de transparence apporte cependant de nombreux bénéfices.

Cela peut être génant dans le cas d'un paquet qui contiendrait un mot de passe, une clé de licence, ou une donnée sensible.

Heureusement, **nous avons une solution**!

Principe de fonctionnement

Lorsque qu'un agent WAPT s'enregistre auprès du serveur WAPT, il génère un couple clé privée / certificat public dans `C:\Program Files (x86)\wapt\private`.

- Le certificat est envoyé au serveur avec l'inventaire lors de l'enregistrement initial du client WAPT.
- La clé privée est conservée par l'agent et n'est accessible en lecture que par les *Administrateurs Locaux*.

Nous allons donc chiffrer la donnée sensible contenue dans le paquet avec le certificat appartenant à la machine.

Lors de l'installation l'agent WAPT pourra ainsi déchiffrer la donnée sensible grâce à sa clé privé.

Avec ce mode de fonctionnement le serveur WAPT et les dépôts secondaires n'ont pas connaissance de la donnée sensible.

Cas pratique

Vous trouverez ici un exemple de paquet WAPT ou nous chiffrons un texte dans dans une fonction **`update_package`** puis nous déchiffrons ce texte dans la partie **`install`**.

Dans cet exemple, la fonction **`update_package`** nous permet de parcourir la base de donnée du serveur WAPT pour récupérer le certificat de chaque machine pour ensuite chiffrer le texte sensible avec celui-ci.

Le texte chiffré pour chaque machine est ensuite stocké dans un fichier `encrypt-txt.json` à la racine du paquet.

Lors de l'installation du paquet, l'agent WAPT prendra le texte chiffré et le déchiffrera avec sa clé privé.

Vous pouvez le tester par vous-même en téléchargeant le packaging d'exemple *tis-encrypt-sample* (https://store.wapt.fr/store/tis-encrypt-sample)

> **Attention:** La sortie python (log install du paquet) est accessible en lecture aux utilisateurs de la machine, **vous ne devez donc pas afficher le texte déchiffré avec un print lors de l'installation**.

CHAPTER 47

Utiliser des IDE différents pour développer les paquet WAPT

47.1 Configuring WAPT to use supported IDEs

Si vous êtes habitué(e) à travailler avec un autre *IDE*, vous pouvez être soulagé maintenant car WAPT supporte d'autres éditeurs de développement intégrés.

Note: Utiliser un IDE supporté lancera le projet de paquet WAPT avec une configuration de débogage valide.

47.1.1 Sur Windows

Table1: Éditeurs de texte supportés en natif dans WAPT sous Windows

Récupérer le nom du logiciel	Logo de l'éditeur de texte
`PyScripter`	
`Visual Studio Code`	
`Visual Studio Codium`	

Pour configurer un autre éditeur pour WAPT, vous devez modifier l'attribut `editor_for_packages` dans la section `[global]` du fichier de configuration `%LOCALAPPDATA%\waptconsole\waptconsole.ini` de votre console WAPT.

```
[global]
...
editor_for_packages = vscode
```

47.1.2 Sur Linux / macOS

Table2: Éditeurs de texte supportés en natif dans WAPT sous Windows

Récupérer le nom du logiciel	Logo de l'éditeur de texte
Visual Studio Code	
Visual Studio Codium	
Nano	
Vim	

Pour configurer un autre éditeur pour WAPT, vous devez modifier l'attribut `editor_for_packages` dans la section `[global]` du fichier de configuration `/opt/wapt/wapt-get.ini` de votre agent WAPT.

Par défaut, si l'attribut `editor_for_packages` est vide, le WAPT essaiera de lancer (dans cet ordre) :

- **vscodium** ;
- **vscode**;
- **nano** ;
- **vim** ;
- **vi**.

```
[global]
...
editor_for_packages = vim
```

47.2 Configurer WAPT pour utiliser un éditeur de code personnalisé

Windows

```
[global]
...
editor_for_packages = C:\Program Files\Notepad++\notepad++.exe {setup_filename}
```

Linux/ macOS

```
[global]
...
editor_for_packages = /opt/pycharm/bin/pycharm_x64 {wapt_sources_dir}
```

47.2.1 Arguments personnalisés

Table3: Les arguments peuvent être passés dans la commande `editor_for_packages`

Argument	Description
`{setup_filename}`	Lance l'éditeur de code personnalisé et édite le fichier WAPT `setup.py`.
`{control_filename}`	Lance l'éditeur de code personnalisé et modifie le fichier `control` des paquets WAPT.
`{wapt_sources_dir}`	Lance l'éditeur de code personnalisé et ouvre le dossier du paquet WAPT.
`{wapt_base_dir}`	Lance l'éditeur de code personnalisé et ouvre le dossier d'installation WAPT.

CHAPTER 48

Mise à jour des packages WAPT de Python 2 à Python 3

Attention: Avec WAPT 2.0, le fonctionnement interne de WAPT est passé à python3. Les paquets WAPT doivent aussi suivre la nouvelle syntaxe python3.

Table1: Les principales différences de syntaxe

Syntaxe	Python 2	Python 3
print	`print'Hello'`	`print('Hello')`
unicode string	`ur`	`r`
opérateurs	`<> <=> !=`	`!=`
Acces à la base de registre Windows	`_winreg`	`winreg`

Indication: Pour plus de détails sur les changements, consultez :

- https://python-future.org/compatible_idioms.html.
- https://blog.couchbase.com/tips-and-tricks-for-upgrading-from-python-2-to-python-3/.

CHAPTER 49

Structure d'un paquet WAPT

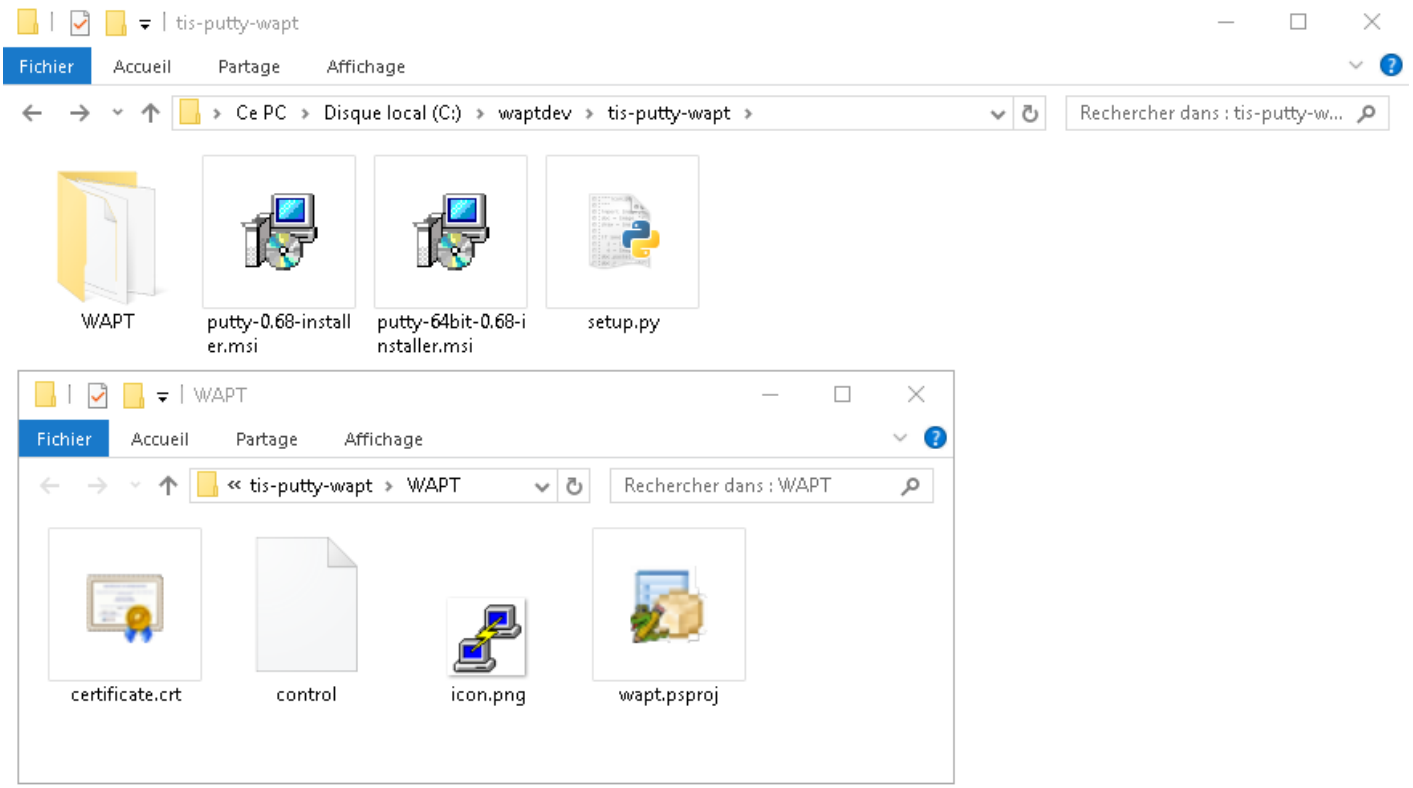


Figure1: Structure du packaging WAPT affichée dans l'explorateur Windows

Un package WAPT est un fichier: mimetype: *.zip* contenant plusieurs éléments:

- Un fichier `setup.py` à la racine.
- Un fichier ou plusieurs fichiers binaires.
- Un fichier ou plusieurs autres fichiers additionnels.
- Un fichier `control` dans le dossier `WAPT`.
- Un fichier `icon.png` dans le dossier `WAPT`.
- Un fichier `certificate.crt` dans le dossier `WAPT`.
- Un fichier `manifest.sha256` dans le dossier `WAPT`.
- Un fichier `signature.sha256` dans le dossier `WAPT`.
- Un fichier `wapt.psproj` dans le dossier `WAPT`, ce fichier est utilisé pour stocker les données de configuration **`PyScripter`** pour le paquet WAPT.
- Depuis WAPT 1.8, un dossier caché `.vscode` qui contient un fichier `launch.json` et un fichier `settings.json` utilisés pour stocker les données de configuration **`VScode`** pour le paquet WAPT.

49.1 Le fichier *control*

Le fichier `control` est la fiche d'identité du paquet.

```
package           : tis-firefox-esr
version           : 62.0-0
architecture      : all
section           : base
priority          : optional
maintainer        : Administrateur
description       : Firefox Web Browser French
description_fr    : Navigateur Web Firefox Français
description_es    : Firefox Web Browser
depends           :
conflicts         :
maturity          : PROD
locale            : fr
target_os         : windows
min_os_version    :
max_os_version    :
min_wapt_version  : 1.6.2
sources           :
installed_size    :
impacted_process  : firefox.exe
audit_schedule    :
editor            : Mozilla
keywords          : Navigateur
licence           : MPL
homepage          : https://www.mozilla.org/en-US/firefox/organizations/
package_uuid      : dc66ccd1-d987-482e-b792-04e89a3803f7
valid_from        : 2022-02-23T00:00:00
```

(suite sur la page suivante)

(suite de la page précédente)

```
valid_until       : 2022-03-23T00:00:00
forced_install_on : 2022-03-23T00:00:00
signer            : Tranquil IT
signer_fingerprint: 459934db53fd804bbb1dee79412a46b7d94b638737b03a0d73fc4907b994da5d
signature         : MLOzLiz0qCHN5fChdylnvXUZ8xNJj4rEu5FAAsDTdEtQ(...)hsduxGRJpN1wLEjGRaMLBlod/p8w==
signature_date    : 20170704-164552
signed_attributes : package,version,architecture,section,priority,maintainer,description,depends,
↪conflicts,maturity,locale,min_os_version,max_os_version,min_wapt_version,sources,installed_size,
↪signer,signer_fingerprint,signature_date,signed_attributes
```

Table1: Description des options du fichier control

Paramètre	Description	Valeur Exemple
`package`	**Nom du paquet WAPT, sans accent, sans espace, sans caractère spécial, sans majuscule**.	tis-geogebra
`version`	Définit la version du paquet (note : la version **NE DOIT PAS** contenir plus de 5 délimiteurs, le dernier chiffre étant le numéro de version du paquet). La version **Doit**commencer par la version du logiciel packagé (**chiffres seulement**) divisée par des points (`.`) et **Doit**se terminer par la version du packaging WAPT séparée par un caractère tiret (`-`).	5.0.309.0-1
`architecture`	Définit l'architecture du processeur sur lequel le packaging WAPT sera installé. Un paquet prévu pour une architecture x64 ne sera pas visible avec un agent WAPT installé sur un poste équipé d'un processeur x86. Valeurs possibles : • **x86** : le paquet est destiné uniquement aux machines avec un processeur 32bits ; • **x64** : le paquet est destiné uniquement aux machines avec un processeur 64bits ; • **all** : le paquet est conçu pour n'importe quelle architecture.	x64
`section`	Type de paquet (`host`, `group`, `base`) Valeurs possibles : • **host** : paquet machine ; • **group** : paquet groupe ; • **base** : paquet logiciel ; • **unit** : paquet UO ;	base
`priority`	Niveau de priorité d'installation du paquet (optionnel pour le moment) Cette option n'est pas prise en en charge pour le moment, ce champ permettra à terme de définir la priorité d'installation d'un paquet WAPT.	Non pris en charge pour le moment
`maintainer`	Indique la maturité du paquet. Indiquer l'adresse email peut être utile.	Arnold Schwarzenegger <terminator@mydomain.lan>

suite sur la page suivante

Table 1 – suite de la page précédente

Paramètre	Description	Valeur Exemple
`description`	Description du paquet qui apparaîtra dans la console et dans le self-service. Ajouter un champ description_fr ou description_es permet par exemple de préciser une description pour une langue précise. Si la langue n'existe pas, l'agent wapt utilisera la champ description classique.	The Graphing Calculator for Functions, Geometry, Algebra, Calculus, Statistics and 3D
`description_fr`	Description du paquet dans une langue précise.	Calculatrice graphique
`depends`	Définit quelle(s) dépendance(s) doit(doivent) être satisfaite(s) avant d'installer le paquet WAPT, par exemple *tis-java* devra être installé avant le paquet *LibreOffice*. On peut définir plusieurs dépendances en les séparant par des virgules (,)	tis-java
`conflicts`	Définit les packages WAPT qui **Doivent** être **supprimés** avant l'installation du package, par exemple *tis-firefox* **Doivent** être supprimés avant l'installation du package *tis-firefox-esr*, ou *OpenOffice* **Doivent** être supprimés avant l'installation de *LibreOffice*. Fonctionne exactement à l'inverse de *depends*. On peut définir plusieurs conflits en les séparant par des virgules (,).	tis-graph
`maturity`	Définit le niveau de maturité du paquet WAPT (`BETA`, `DEV`, `PROD`, etc). Un agent verra par défaut les paquets *PROD* et les paquet sans maturité. Pour qu'un client puisse voir un paquet d'un autre niveau de maturité, il faudra définir l'attribut `maturities` dans le fichier de configuration `wapt-get.ini` de l'agent WAPT.	PROD
`locale`	Définit la langue pour le paquet WAPT. Un agent WAPT verra par défaut les paquets qui sont configurés pour son (ses) environnement(s) linguistique(s) et les paquets sans langue spécifiée. Pour qu'un ordinateur puisse voir un paquet dans une autre langue, vous devrez configurer les *locales* dans `wapt-get.ini` de l'agent WAPT. La casse et l'ordre n'ont pas d'importance. Si vous voulez respecter un ordre pour les maturités, vous devrez définir l'ordre dans le fichier de configuration `wapt-get-ini` de l'agent WAPT.	fr,en,es
`target_os`	Indique le système d'exploitation prévu pour le paquet. Un agent verra par défault les paquets prévus pour son système d'exploitation et les paquets avec un champ `target_os` sans valeur. Depuis la version 2.3 l'attribut `target_os` peut avoir plusieurs valeurs, il doit correspondre à la balise `host_capabilities`. Avec debian, vous pouvez utiliser `<` ou `>`. Les trois premières dans l'exemple sont les plus utilisées.	windows, mac, linux, debian-bullseye, redhat_based, centos8, debian(>8), ubuntu, almalinux8

suite sur la page suivante

Table 1 – suite de la page précédente

Paramètre	Description	Valeur Exemple
`min_os_version`	Définit la version minimale de Windows pour que le paquet soit vu par l'agent WAPT. Pour un `target_os` *windows*, ce champ définit la version minimale du système d'exploitation. Par exemple, cet attribut peut être utilisé pour éviter d'installer sur WindowsXP des paquets qui ne fonctionnent que sur Windows7 et supérieur. Depuis la version 1.8, il peut également définir la version minimale de Mac OS. Nous conseillons de ne pas l'utiliser avec Linux car il existe plusieurs distributions différentes.	6.0
`max_os_version`	Définit la version maximale de Windows pour que le paquet soit vu par l'agent WAPT. Pour un *windows target_os*, il définit le maximum Windows Operating System Version. Par exemple, cet attribut peut être utilisé pour installer sur Windows7 des versions plus récentes d'un logiciel qui ne sont plus supportées par Windows XP. Depuis la version 1.8, il peut également définir la version minimale de Mac OS. Nous conseillons de ne pas l'utiliser avec Linux car il existe plusieurs distributions différentes.	10.0
`min_wapt_version`	Définit la version minimale de l'agent WAPT pour que le paquet WAPT fonctionne correctement. Le code de WAPT évoluant, certaines fonctions que vous aurez utilisées dans vos anciens paquets peuvent devenir obsolètes avec les nouvelles versions des agents WAPT.	1.3.8
`sources`	Définit le lien de stockage des versions historisées du paquet (commande **`wapt-get source`**). Définit un dépôt pour le versionnement des paquets WAPT, par exemple https://svn.mydomain.lan/sources/tis-geogebra-wapt/trunk/. Cela permet de versionner le paquet et de concevoir collaborativement le paquet. Le versionnage de paquets est particulièrement utile lorsque plusieurs personnes créent des paquets de manière collaborative. Cette fonction est également utile pour retracer l'historique d'un paquet si vous êtes soumis à une réglementation particulière dans votre secteur d'activité.	https://srv-svn.mydomain.lan/sources/tis-geogebra-wapt/trunk/
`installed_size`	Définit l'espace disque libre minimum requis pour installer le paquet WAPT. Le test d'espace disque disponible est fait sur le dossier `C:Program Files`. La valeur renseignée dans *installed_size* doit être en bytes. Pour convertir des valeurs de stockage en bytes, visiter https://www.convertworld.com/fr/mesures-informatiques/.	254251008

suite sur la page suivante

Table 1 – suite de la page précédente

Paramètre	Description	Valeur Exemple
impacted_process	Indique une liste de processus impactés lors de l'installation du paquet. Ce champ est utilisé par les fonctions **install_msi_if_needed** et **install_exe_if_needed** si *killbefore* n'est pas renseigné. *impacted_process* est également utilisé lors de la désinstallation d'un paquet. Cela permet de fermer l'application si l'application est ouverture avant sa désinstallation.	firefox.exe
audit_schedule	Définit la périodicité d'exécution de la fonction d'audit dans le paquet WAPT. La valeur peut être indiquée de plusieurs manières : • un entier (en minutes); • un entier suivi d'une lettre (*m* = minutes , *h* = heure , *d* = jour , *w* = semaine).	60
editor	Éditeur de logiciels des binaires intégrés dans le paquet *base* de WAPT. La liste des valeurs pourra être utilisée dans la console et dans le selfservice pour trier les paquets.	Mozilla
license	Indique la page d'accueil du site officiel de logiciel integrée dans le paquet. La liste des valeurs pourra être utilisée dans la console et dans le selfservice pour trier les paquets.	GPLV3
keywords	Définit un ensemble de mots-clés décrivant le paquet WAPT. La liste des valeurs pourra être utilisée dans la console et dans le selfservice pour trier les paquets.	Bureautique,Editeur,calcul
homepage	Indique la page d'accueil du site officiel de logiciel integrée dans le paquet. La liste des valeurs pourra être utilisée dans la console et dans le selfservice pour trier les paquets.	https://www.tranquil.it/
package_uuid	Identifiant unique du packaging. Il est généré automatiquement lors de la construction du packaging.	dc66ccd1-d987-482e-b792-04e89a3803f7
valid_from	Date / heure après laquelle le paquet peut être installé. L'agent WAPT refusera de l'installer avant cette date. La chaîne est formatée selon la norme ISO8601 : YYYY-MM-DDTHH:MM:SS. Lorsque la date est dépassée, WAPT installera le paquet lorsqu'une mise à jour sera déclenchée.	2022-02-23T00:00:00
valid_until	Date / heure après laquelle le packaging peut être installé. L'agent refusera de l'installer avant cette date. La chaîne est formatée selon la norme ISO8601 : YYYY-MM-DDTHH:MM:SS.	2022-02-23T00:00:00
forced_install_on	Date / heure après laquelle l'Agent WAPT déclenchera une installation forcée du paquet. La chaîne est formatée selon la norme ISO8601 : YYYY-MM-DDTHH:MM:SS.	2022-02-23T00:00:00
signer	Définit le CN du signataire du paquet WAPT. Il s'agit généralement du nom complet du signataire. La valeur est automatiquement insérée lors de la signature du packaging WAPT.	Tranquil IT

suite sur la page suivante

Table 1 – suite de la page précédente

Paramètre	Description	Valeur Exemple
`signer_fingerprint`	Fournit l'empreinte digitale de la signature du titulaire du certificat. La valeur est automatiquement insérée lors de la signature du packaging WAPT.	2BAFAF007C174A3B00F12E9CA1E749
`signature`	Fournit le hachage SHA256 du paquet WAPT. La valeur est automatiquement insérée lors de la signature du packaging WAPT.	MLOzLiz0qC(…)hsEjGRaMLBlod/p8w=
`signature_date`	Indique la date de signature du paquet. La valeur est automatiquement insérée lors de la signature du packaging WAPT.	20180307-230413
`signed_attributes`	Listes des attributs du fichier `control` du packaging WAPT qui sont signés. La valeur est automatiquement insérée lors de la signature du packaging WAPT.	package, version, architecture, section, priority, maintainer, description, depends, conflicts, maturity, locale, min_wapt_version, sources, installed_size, signer, signer_fingerprint, signature_date, signed_attributes

Attention: Si le fichier `control` comporte des caractères accentués, le fichier doit être enregistré en format **UTF-8 (No BOM)**.

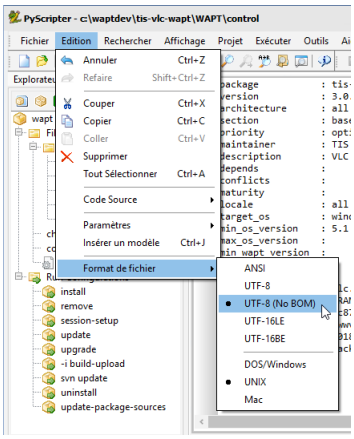


Figure2: PyScripter - UTF-8 (No BOM)

49.2 Le fichier *setup.py*

- Cette ligne se trouve au début de chaque paquet WAPT qui contient un `setup.py` :

```
from setuphelpers import *
```

Nous demandons au paquet d'importer toutes les fonctions depuis la librairie `Setuphelpers`.

Setuphelpers est la librairie intégrée à WAPT, elle embarque des fonctions simplifiées pour aider à créer des paquets fonctionnels rapidement.

- suivi d'une liste `uninstallkey` pour associer une liste de *clés de désinstallation* au packaging WAPT.

```
uninstallkey = ['tisnaps2','Mozilla Firefox 45.6.0 ESR (x86 fr)']
```

Nous déclarerons ici une liste des *uninstall keys* associées au paquet. Quand un paquet est supprimé, l'agent WAPT recherche dans la base de registre la *uninstallkey* correspondant au paquet. Cette *uninstallkey* indiquera à l'agent WAPT les actions à déclencher pour supprimer le logiciel.

Même s'il n'y a pas de *uninstallkey* pour le paquet, il faudra quand même déclarer une *uninstallkey* vide :

```
uninstallkey = []
```

- suivi de fonctions telles que `def_install()`, `def_uninstall()`, `def_session-setup()` et `def_audit()`

Ces fonctions décrivent les recettes du paquet WAPT, l'ensemble des instructions qui seront exécutées pour installer, supprimer, configurer et auditer un paquet WAPT.

49.3 Le fichier *wapt.psproj*

On trouve dans le dossier WAPT un fichier `wapt.psproj`.

Il s'agit du fichier projet **`PyScripter`** pour le paquet WAPT.

Pour éditer un paquet avec **`PyScripter`**, il suffira d'ouvrir ce fichier.

49.4 Le fichier *icon.png*

On trouve dans le dossier WAPT un fichier `icon.png`.

Il permet d'associer une icône au paquet.

Indication:

- L'icône est utilisée dans le Self-Service, elle est téléchargée avec sa somme MD5 pour la sécurité ; si la somme MD5 n'est pas bonne, l'icône est supprimée.
- L'icône **DOIT* être au format *.png* en 48px par 48px.

49.5 Le fichier *manifest.sha256*

On trouve dans le dossier WAPT un fichier `manifest.sha256`.

Il contient l'empreinte sha256 de chaque fichier du package.

49.6 Le fichier *signature*

Le fichier `signature` est situé dans le dossier WAPT.

Le fichier contient la signature du fichier `manifest.sha256`.

A l'installation du paquet WAPT, **`wapt-get`** vérifie :

- Que la signature de `manifest.sha256` correspond au fichier `manifest.sha256` actuel (l'agent vérifiera les certificats publics dans `C:\Program Files (x86)\wapt\ssl` sous Windows et `/opt/wapt/ssl` sous Linux et MacOS).
- Que l'empreinte sha256 de chaque fichier est identique à celle contenue dans le fichier `manifest.sha256`.

49.7 Le fichier *certificate.crt*

Le fichier `certificate.crt` est situé dans le dossier WAPT.

C'est le certificat du mainteneur qui a signé le paquet.

Lors de l'installation d'un paquet, **`wapt-get`** vérifie que le `certificate.crt` ou son parent correspond aux certificats présents dans `C:\Program Files (x86)wapt\ssl` sous Windows et `/opt/wapt/ssl` sous Linux et macOS. Si le certificat n'est pas validé, le paquet ne sera pas installé.

49.8 Autres fichiers

D'autres fichiers peuvent être embarqués dans le paquet WAPT. Par exemple :

- Un installeur à côté du `setup.py` à appeler dans le **`setup.py`**.
- Un fichier de réponses à passer à l'installeur du logiciel.
- Un fichier de licence.
- Etc.

CHAPTER 50

WAPT Setuphelpers apidoc

50.1 Setuphelpers for Windows

Link for Windows

50.2 Setuphelpers for Linux

Link for Linux

50.3 Setuphelpers for MacOS

Link for MacOS

CHAPTER 51

Problèmes fréquents

51.1 Réinitialiser le mot de passe du serveur WAPT Linux

Il arrive parfois de configurer un serveur WAPT et d'oublier son mot de passe.

Pour réinitialiser le mot de passe *SuperAdmin* de la console WAPT, vous devez relancer le processus de post-configuration sur le serveur WAPT :

- Se connecter au serveur avec SSH.
- Se connecter avec l'utilisateur root (ou utiliser sudo).
- Lancer le *script de post-configuration.*.

51.2 J'ai perdu ma clé privée WAPT

La sécurité et le bon fonctionnement de WAPT s'appuient sur les jeux de clés privés et de certificats publics.

La perte d'une clé privée nécessite donc de *regénérer une nouvelle clé* et les certifcats associés, et ensuite déployer sur le parc de machines les certificats pour la nouvelle clé.

Par conséquent, perdre la clé entraîne quelques problèmes, la procédure de récupération n'est pas anodine, même si elle est simple.

51.2.1 Procédure de renouvellement ou de création d'une clé privée

La procédure va être la suivante :

- *Générer une nouvelle clé privée/certificat public*. Vous conserverez alors la clé privée (fichier `.pem`) dans un endroit sûr ;
- Déployer manuellement avec une GPO ou en utilisant un role Ansible (non documenté), le nouveau certificat `.crt` sur vos clients dans le dossier **ssl**.
  - `C:\Program Files (x86)\ssl` sur Windows;
  - `/opt/wapt/ssl` sur Linux et MacOS.

51.2.2 Re-signer les paquets dans le dépot

Les paquets WAPT du dépôt local étant signés avec l'ancienne clé, il convient de re-signer l'intégralité des paquets avec la nouvelle clé :

- *Utiliser la console WAPT*, ou
- *Utiliser la ligne de commande*.

51.3 Je me suis fait voler ma clé privée

Attention: Toute la sécurité de WAPT repose sur la séquestration de cette clé privée.

WAPT ne gère pas encore la révocation des clés en utilisant une CRL.

La solution consiste à supprimer chaque certificat `.crt` associé à la clé privée volée, situé dans le dossier **ssl** :

- `C:\Program Files (x86)\ssl` sur Windows;
- `/opt/wapt/ssl` sur Linux et MacOS.

Cette opération peut être effectuée à l'aide d'une GPO, manuellement, avec un paquet WAPT ou avec un rôle Ansible (non documenté).

51.4 Mon UUID BIOS bogue

- Il arrive parfois qu'un problème survienne avec certains BIOS. WAPT utilise l'*UUID* de la machine comme identifiant pour reconnaître les machines.
- L'*UUID* BIOS est censé être unique, malheureusement chez certains constructeurs et pour certaines séries de machines, les *UUID* des BIOS sont identiques.
- Le PC remontera bien dans la console mais il écrasera le PC déjà présent considérant que l'ordinateur a changé de nom.

51.4.1 Résoudre des problèmes de UUID BIOS

WAPT permet de générer un *UUID* aléatoire pour remplacer celui indiqué dans le BIOS.

```
wapt-get generate-uuid
```

L'agent WAPT FQDN peut être utilisé à la place du UUID. Dans le fichier de configuration `wapt-get.ini`, définissez dans la section *[global]* :

```
use_fqdn_as_uuid = True
```

51.5 Mon WAPTdeploy ne fonctionne pas

L'utilitaire **waptdeploy** n'arrive pas à installer l'agent WAPT.

51.5.1 Lancer WAPTdeploy localement

Lancer **waptdeploy** localement peut mettre en évidence un problème en affichant explicitement les erreurs.

Attention: Ne pas oublier de lancer l'invite de commande en tant qu'*Administrateur*.

Exemple de commande à lancer :

```
C:\Program Files (x86)\wapt\waptdeploy.exe --
↪hash=2a9971aad083d6822b6e4d1ccfb9886be9429ec58bb13246810ff3d6a56ce887 --minversion=2.1.0.10550 --
↪wait=15 --waptsetupurl=https://srvwapt.mydomain.lan/wapt/waptagent.exe
```

Dans notre cas le hash n'est pas le bon.

WAPTdeploy fonctionne manuellement mais ne fonctionne pas via GPO

Vérifier que le port 8088 écoute correctement sur l'hôte:

```
gpresult /h gpo.html & gpo.html
```

Pour forcer l'application des GPO :

```
gpupdate /force
```

Si **waptdeploy** n'apparaît pas, il faut re-vérifier la configuration de la GPO:

- Il se peut que vous utilisiez une ancienne version de l'utilitaire WAPT Deployment, alors `téléchargez la dernière version` de l'utilitaire WAPT Deployment à partir de la page Web du serveur WAPT.
- Merci à Emmanuel EUGENE de l'institution de recherche publique française INSERM qui a soumis cette cause possible du mauvais fonctionnement de l'utilitaire de déploiement WAPT, si vous répliquez des contrôleurs de domaine, assurez-vous que les GPO sont correctement synchronisées entre vos DC et que les ACL sont appliquées de manière identique sur les `SysVols`.

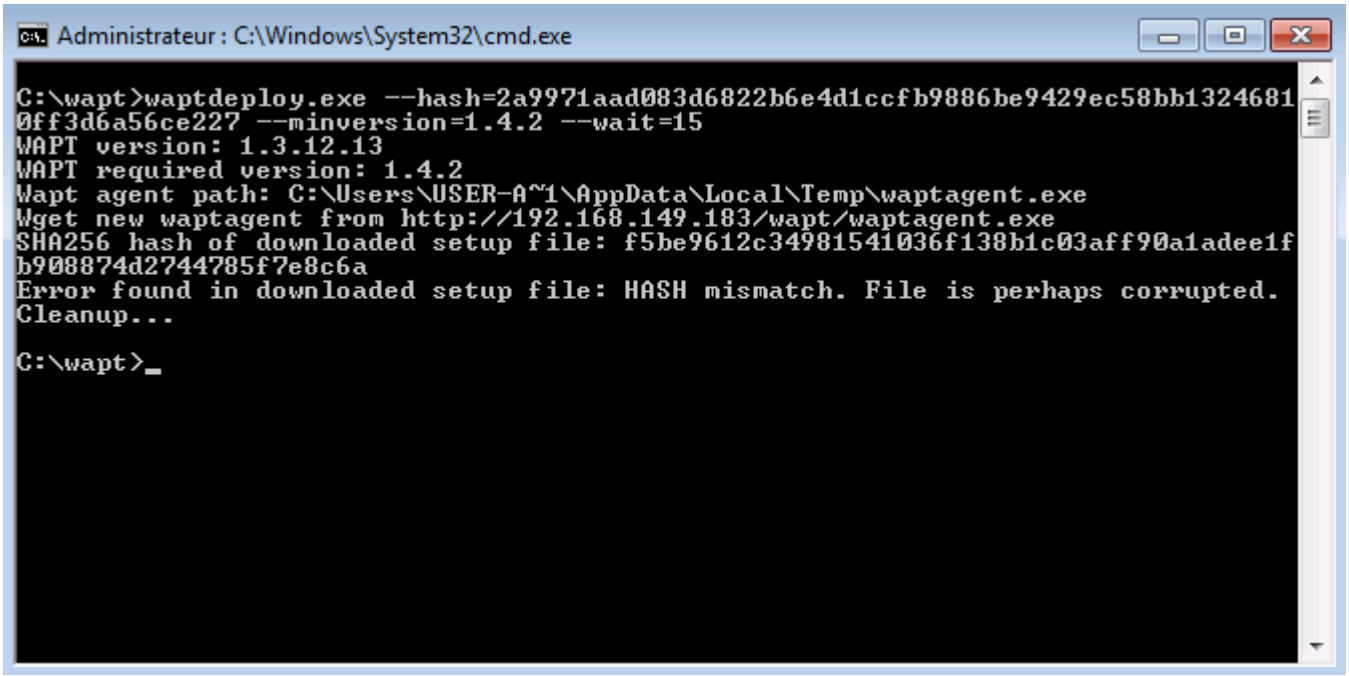


Figure1: Erreur avec le hachage de l'utilitaire WAPT Deploy dans une fenêtre de terminal texte

51.6 Windows n'attend pas le réseau au démarrage

Par défaut Windows n'attend pas le réseau au démarrage de la machine.

Cela peut poser soucis pour l'exécution de **`waptdeploy`** car celui-ci a besoin du réseau au démarrage pour télécharger l'agent WAPT.

2 solutions :

1. Nous vous recommandons d'ajouter `waptdeploy.exe` aux script de démarrage et d'extinction *sur la GPO*.
2. Vous pouvez activer la GPO : **Toujours attendre le réseau au démarrage de l'ordinateur et à la connexion** avec *Configuration de l'ordinateur → Modèles d'administration → Système → Connexion → Toujours attendre le réseau au démarrage de l'ordinateur et à la connexion*

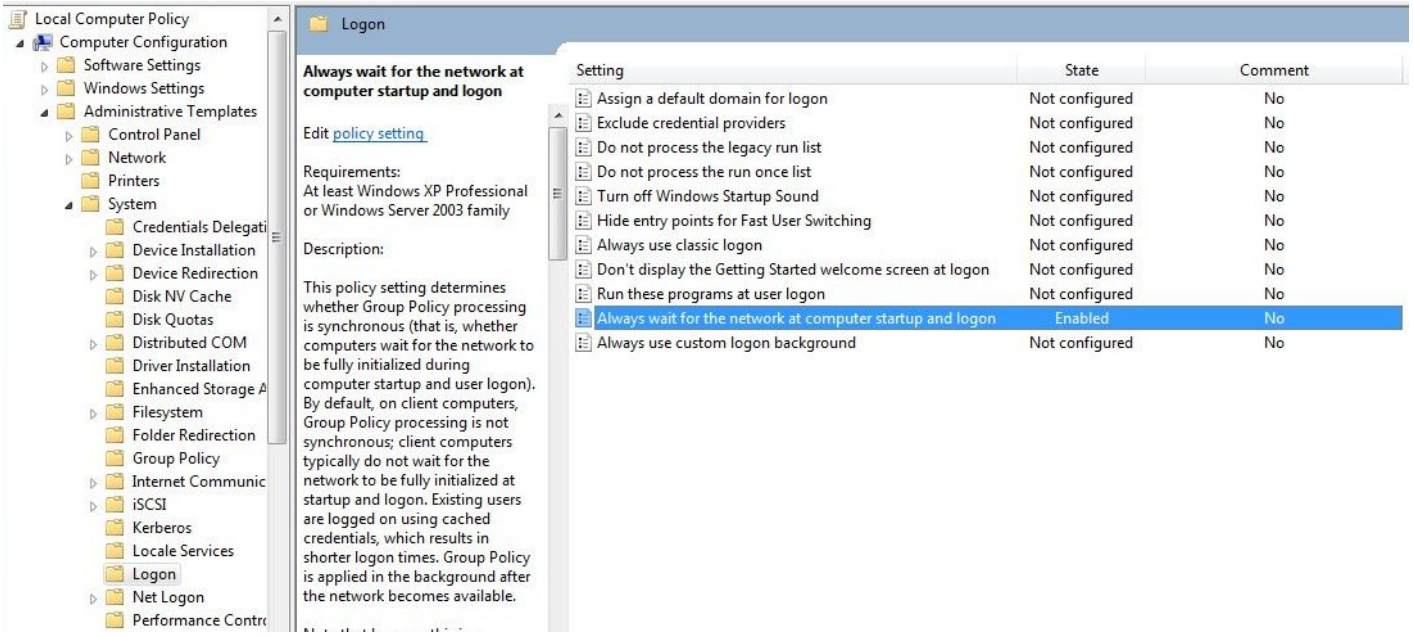


Figure2: GPO pour attendre la connexion réseau

51.7 WAPTexit ne se lance pas

Malgré la présence du script dans les stratégies locales d'arrêt de la machine, **`waptexit.exe`** ne se lance pas à l'extinction du poste.

51.7.1 Résolution: Hybrid Shutdown

Il faut désactiver l'arrêt hybride de Windows10, qui cause par ailleurs beaucoup d'autres comportements étranges. La désactivation de l'arrêt hybride rétablit l'exécution des scripts à l'extinction de la machine.

L'arrêt hybride peut être désactivé en précisant une valeur dans le fichier `wapt-get.ini` de l'agent WAPT, voir *Paramètres pour le wapttray*.

Il est possible de définir cette valeur lorsque *créer un agent WAPT*.

Un paquet WAPT existe pour cet effet tis-disable-hybrid-shutdown.

51.7.2 Windows édition familiale

Les GPO n'étant pas présentes sur les versions familiales de Windows, il est normal qu'elles ne s'exécutent pas.

La solution de contournement est d'utiliser une tâche planifiée qui appelle `C:\Program Files (x86)\wapt\wapt-get.exe` avec le paramètre `upgrade`.

51.7.3 GPO locale corrompue

Il peut arriver que les stratégies de groupes locales de la machine soient corrompues.

Une des solutions possibles consiste à :

- Une des solutions consiste à supprimer les stratégies locales actuelles en supprimant le fichier `C:\windows\system32\GroupPolicy\gpt.ini`, puis en redémarrant la machine, et enfin en relançant l'installation de la tâche d'extinction ;
- Redémarrer l'ordinateur;
- Re-installer la tâche planifiée « à l'arrêt » en :

```
wapt-get add-upgrade-shutdown
```

Si le problème se reproduit, cela signifie peut être qu'une autre application manipule également la GPO.

51.8 WAPTexit se coupe après 15 minutes et n'achève pas l'installation

Par défaut sous Windows, les scripts d'extinction ne peuvent s'exécuter plus de 15 minutes.

Si à l'arrêt de la machine, un script d'extinction n'a pas rendu la main au bout de 15 minutes, le script est interrompu.

Pour résoudre le soucis, il faut modifier la valeur `pre_shutdown_timeout` ainsi que la valeur `max_gpo_script_wait` dans le fichier `wapt-get.ini` de l'Agent WAPT.

Définissez ces valeurs dans `C:\Program Files (x86)\wapt\wapt-get.ini` pour modifier le comportement par défaut.

```
max_gpo_script_wait = 360
pre_shutdown_timeout = 360
```

Le paquet tis-wapt-conf-policy embarque cette configuration.

L'autre solution est d'utiliser la GPO `File.ini`.

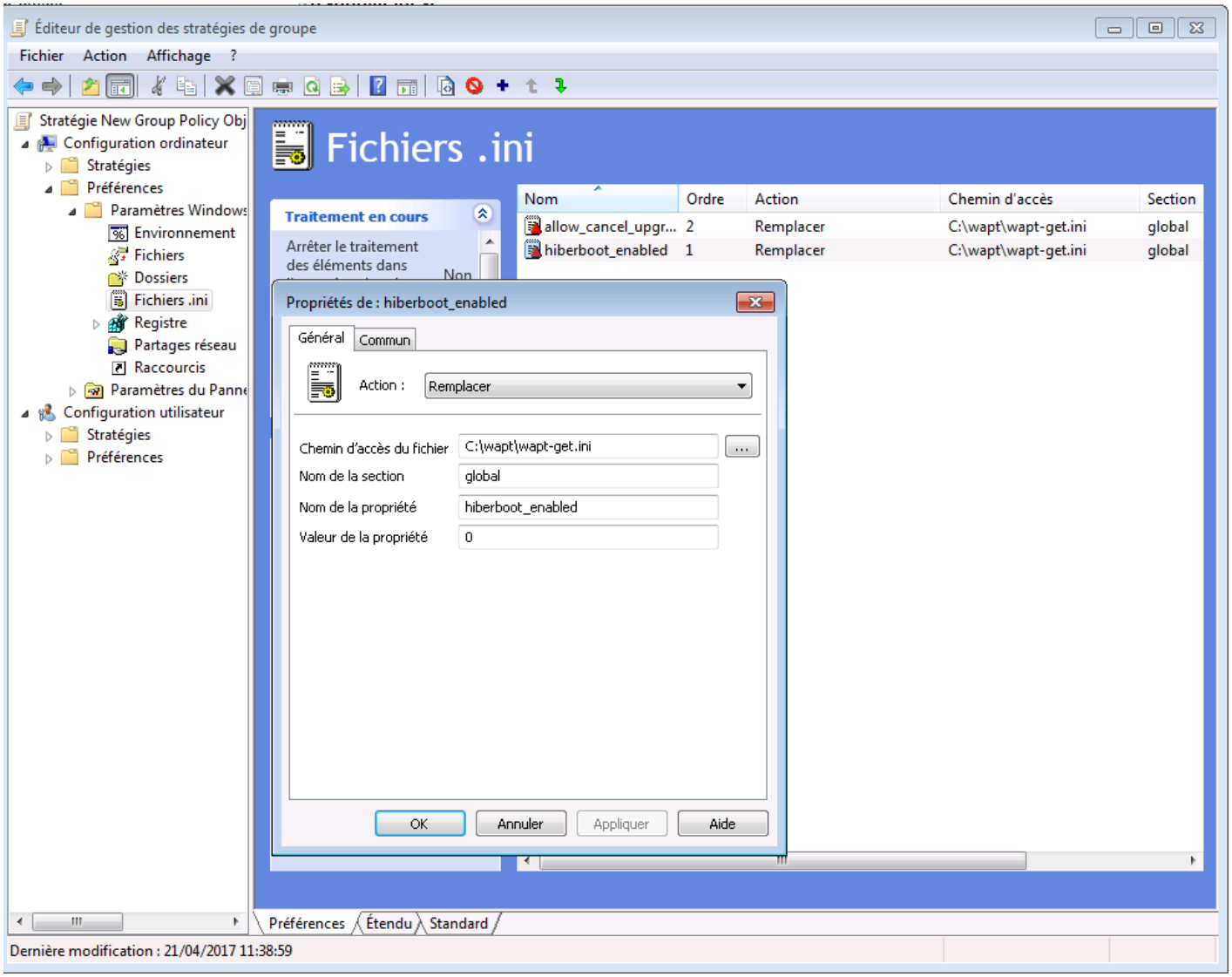
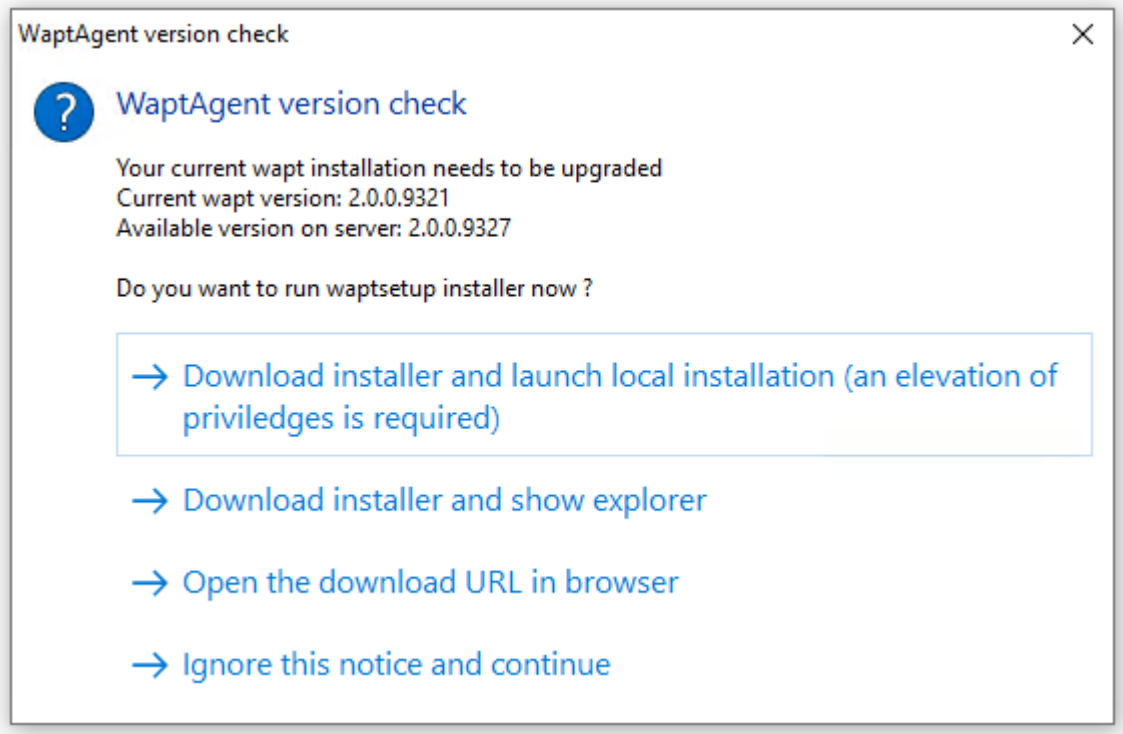


Figure3: Utilisation d'un fichier GPO ini pour configurer le délai d'exécution du script

51.9 Message d'erreur à l'ouverture de la console

51.9.1 Vérification de la version



La version de la console WAPT n'est pas la même que celle du serveur WAPT. Il est recommandé de mettre à jour la console WAPT avec la même version que le serveur WAPT.

51.9.2 Connection refused

La console WAPT ne parvient pas à joindre le port 443 du serveur :

- Vérifier si le service **`Nginx`** est démarré sur le serveur.

```
systemctl status nginx
```

- Si **`Nginx`** n'est pas lancé, relancer **`Nginx`**.

```
systemctl restart nginx
```

- Si le **`Nginx`** ne démarre toujours pas, vous devez analyser les fichiers de logs dans :
  - `/var/log/Nginx/` sur Linux ;
  - `C:\Program Files (x86)\wapt\waptserver\nginx\logs` sur Windows.

51.9.3 Service indisponible

Il est possible que le service *waptserver* soit stoppé :

- Vérifier si le service **waptserver** est en cours d'éxécution.

```
systemctl status waptserver
```

- Si la commande ne retourne rien, lancer le **waptserver**.

```
systemctl start waptserver
```

51.9.4 Error connecting with SSL ... verify failed

La console ne semble pas réussir à vérifier le certificat HTTPS du serveur.

Attention: Attention, avant toute chose vérifiez que vous n'êtes pas victime d'une attaque *man in the middle* !

Note: Si vous venez de refaire votre serveur WAPT et que vous utilisez un certificat auto-signé, vous pouvez récupérer les anciennes clés de votre ancien serveur wapt dans `/opt/wapt/waptserver/apache/ssl`.

- Fermer votre console WAPT.
- Supprimer le dossier `%appdata%\..\Local\waptconsole`.
- Lancer ensuite la commande `wapt-get enable-check-certificate`.
- Assurez-vous que la commande précédente s'est bien déroulée.
- Redémarrer le service WAPT avec `net stop waptservice && net start waptservice`.
- Relancer la console WAPT.

Dans le cas ou vous ne pratiquez pas le *certificate pinning*, cela signifie que le certificat envoyé par le serveur ne pas être vérifié avec le bundle python **certifi**. Veillez à bien fournir la chaîne complète pour le certificat sur le serveur WAPT.

51.9.5 Je ne peux rien faire dans la console WAPT, tout est grisé

La console WAPT semble verrouillée, vous ne pouvez exécuter aucune action, tout est grisé.

Si vous êtes connecté avec un autre utilisateur que le *Superadmin*, vos règles ACL peuvent ne pas être définies correctement.

Pour résoudre ce problème, fermer la Console WAPT et l'ouvrir à nouveau avec le compte *Superadmin*. Ensuite, aller dans :menuselection:`Outils --> Gérer les utilisateurs et les droits WAPT`. Ici, vous verrez l'utilisateur dans la liste, donner à l'utilisateur les permissions appropriées, puis enregistrer et fermer la Console WAPT. Ré-ouvrir la Console WAPT en utilisant votre login.

51.10 Message d'erreur concernant un paquet dans la Console WAPT

51.10.1 Erreur de locale

Figure4: Fenêtre montrant que la console WAPT ne trouve pas un paquet

La Console WAPT affiche cette erreur dans deux situations.

Le paquet n'existe plus dans le dépôt, hors une machine en a besoin

Deux solutions sont envisageables :

- Essayer de récupérer un nouveau paquet dans le store de Tranquil IT.
- Supprimer le paquet des dépendances de la machine.

Lorsque vous essayez d'installer un paquet avec une locale inconnue de la machine

Trois solutions sont envisageables :

- Essayer de récupérer un nouveau paquet dans le store de Tranquil IT.
- Éditer le paquet WAPT et mettre dans le fichier `control` l'option `locale` avec la bonne locale (`locale=en,fr`).
-

51.11 Problèmes pour enregistrer une machine avec le serveur WAPT

Si vous faites un **`wapt-get register`** et que la commande renvoie :

```
FATAL ERROR : ConnectionError: HTTPSConnectionPool(host='XXX.XXX.XXX.XXX', port=443): Max retries␣
↪exceeded with url: /add_host
```

Vous devez vérifier que le port 443 est correctement transmis au serveur WAPT et qu'il n'est pas bloqué par un pare-feu.

51.12 Problème lors du enable-check-certificate

51.12.1 J'ai le message « certificate CN ### sent by server does not match URL host ### lors du enable-check-certificate »

Cela signifie que le CN envoyé par le certificat du serveur ne correspond pas au *wapt_server* du fichier `wapt-get.ini`.

- **2** solutions :

1. Vérifier le paramètre *wapt_server* dans votre fichier `wapt-get.ini`.

   Si votre valeur est correcte, cela signifie sûrement qu'une erreur est survenue lors de la génération du certificat autosigné par le post conf, une faute de frappe ...

   Vous pouvez donc regénérer vos certificats autosignés.

2. Sur le serveur WAPT, supprimez le contenu du dossier `/opt/wapt/waptserver/apache/ssl/`.

   Ensuite, relancez le script de postconf (le même que pendant l'installation initiale, avec les mêmes arguments et les mêmes valeurs).

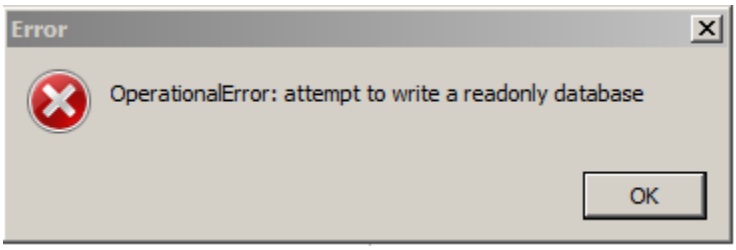
   Enfin, vérifiez bien le nom renseigné lors de l'étape *FQDN for the WAPT serveur* est correcte.

- Vous pouvez maintenant retenter votre **`enable-check-certificate`**.

51.13 Problème avec la création de paquet

51.13.1 Problème de droits avec PyScripter

Lorsqu'on souhaite tester l'installation d'un paquet sur son PC de développement de paquet à partir de **`PyScripter`**, on obtient le message :



Ouvrir une session en tant qu"*Administrateur Local* et refaire l'opération souhaitée.

51.13.2 Mon paquet WAPT est trop volumineux et je n'arrive pas à l'uploader

Quand un paquet est trop volumineux, il faut en général lancer le builder localement puis l'uploader avec **`WinSCP`**.

- Assembler le paquet dans PyScripter ou *manuellement*.

Indication: Le packaging WAPT dans `C:\waptdev`.

- Télécharger et installer **`WinSCP`**

- En utilisant **WinSCP**, **uploadez** votre paquet dans *le dépôt*.
- Une fois le téléchargement terminé, recréer le fichier d'index `Packages` sur le référentiel WAPT en utilisant la commande suivante et en remplaçant **repository**par le *repository location* selon la version du référentiel WAPT.

```
wapt-scanpackages repository
```

51.13.3 Erreur de violation d'accès lors de la re-signature du paquet

Si l'erreur **Access violation** apparaît, c'est parce que le paquet est trop gros.

Éditez le paquet et suivez *cette procédure*.

51.13.4 Paquet WAPT en erreur

Problème d'installation

J'ai un paquet WAPT qui renvoie une erreur et le logiciel n'est pas installé sur l'ordinateur lorsque je vais physiquement vérifier sur l'ordinateur.

Explication

Une erreur s'est produite lors de l'exécution du `setup.py`.

Vous pouvez lire et analyser les messages d'erreur retournés dans la console et tenter de les comprendre.

L'installation sera retentée à chaque **upgrade** jusqu'à ce que le paquet ne génère plus d'erreur.

Solution

- Si WAPT fournit un code d'erreur, chercher ce code d'erreur sur Internet.

  Exemple with a MSI : *1618* : Une autre installation est déjà en cours. Un redémarrage devrait solutionner le problème.

Note: Les différents codes d'erreur MSI sont disponibles ici.

- Se déplacer physiquement sur la machine en erreur et relancer l'installation silencieuse en ligne de commande. Vérifier ensuite que le logiciel à bien été installé.

Attention: Une fois l'installation silencieuse lancée, ne pas intervenir.

L'objectif est de reproduire le comportement de l'agent WAPT.

- Si l'installation fonctionne en mode silencieux, en contexte utilisateur, cela peut signifier que l'installeur ne supporte pas l'installation en compte *SYSTEM*.

- Si cela ne fonctionne toujours pas, lancer l'installation manuellement. Il est possible qu'une erreur apparaisse indiquant explicitement le problème. Exemple (Dépendance manquante: .Net , Java, etc..).
- Il est possible que l'installeur ne supporte pas l'écrasement d'une installation précédente, alors prévoir la désinstallation des anciennes version avant d'installer la nouvelle version .

Erreur « timed out after seconds with output "600.0" »

Certains paquets dans la console retournent l'erreur :

```
"Error timed out after seconds with output '600.0'"
```

Explication

Par défaut lors d'une installation (avec **run**, **install_msi_if_needed** ou **install_exe_if_needed**) WAPT va attendre 600 secondes que l'installeur lui rende la main.

Si l'installeur n'a pas terminé dans ce délai, WAPT coupera l'installation en cours.

Solution

Si vous tentez d'installer un gros logiciel (Office, Solidworks, Libreoffice . . .), il est possible que l'intervalle de 600 secondes ne soit pas suffisant.

Vous devez alors, augmentez cette valeur avec l'argument *timeout*, ex: `timeout = 1200`.

```
run('"setup.exe" /adminfile office2010noreboot.MSP', timeout = 1200)
```

Erreur « has been installed but the uninstall key can not be found »

Certains paquets dans la Console WAPT retournent l'erreur :

```
XXX has been installed but the uninstall key can not be found.
```

Explication

WAPT s'appuie sur Windows pour installer les binaires *.msi* avec **install_msi_if_need** et les binaires *.exe* avec **install_exe_if_need**.

Par défaut, WAPT accepte les codes de retour : *0* (OK) et *3010* (redémarrage nécessaire), et il vérifie la clé de désinstallation résultante (*uninstall key*).

Malheureusement, on ne peut pas toujours se fier à ces codes d'erreur, WAPT vérifie enfin que tout s'est bien déroulé :

- Il vérifie la présence de la clé de désinstallation sur la machine.
- Il vérifie que la version est bien égale ou supérieure à celle renseignée du fichier `control`.
- Si ce n'est pas le cas, il en déduit que le logiciel n'est peut-être pas présent sur la machine.

La fonction bascule alors volontairement le paquet en erreur. L'installation sera retentée lors de chaque upgrade, jusqu'à ce que le paquet ne génère plus d'erreur.

Solution

Attention: Avant toute chose, il convient de se connecter sur la machine en erreur et de vérifier manuellement **si le logiciel est correctement installé** . Si ce n'est pas le cas, se référer à la documentation sur les *problèmes d'installation d'un paquet*..

- Si le logiciel est bien installé, cela signifie peut être que la clé de désinstallation ou la version fournie dans le paquet n'est pas bonne.
- Récupérer la bonne clé de désinstallation et corriger le paquet en conséquence.
- Si l'erreur se produit avec **`install_msi_if_needed`** cela signifie que l'installeur MSI est mal conçu et renvoie une mauvaise *clé de désinstallation*.

Erreur « has been installed and the uninstall key found but version is not good »

Certains paquets dans la Console WAPT retournent l'erreur :

```
XXX has been installed and the *uninstall key* found but version is not good.
```

Explication

Avec les commandes **`install_msi_if_needed`** et **`install_exe_if_needed`**, des vérifications supplémentaires sont effectuées pour vérifier que tout s'est bien passé.

Solution

Attention: Avant toute chose, il convient de se connecter sur la machine en erreur et de vérifier manuellement **si le logiciel est correctement installé** . Si ce n'est pas le cas, se référer à la documentation sur les *problèmes d'installation d'un paquet*..

Solution: Avec `install_msi_if_needed`

Les informations étant extraites depuis l'installeur MSI, cela signifie que le fichiers MSI ne renvoie pas la bonne version ou que la clé de désinstallation retournée n'est pas la bonne version.

Vous pouvez vérifier en utilisant l'utilitaire de ligne de commande Windows.

```
wapt-get list-registry
```

Si la clé retournée n'est pas celle renseignée dans la partie installation du fichier `setup.py` , il n'est pas possible d'utiliser la fonction **`install_msi_if_needed`**.

Il faut rebasculer l'installation avec un simple **`run()`** et gérer les exceptions manuellement.

Avec install_exe_if_needed

Cela signifie probablement que la version renseignée dans la fonction **install_exe_if_needed** n'est pas la bonne. Corriger le paquet WAPT en conséquence.

Note: Si l'argument `min_version` n'a pas été renseigné, WAPT va tenter de récupérer automatiquement la version depuis l'installeur exe.

Pour vérifier la clé de désinstallation utilisée et le numéro de version, utiliser la commande :

```
wapt-get list-registry
```

Si aucune version n'est fournie avec la commande **list-registry**, cela signifie que la clé de désinstallation du logiciel ne fournit pas de version.

2 solutions :

- Utiliser l'argument `get_version` pour fournir un chemin vers une autre `uninstallkey`.

```
def install():

   def versnaps2(key):
       return key['name'].replace('NAPS2 ','')

   install_exe_if_needed('naps2-5.3.3-setup.exe',silentflags='/VERYSILENT',key='NAPS2 (Not Another␣
↪PDF Scanner 2)_is1',get_version=versnaps2)
```

- Fournir une valeur vide en argument pour `min_version` afin d'indiquer à WAPT qu'aucune version n'est à vérifier.

```
min_version=' '
```

Attention: Avec cette méthode **on ne vérifiera plus la version lors de mise à jour !**

51.14 Problèmes fréquents liés aux Antivirus

Certains Antivirus lèvent des alertes pour des composants de WAPT.

Parmis ceux-ci le composant **nssm.exe** est utilisé par WAPT comme utilitaire de service pour l'agent WAPT.

Voici une liste des exceptions possibles à déclarer dans votre interface de gestion centralisé antivirus :

```
"C:\Program Files (x86)\wapt\waptservice\win32\nssm.exe"
"C:\Program Files (x86)\wapt\waptservice\win64\nssm.exe"
"C:\Program Files (x86)\wapt\waptagent.exe"
"C:\Program Files (x86)\wapt\waptconsole.exe"
"C:\Program Files (x86)\wapt\waptexit.exe"
"C:\wapt\waptservice\win32\nssm.exe"
"C:\wapt\waptservice\win64\nssm.exe"
"C:\wapt\waptagent.exe"
```

(suite sur la page suivante)

(suite de la page précédente)

```
"C:\wapt\waptconsole.exe"
"C:\wapt\waptexit.exe"
"C:\Windows\Temp\waptdeploy.exe"
"C:\Windows\Temp\waptagent.exe"
"C:\Windows\Temp\is-?????.tmp\waptagent.tmp"
```

51.15 Je rencontre un problème avec mon proxy - THttpClientSocket.SockRecv(1) read = 0

Si vous avez ce problème :

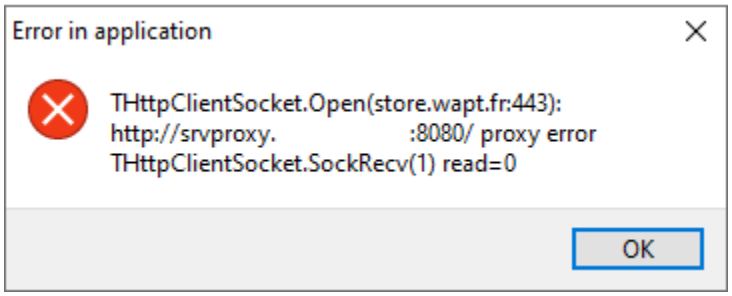


Figure5: Fenêtre montrant une erreur de dépassement de délai du proxy dans la console WAPT

L'erreur vient d'une option *timeout* dans votre `waptconsole.ini`.

En effet, depuis la version WAPT 2.1, le timeout est défini en millisecondes et non pas en secondes comme avant.

Vous allez devoir supprimer votre option *timeout* dans votre `Console WAPT` situé ici : `%localappdata%\waptconsole`.

CHAPTER 52

Erreurs courantes

52.1 Comment déplacer mon dépôt sur une autre partition

Pour de multiples raisons, vous devrez éventuellement déplacer le référentiel sur une autre partition.

Votre dépôt contient 3 dossiers qui peuvent être assez volumineux :

- `wapt` ;
- `wapt-host` ;
- `waptwua`.

52.1.1 Linux

Sous Linux, créez un point de montage sur `fstab`.

Dans cet exemple, la deuxième partition est nommée *part2*.

part2 est une partition **formatée en ext4**.

Debian / Ubuntu

- Créer le dossier temporaire.

```
mkdir /mnt/tmp
```

- Création d'un point de montage temporaire.

```
mount /dev/part2 /mnt/tmp
```

- Déplcaer les dossiers.

```
mv /var/www /mnt/tmp
```

- Démonter la partition.

```
umount /dev/part2
```

- Modifier le fichier `fstab`.

```
vi /etc/fstab
```

- Ajouter la ligne suivante au fichier `fstab`.

```
# <file system> <mount point> <type>  <options> <dump>  <pass>
/dev/part2      /var/www      ext4    defaults  0       0
```

- Monter la partition.

```
mount -a
```

Indication: Si aucune erreur, la partition est montée.

- Vous pouvez vérifier en exécutant.

```
df -h

#Result
Filesystem     1K-blocks     Used Available Use% Mounted on
dev/part2       15G          944M       14G   7% /var/www
```

- Supprimer le dossier temporaire.

```
rm -rf mnt/tmp
```

RedHat et dérivés

- Créer un dossier temporaire pour copier les dossiers.

```
mkdir /mnt/tmp
```

- Création d'un point de montage temporaire.

```
mount /dev/part2 /mnt/tmp
```

- Déplcaer les dossiers.

```
mv /var/www/html /mnt/tmp
```

- Démonter la partition.

```
umount /dev/part2
```

- Modifier le fichier `fstab`.

```
vi /etc/fstab
```

- Ajouter la ligne suivante au fichier `fstab`.

```
# <file system> <mount point> <type>  <options> <dump>  <pass>
/dev/part2      /var/www/html      ext4    defaults  0       0
```

- Monter la partition.

```
mount -a
```

Indication: Si aucune erreur, la partition est montée.

- Vous pouvez vérifier en exécutant.

```
df -h

#Result
Filesystem     1K-blocks     Used Available Use% Mounted on
dev/part2       15G          944M       14G   7% /var/www
```

- Supprimer le dossier temporaire.

```
rm -rf mnt/tmp
```

Windows

Sous Windows, la meilleure méthode est de *sauvegarder* et *restaurer* le serveur sur la nouvelle partition.

Note: Il est possible d'installer le serveur sur une autre partition que `C:`.

52.2 Utiliser un lecteur réseau pour stocker et livrer des paquets WAPT

Le mode de fonctionnement standard de WAPT est avec un serveur Web sécurisé qui fournit les paquets WAPT aux clients WAPT.

Tranquil IT déconseille l'utilisation d'un lecteur réseau pour la livraison de paquets WAPT pour plusieurs raisons :

- Un serveur web est extrêmement facile à installer, à sécuriser, à maintenir, à sauvegarder et à surveiller.
- Pour fonctionner correctement, un paquet WAPT doit être autonome. En effet, nous ne savons pas si le réseau sera disponible au moment du lancement de l'installation (par exemple si nous avons un waptexit qui démarre lorsque la station de travail s'arrête sur un réseau avec une authentification utilisateur 802.1x, il n'y aura plus de réseau disponible au moment de l'installation). La nature autonome du WAPT le rend plus déterministe que les autres solutions de déploiement.

- Une congestion du réseau peut résulter du téléchargement de gros paquets sur de grandes flottes d'appareils parce que vous avez moins de contrôle sur la consommation de bande passante ou bien vous ne pouvez pas terminer un téléchargement partiel.
- Cette méthode casse ou au moins affaiblit le cadre de sécurité du WAPT.
- Cette méthode ne vous permet pas d'exposer vos dépôts sur Internet pour votre personnel itinérant.

Attention: Même si WAPT *peut fonctionner* indépendamment du mode de transport, **Tranquil IT ne supportera pas officiellement l'utilisation d'un lecteur réseau pour stocker et livrer des paquets WAPT**.

52.3 Utiliser la fonction register() dans vos scripts d'audit

La fonction register() force l'envoi au serveur WAPT de l'inventaire matériel et logiciel de l'agent WAPT.

Cette fonction est très éprouvante pour les performances du serveur car elle oblige le serveur à analyser un JSON (Java Script Object Notation) BLOB (Binary Large OBject) relativement grand et à injecter le résultat dans la base de données PostgreSQL.

La fonction est par défaut déclenchée manuellement ou lorsqu'une nouvelle mise à niveau de paquet est appliquée.

Lorsque vous utilisez la fonction register() dans un script d'audit, elle sera exécutée à chaque fois que le script d'audit est déclenché et chargera le serveur sans bénéfice apparent.

Par conséquent, **nous ne recommandons pas l'utilisation de la fonction register() dans les scripts d'audit**.

52.4 EWaptBadControl: "utf8" codec can't decode byte

Si vous recevez ce message, cela peut signifier que vous n'avez pas mis en place correctement votre environnement de développement. Visitez cette *section de la documentation sur la configuration de l'UTF-8 (pas de BOM)*.

52.5 J'ai bien plus d'hôtes dans la console que de paquets host sur mon serveur ?

Suite à une remarque de Philippe LEMAIRE du Lycée Français Alexandre Yersin à Hanoï, si vous utilisez la version Entreprise du WAPT et que vous faites un usage intensif des paquets *unit* ou *profile packages*, vous pouvez réaliser que vous aurez beaucoup plus d'hôtes dans votre console que de **host packages** sur votre serveur WAPT. **C'est normal**.

En fait, les packages WAPT *unit* et *profile* ne sont pas explicitement affectés à l'hôte (c'est-à-dire comme des dépendances dans le *host package*) mais sont implicitement pris en compte par le moteur de dépendance de l'agent WAPT lors de la mise à niveau WAPT.

On peut donc ne pas avoir de paquet *host* sur le serveur si seuls des paquets *unit* sont utilisés pour gérer une flotte d'appareils.

CHAPTER 53

Installer le serveur WAPT avec Ansible

Pour éviter les erreurs et automatiser le déploiement de votre serveur WAPT, nous fournissons des rôles Ansible pour l'installation du serveur WAPT.

Vous pouvez explorer le code source du rôle en visitant le dépôt Tranquil IT sur Github.

53.1 Pré-requis

- Hôtes Debian Linux ou RedHat et dérivées.
- Un compte sudoer sur ces machines.
- Ansible 2.8.

53.2 Installer le rôle Ansible

53.2.1 Découverte

53.2.2 Entreprise

- Installer le rôle Ansible `tranquilit.waptagent`.

```
ansible-galaxy install tranquilit.waptserver
```

- Pour installer le rôle ailleurs, utilisez la sous-commande *-p* comme ceci.

```
ansible-galaxy install tranquilit.waptserver -p /path/to/role/directory/
```

53.3 Utiliser le rôle Ansible

- Assurez-vous d'avoir une clé ssh fonctionnelle déployée sur vos hôtes, sinon vous pouvez en générer et la copier comme indiqué ci-dessous.

```
ssh-keygen -t ed25519
ssh-copy-id -i id_ed25519.pub user@srvwapt.mydomain.lan
ssh user@srvwapt.mydomain.lan -i id_ed25519.pub
```

- Editer l'inventaire ANsible des machines (`./hosts`) et ajouter les machines Linux.

```
[srvwapt]
srvwapt.mydomain.lan ansible_host=192.168.1.40
```

- Créer un *playbook* avec le contenu suivant dans `./playbooks/deploywaptagent.yml`.

```
- hosts: srvwapt
  roles:
    - { role: tranquilit.waptserver }
```

- Lancer votre *playbook* avec la commande suivante.

```
ansible-playbook -i ./hosts ./playbooks/wapt.yml -u user --become --become-method=sudo -K
```

Le serveur est maintenant prêt. Vous pouvez aller à la documentation pour *Installer la console WAPT*!!

53.3.1 Paramètres des rôles Ansible

Les variables disponibles sont énumérées ci-dessous, ainsi que les valeurs par défaut (voir `defaults/main.yml`) :

- Version de WAPT qui sera installée à partir du dépôt WAPT Deb/RPM.

```
wapt_version: "2.0"
```

- Version de PostgreSQL qui sera installée à partir du dépôt WAPT Deb/RPM.

```
pgsql_version: "11"
```

- Version de CentOS utilisée pour l'adresse du dépôt RPM.

```
redhat-based_version: "redhat-based7"
```

- La valeur par défaut de `launch_postconf` est *True*, elle lance le script de postconfiguration du serveur WAPT en mode silencieux.

```
launch_postconf: True
```

53.3.2 Exemple d'un *playbook* Ansible

Voici un exemple d'un *playbook* Ansible.

```
- hosts: srvwapt
  vars_files:
    - vars/main.yml
  roles:
    - tranquilit.waptserver
```

53.4 Déployer l'agent Linux WAPT avec Ansible

Pour éviter les erreurs et automatiser le déploiement de votre serveur WAPT, nous fournissons des rôles Ansible pour l'installation du serveur WAPT sur :

- Debian ;
- Ubuntu ;
- Distribution basée sur Redhat.

Vous pouvez explorer le code source du rôle en visitant le dépôt Tranquil IT sur Github.

53.4.1 Pré-requis

- Hôtes Debian Linux ou RedHat et dérivées.
- Un compte sudoer sur ces machines.
- Ansible 2.8.

53.4.2 Installer le rôle Ansible

- Installer le rôle Ansible `tranquilit.waptagent`.

```
ansible-galaxy install tranquilit.waptagent
```

- Pour installer le rôle ailleurs, utilisez la sous-commande *-p* comme ceci.

```
ansible-galaxy install tranquilit.waptagent -p /path/to/role/directory/
```

53.4.3 Utiliser le rôle Ansible

- Assurez-vous d'avoir une clé ssh fonctionnelle déployée sur vos hôtes, sinon vous pouvez en générer et la copier comme indiqué ci-dessous.

```
ssh-keygen -t ed25519
ssh-copy-id -i id_ed25519.pub user@computer1.mydomain.lan
ssh user@computer1.mydomain.lan -i id_ed25519.pub
```

- Editer l'inventaire ANsible des machines (`./hosts`) et ajouter les machines Linux.

```
[computers]
computer1.mydomain.lan ansible_host=192.168.1.50
computer1.mydomain.lan ansible_host=192.168.1.60
```

- Créer un *playbook* avec le contenu suivant dans `./playbooks/deploywaptagent.yml`.

```
- hosts: computers
  roles:
    - { role: tranquilit.waptagent }
```

- Assurez-vous que toutes les variables sont correctement définies (voir *variables du wapt-get.ini*).
  - `wapt_server_url` ;
  - `wapt_repo_url` ;
  - `wapt_crt`.

Important: Les variables de la configuration est importante car elle va configurer le comportement de WAPT.

Vous **DEVEZ** remplacer le certificat par défaut avec votre certificat public Code-Signing.

- Lancez votre *playbook* avec la commande suivante.

```
ansible-playbook -i ./hosts ./playbooks/deploywaptagent.yml -u user --become --become-method=sudo -K
```

Félicitations, vous avez installé votre serveur WAPT sur votre serveur Linux!

53.4.4 Paramètres des rôles Ansible

Les variables disponibles sont énumérées ci-dessous, ainsi que les valeurs par défaut (voir `defaults/main.yml`).

53.4.5 variables du wapt-get.ini

- Version de WAPT qui sera installée à partir du dépôt WAPT Deb/RPM.

```
wapt_version: "2.0"
```

- Version de CentOS utilisée pour l'adresse du dépôt RPM.

```
redhat-based_version: "redhat-based7"
```

variables du wapt-get.ini

Le paramètre `wapt_server_url` pointe vers votre serveur WAPT et est utilisé par défaut pour le `wapt_repo_url`.

```
wapt_server_url: "https://srvwapt.mydomain.lan"
wapt_repo_url: "{{ wapt_server_url }}/wapt/"
```

Vous pouvez le surcharger comme ceci:

```
wapt_server_url: "https://wapt.landomain.lan"
wapt_repo_url: "https://wapt.otherdomain.com/wapt/"
```

Nom du certificat situé dans le sous-répertoire `files/` du role:

```
wapt_crt: "wapt_ca.crt"
```

53.4.6 Exemple d'un *playbook* Ansible

Voici un exemple d'un *playbook* Ansible.

```
- hosts: hosts
  vars_files:
    - vars/main.yml
  roles:
    - tranquilit.waptagent
```

CHAPTER 54

Utiliser l'API du serveur WAPT

Note: Cette documentation ne décrit pas toutes les APIs (Application Protocol Interfaces) disponibles, mais va cependant se concentrer sur les plus utiles.

Toutes les URLs disponibles peuvent être trouvées dans `/opt/wapt/waptserver/server.py`.

Les URLs sont formées en utilisant la bonne commande depuis le serveur WAPT ex: `https://srvwapt/command_path`.

Indication: Cette documentation contient des exemples en code Python ou bien en curl.

54.1 API V1

54.1.1 /api/v1/hosts

- Récupérer les données enregistrées d'un ou de plusieurs postes.

```
# Args:
#     has_errors (0/1): filter out hosts with packages errors
#     need_upgrade (0/1): filter out hosts with outdated packages
#     groups (csvlist of packages): hosts with packages
#     columns (csvlist of columns):
#     uuid (csvlist of uuid): <uuid1[,uuid2,...]>): filter based on uuid
#     filter (csvlist of field):regular expression: filter based on attributes
#     not_filter (0,1):
#     limit (int): 1000
#     trusted_certs_sha256 (csvlist): filter out hosts based on their trusted package certs
```

(suite sur la page suivante)

(suite de la page précédente)

```
# Returns:
#     result (dict): {'records':[],'files':[]}
#     query:
#       uuid=<uuid>
#     or
#       filter=<csvlist of fields>:regular expression
# """
```

- liste tous les postes. Les paramètres disponibles sont ;
  - *reachable*
  - *computer_fqdn* ==> computer_name
  - *connected_ips*
  - *mac_addresses*

Cette exemple montre une requête avec des paramètres:

```
advanced_hosts_wapt = wgets('https://%s:%s@%s/api/v1/hosts?columns=reachable,computer_fqdn,
↪connected_ips,mac_addresses&limit=10000' % (wapt_user,wapt_password,wapt_url))
parsed = json.loads(advanced_hosts_wapt)
print(json.dumps(parsed, indent=1, sort_keys=True))
```

Cette exemple est une requête globale:

```
hosts_wapt = wgets('https://%s:%s@%s/api/v1/hosts' % (wapt_user,wapt_password,wapt_url))
parsed = json.loads(hosts_wapt)
print(json.dumps(parsed, indent=1, sort_keys=True))
```

Indication: Voici le même exemple avec une simple requête html :

```
https://admin:MYPASSWORD@srvwapt.mydomain.lan/api/v1/hosts
```

Ceui-ci donne une requête avec un statut joignable, le nom de la machine, ses IP connectées et ses adresses MAC. La limite d'affichage est de 10000 postes

```
https://admin:MYPASSWORD@srvwapt.mydomain.lan/api/v1/hosts?columns=reachable,computer_
↪fqdn,connected_ips,mac_addresses&limit=10000
```

54.1.2 /api/v1/groups

- récupère tous les paquets groupes. Les groupes peuvent être trouvés avec la section *groupe* dans le paquet.

```
group_wapt = wgets('https://%s:%s@%s/api/v1/groups' % (wapt_user,wapt_password,wapt_url))
parsed = json.loads(group_wapt)
print(json.dumps(parsed, indent=1, sort_keys=True))
```

Indication: Voici le même exemple avec une simple requête html :

```
https://admin:MYPASSWORD@srvwapt.mydomain.lan/api/v1/groups
```

/api/v1/host_data

dmi

- récupère toutes les informations DMI (Desktop Management Interface) d'un poste :

Note: # # Récupère des données supplémentaires d'un poste # query: # uuid=<uuid> # field=packages, dmi ou softwares

Note: le *dmi* n'est pas la seule option disponible. Vous pouvez aussi chercher des informations en utilisant *installed_packages*, *wsusupdates* ou *installed_softwares*.

```
dmi_host_data_wapt = wgets('https://%s:%s@%s/api/v1/host_data?uuid=UUID&field=dmi' % (wapt_user,
↪wapt_password,wapt_url))
#print(dmi_host_data_wapt)
parsed = json.loads(dmi_host_data_wapt)
print(json.dumps(parsed, indent=1, sort_keys=True))
```

Indication: Voici le même exemple avec une simple requête html :

```
https://admin:MYPASSWORD@srvwapt.mydomain.lan/api/v1/host_data?uuid=14F620FF-DE70-9E5B-996A-
↪B597E8F9B4AD&field=dmi
```

installed_packages

L'option *installed_packages* va lister tous les paquets installés sur un poste en particulier.

```
install_packages_data_wapt = wgets('https://%s:%s@%s/api/v1/host_data?uuid=UUID&field=installed_
↪packages' % (wapt_user,wapt_password,wapt_url))
parsed = json.loads(install_packages_data_wapt)
print(json.dumps(parsed, indent=1, sort_keys=True))
```

Indication: Voici le même exemple avec une simple requête html :

```
https://admin:MYPASSWORD@srvwapt.mydomain.lan/api/v1/host_data?uuid=14F620FF-DE70-9E5B-996A-
↪B597E8F9B4AD&field=installed_packages
```

installed_softwares

L'option *installed_softwares* va lister tous les logiciels installés sur un poste en particulier.

```
install_softwares_data_wapt = wgets('https://%s:%s@%s/api/v1/host_data?uuid=UUID&field=installed_
↪softwares' % (wapt_user,wapt_password,wapt_url))
#print(install_softwares_data_wapt)
parsed = json.loads(install_softwares_data_wapt)
print(json.dumps(parsed, indent=1, sort_keys=True))
```

Indication: Voici le même exemple avec une simple requête html :

```
https://admin:MYPASSWORD@srvwapt.mydomain.lan/api/v1/host_data?uuid=14F620FF-DE70-9E5B-996A-
↪B597E8F9B4AD&field=installed_softwares
```

wsusupdates

L'option *wsusupdates* va lister toutes les mises à jour installés sur un poste en particulier.

```
wsusupdates_data_wapt = wgets('https://%s:%s@%s/api/v1/host_data?uuid=UUID&field=wsusupdates' %␣
↪(wapt_user,wapt_password,wapt_url))
#print(wsusupdates_data_wapt)
parsed = json.loads(wsusupdates_data_wapt)
print(json.dumps(parsed, indent=1, sort_keys=True))
```

Indication: Voici le même exemple avec une simple requête html :

```
https://admin:MYPASSWORD@srvwapt.mydomain.lan/api/v1/host_data?uuid=14F620FF-DE70-9E5B-996A-
↪B597E8F9B4AD&field=wsusupdates
```

54.1.3 /api/v1/usage_statistics

Récupère les statistiques d'usage du serveur.

Indication: Cette API est utile si vous avez plusieurs serveurs WAPT et si vous voulez savoir combien de postes il y a.

```
usage_statistics_wapt =wgets('https://%s:%s@%s/api/v1/usage_statistics' % (wapt_user,wapt_password,
↪wapt_url))
#print(usage_statistics_wapt)
parsed = json.loads(usage_statistics_wapt)
print(json.dumps(parsed, indent=1, sort_keys=True))
```

Indication: Voici le même exemple avec une simple requête html :

```
https://admin:MYPASSWORD@srvwapt.mydomain.lan/api/v1/usage_statistics
```

54.2 API V2

54.2.1 /api/v2/waptagent_version

Affiche la version du **waptagent.exe** sur le serveur.

```
waptagent_version = wgets('https://%s:%s@%s/api/v2/waptagent_version' % (wapt_user,wapt_password,
↪wapt_url))
parsed = json.loads(waptagent_version)
print(json.dumps(parsed, indent=1, sort_keys=True))
```

Indication:

Voici le même exemple avec une simple requête html :

```
https://admin:MYPASSWORD@srvwapt.mydomain.lan/api/v2/waptagent_version
```

54.3 API V3

54.3.1 /api/v3/packages

Liste les paquets sur le dépôt privé, il récupère le fichier control sur les paquets.

```
packages_wapt = wgets('https://%s:%s@%s/api/v3/packages' % (wapt_user,wapt_password,wapt_url))
parsed = json.loads(packages_wapt)
print(json.dumps(parsed, indent=1, sort_keys=True))
```

Indication: Voici le même exemple avec une simple requête html :

```
https://admin:MYPASSWORD@srvwapt.mydomain.lan/api/v3/packages
```

54.3.2 /api/v3/known_packages

Liste tous les paquets avec l'information *signed_on*.

```
known_packages_wapt = wgets('https://%s:%s@%s/api/v3/known_packages' % (wapt_user,wapt_password,
↪wapt_url))
parsed = json.loads(known_packages_wapt)
print(json.dumps(parsed, indent=1, sort_keys=True))
```

Indication: Voici le même exemple avec une simple requête html :

```
https://admin:MYPASSWORD@srvwapt.mydomain.lan/api/v3/known_packages
```

54.3.3 /api/v3/trigger_cancel_task

Annule une tâche en cours.

```
trigger_cancel_task = wgets('https://%s:%s@%s/api/v3/trigger_cancel_task' % (wapt_user,wapt_
↪password,wapt_url))
parsed = json.loads(trigger_cancel_task)
print(json.dumps(parsed, indent=1, sort_keys=True))
```

54.3.4 /api/v3/get_ad_ou

Liste les OU vues par les postes et affichées dans la console WAPT.

```
get_ad_ou = wgets('https://%s:%s@%s/api/v3/get_ad_ou' % (wapt_user,wapt_password,wapt_url))
parsed = json.loads(get_ad_ou)
print(json.dumps(parsed, indent=1, sort_keys=True))
```

Indication: Voici le même exemple avec une simple requête html :

```
https://admin:MYPASSWORD@srvwapt.mydomain.lan/api/v3/get_ad_ou
```

54.3.5 /api/v3/get_ad_sites

Liste les sites Active Directory.

```
get_ad_sites = wgets('https://%s:%s@%s/api/v3/get_ad_sites' % (wapt_user,wapt_password,wapt_url))
parsed = json.loads(get_ad_sites)
print(json.dumps(parsed, indent=1, sort_keys=True))
```

Indication: Voici le même exemple avec une simple requête html :

```
https://admin:MYPASSWORD@srvwapt.mydomain.lan/api/v3/get_ad_sites
```

54.3.6 /api/v3/hosts_for_package

Liste les hôtes avec un packaging spécifique installé.

```
hosts_for_package = wgets('https://%s:%s@%s/api/v3/hosts_for_package?package=PACKAGE' % (wapt_user,
↪wapt_password,wapt_url))
parsed = json.loads(hosts_for_package)
print(json.dumps(parsed, indent=1, sort_keys=True))
```

Indication: Voici le même exemple avec une simple requête html :

```
https://admin:MYPASSWORD@srvwapt.mydomain.lan/api/v3/hosts_for_package?package=demo-namepackage
```

54.3.7 /api/v3/host_tasks_status

Liste les tâches d'un poste en particulier.

```
host_tasks_status = wgets('https://%s:%s@%s/api/v3/host_tasks_status?uuid=UUID' % (wapt_user,wapt_
↪password,wapt_url))
parsed = json.loads(host_tasks_status)
print(json.dumps(parsed, indent=1, sort_keys=True))
```

Indication: Voici le même exemple avec une simple requête html :

```
https://admin:MYPASSWORD@srvwapt.mydomain.lan/api/v3/host_tasks_status?uuid=14F620FF-DE70-9E5B-996A-
↪B597E8F9B4AD
```

Attention: **Les API ci-après suivent la méthode POST**.

54.3.8 /api/v3/upload_packages

À faire: Tests

54.3.9 /api/v3/upload_hosts

À faire: Tests

54.3.10 /api/v3/change_password

Change le mot de passe du compte admin [ce compte uniquement]. La requête doit être un dictionnaire python *{}*. Les clés doivent être:

- user;
- password;
- new_password;

```
curl --insecure -X POST --data-raw '{"user":"USER","password":"old_password","new_password":"new_
↪password"}' -H "Content-Type: application/json" "https://user:old_password@srvwapt/api/v3/change_
↪password"
```

54.3.11 /api/v3/login

Initialiser une connexion au serveur.

```
curl --insecure -X POST --data-raw '{"user":"admin","password":"MYPASSWORD"}' -H "Content-Type:␣
↪application/json" "https://srvwapt.mydomain.lan/api/v3/login"

{"msg": "Authentication OK", "result": {"edition": "enterprise", "hosts_count": 6, "version": "1.7.4
↪", "server_domain": "mydomain.lan", "server_uuid": "32464dd6-c261-11e8-87be-cee799b43a00"},
↪"success": true, "request_time": 0.03377699851989746}
```

Indication: Nous pouvons faire une connexion avec un formulaire html plutôt que POST: https://admin:MYPASSWORD@srvwapt.mydomain.lan/api/v3/get_ad_sites

54.3.12 /api/v3/packages_delete

Supprime un paquet d'une version précise. La requête doit être une liste *[]*. Elle peut prendre plusieurs paquets séparés par des virgules ,.

Exemple:

```
curl --insecure -X POST --data-raw '["demo-libreoffice-stable_5.4.6.2-3_all.wapt"]' -H "Content-
↪Type: application/json" "https://admin:MYPASSWORD@srvwapt.mydomain.lan/api/v3/packages_delete"
```

54.3.13 /api/v3/reset_hosts_sid

Initialiser une connexion au serveur.

La syntaxe est : **--data-raw** : un dictionnaire avec pour clé les uuid et pour valeur l'uuid du poste.

```
curl --insecure -X POST --data-raw '{"uuids":["UUID"]}' -H "Content-Type: application/json" "https:/
↪/admin:MYPASSWORD@srvwapt.mydomain.lan/api/v3/reset_hosts_sid"

{"msg": "Hosts connection reset launched for 1 host(s)", "result": {}, "success": true, "request_
↪time": null}
```

Indication: Si vous voulez plusieurs postes:

```
curl --insecure -X POST --data-raw '{"uuids":["UUID#1","UUID#2"]}' -H "Content-Type: application/
↪json" "https://admin:MYPASSWORD@srvwapt.mydomain.lan/api/v3/reset_hosts_sid"

{"msg": "Hosts connection reset launched for 2 host(s)", "result": {}, "success": true, "request_
↪time": null}
```

54.3.14 /api/v3/trigger_wakeonlan

Si les postes ont le WakeOnLan d'activé, cette API est utile.

La syntaxe est : **--data-raw** : un dictionnaire avec pour clé les uuid et pour valeur l'uuid du poste.

```
curl --insecure -X POST --data-raw '{"uuids":["UUID"]}' -H "Content-Type: application/json" "https:/
↪/admin:MYPASSWORD@srvwapt.mydomain.lan/api/v3/trigger_wakeonlan"

{"msg": "Wakeonlan packets sent to 1 machines.", "result": [{"computer_fqdn": "computer_fqdn", "mac_
↪addresses": ["mac_addresses"], "uuid": "UUID"}], "success": true, "request_time": null}
```

Indication: Si vous voulez plusieurs postes:

```
curl --insecure -X POST --data-raw '{"uuids":["UUID#1","UUID#2"]}' -H "Content-Type: application/
↪json" "https://admin:MYPASSWORD@srvwapt.mydomain.lan/api/v3/trigger_wakeonlan"

{"msg": "Wakeonlan packets sent to 2 machines.", "result": [{"computer_fqdn": "computer_fqdn#1",
↪"mac_addresses": ["mac_addresses#1"], "uuid": "UUID#1"}, {"computer_fqdn": "computer_fqdn#2",
↪"mac_addresses": ["mac_addresses#2"], "uuid": "UUID#2"}], "success": true, "request_time": null}
```

(suite sur la page suivante)

(suite de la page précédente)

54.3.15 /api/v3/hosts_delete

```
"""Remove one or several hosts from the WAPT Server database and optionnally the host packages

Args:
    uuids (list): list of uuids to delete
    filter (csvlist of field:regular expression): filter based on attributes
    delete_packages (bool): delete host's packages
    delete_inventory (bool): delete host's inventory

Returns:
    result (dict):
"""
```

Si vous voulez supprimer un poste de l'inventaire:

```
curl --insecure -X POST --data-raw '{"uuids":["UUID"],"delete_inventory":"True","delete_packages":
↪"True"}' -H "Content-Type: application/json" "https://admin:MYPASSWORD@srvwapt.mydomain.lan/api/
↪v3/hosts_delete"

{"msg": "1 files removed from host repository\n1 hosts removed from DB", "result": {"files": ["/var/
↪www/wapt-host/UUID.wapt"], "records": [{"computer_fqdn": "computer_fqdn", "uuid": "UUID"}]},
↪"success": true, "request_time": null}
```

Si vous ne voulez pas le supprimer de l'inventaire du serveur:

```
curl --insecure -X POST --data-raw '{"uuids":["UUID"],"delete_inventory":"False","delete_packages":
↪"False"}' -H "Content-Type: application/json" "https://admin:MYPASSWORD@srvwapt.mydomain.lan/api/
↪v3/hosts_delete"

{"msg": "0 files removed from host repository\n1 hosts removed from DB", "result": {"files": [],
↪"records": [{"computer_fqdn": "computer_fqdn", "uuid": "UUID"}]}, "success": true, "request_time
↪": null}
```

54.3.16 /api/v3/trigger_host_action

À faire: Tests

54.3.17 /api/v3/upload_waptsetup

```
# Upload waptsetup

#Handle the upload of customized waptagent.exe into wapt repository

### DOES NOT WORK
#curl --insecure -X POST -H "Content-Type: multipart/form-data" -F 'data=@waptagent.exe' "https://
↪admin:MYPASSWORD@srvwapt.mydomain.lan/upload_waptsetup"
```

54.3.18 /api/v3/ping

Ping va récupérer les informations générales d'un serveur WAPT.

```
# https://srvwapt.mydomain.lan/ping
# Lists WAPT Server informations

ping_wapt = wgets('https://%s:%s@%s/ping' % (wapt_user,wapt_password,wapt_url))
parsed = json.loads(ping_wapt)
print(json.dumps(parsed, indent=1, sort_keys=True))
```

CHAPTER 55

Contacter l'éditeur de WAPT

Contactez-nous pour plus d'informations :

- **Tranquil IT**: https://www.tranquil.it/
- **Twitter**: https://twitter.com/tranquil_it
- **Linkedin**: https://www.linkedin.com/company/tranquil-it
- **Forum en Français**: https://forum.tranquil.it/
- **Forum en Anglais**: https://www.reddit.com/r/WAPT
- **Discord**: https://discord.gg/hFdrqs2C5g

CHAPTER 56

Glossaire

Administrateur

Administrateurs

Développeur de Paquet

Développeurs de Paquets Un **Administrateur** est un individu pouvant signer des paquets, qu'ils intègrent ou non du code python et les charger sur le dépôt principal.

Administrateur Local

Administrateurs Locaux Un **Administrateur Local** est un utilisateur disposant des droits d'administration locaux sur les postes équipés de WAPT.

Déployeur de Paquet

Déployeurs de Paquets Un **Déployeur de Paquet** est un individu pouvant signer des paquets ne contenant pas de code python (en général les paquets de type *group*, *unit* et *host*) et les charger sur le dépôt principal. Il est typiquement un membre d'une équipe informatique locale qui une bonne connaissance des besoins des utilisateurs.

SuperAdmin Le **SuperAdmin** est l'*Utilisateur* dont l'identifiant et le mot de passe est défini lors de la post-configuration du serveur WAPT. Dans la version WAPT Discovery, il est l'unique *Administrateur* de WAPT.

Utilisateur

Utilisateurs Un **Utilisateur** est un individu qui utilise une machine équipée de l'agent WAPT (WAPT **Enterprise** et **Discovery**).

Organisation

Organisations L'**Organisation** correspond au périmètre de responsabilité dans lequel est exploitée la solution WAPT.

ANSSI **Agence Nationale de la Sécurité des Systèmes d'Information** est un service français en charge d'assurer la sécurité des informations sensibles de l'État Français et d'une mission de conseil et de soutien aux administrations et aux opérateurs d'importance vitale.

Site internet : https://www.ssi.gouv.fr/

DNS **Domain Name System** est un service permettant de traduire un nom de domaine en informations de plusieurs types qui y sont associées, notamment en adresses IP de la machine portant ce nom.

FQDN **Fully Qualified Domain Name** est un nom de domaine complètement qualifié. C'est la notation complète d'un nom de domaine qui révèle la position absolue de la machine dans l'arborescence DNS en indiquant tous les niveaux supérieurs jusqu'à la racine. Exemple de FQDN : wapt.nantes.pdl.organisation.fr.

EPEL **Extra Packages for Enterprise Linux** est un dépôt additionel pour CentOS et RedHat.

GPO Les **Group Policy Objects** ou **Objets de Stratégie de Groupe** sont des objets définissant des stratégies de sécurité dans un environnement Windows. Les stratégies peuvent être définies localement à l'aide de `gpedit.msc` ou définies globalement en domaine Active Directory.

IDE **Integrated Development Environment** (environnement de développement intégré) est un ensemble d'outils qui augmente la productivité des développeurs logiciels. Un IDE permet notamment de déboguer ligne par ligne le code source d'un programme, d'éditer, compiler et exécuter dans une seule interface.

MMC **Microsoft Management Console** est un gestionnaire de console virtuelle incorporé dans Microsoft Windows, qui sert de conteneur pour des interfaces graphiques de configuration.

NAT **Network Address Translation** est un mécanisme qui permet à des machines disposant d'adresses qui font partie d'un intranet de communiquer avec le reste d'Internet en semblant utiliser des adresses externes uniques et routables, au travers d'un routeur.

Setuphelpers **Setuphelpers** est une librairie Python écrite spécialement pour WAPT. Elle contient un ensemble de fonctions et de variables utiles au développement de paquets, pour la manipulation de fichiers, création de raccourcis, etc.

SRV Le champ **SRV** permet de définir un serveur spécifique pour une application, notamment pour la répartition de charge.

virtualhost En informatique, l'"**hébergement virtuel** (de l'anglais virtual hosting abrégé **vhost**) est une méthode que les serveurs tels que les serveurs Web utilisent pour accueillir plus d'un nom de domaine sur le même ordinateur, parfois sur la même adresse IP, tout en maintenant une gestion séparée de chacun de ces noms. Cela permet de partager les ressources du serveur, comme la mémoire et le processeur, sans nécessiter que tous les services fournis utilisent le même nom d'hôte.

waptagent **waptagent** est l'agent WAPT installé sur chaque ordinateur client.

waptexit **waptexit**est une commande WAPT lancée par le script d'arrêt de Windows pour mettre à jour les packages qui ont un statut *PENDING* (sur les versions professionnelles de Windows).

waptsetup **waptsetup** est un logiciel permettant d'installer la console WAPT.

Websocket **Websocket** est une couche applicative Web bidirectionnelle permettant la communication client-serveur en utilisant une connexion TCP.

UUID Un **UUID** est un identifiant normalisé et réputé unique ; ainsi un UUID dans le contexte de WAPT permet d'identifier de manière unique une machine. Pour en savoir plus, suivre https://en.wikipedia.org/wiki/Universally_unique_identifier.

Champ CNAME

Champs CNAME Un enregistrement **CNAME** ou enregistrement de nom canonique est un type d'enregistrement-ressource dans le Domain Name System (*DNS*) qui spécifie que le nom de domaine est un alias d'un autre nom de domaine canonique.

Champ A

Champs A Un **champ A** met en relation un nom (en général le nom physique d'un serveur) avec une IP.

Autorité de Certification Une CA est un tiers de confiance permettant d'authentifier l'identité des correspondants.

PKI **Public Key Infrastructure**, ou Infrastructure à Clés Publiques est un ensemble de composants physiques, de procédures humaines et de logiciels destiné à gérer les clés publiques des utilisateurs d'un système.

CHAPTER 57

Présentation des principes de sécurité

Date	avr. 17, 2023
Rédigé par	Hubert TOUVET, Vincent Cardon
S'applique à WAPT	>= 2.3.0.13180
Version du document	2.3.0.0-0
Hash Git	368951c0f29223af79a7ad6f74d01726029c12fa

- *Préambule et définitions*
- *Périmètre à sécuriser*
- *Description des utilisateurs typiques*
- *Description des menaces pesant sur les biens sensibles WAPT*
  - *Bien sensible B1 : Les communications*
  - *Bien sensible B2 : Les données d'inventaire*

  - *Bien sensible B3 : Les journaux d'historique*
  - *Bien sensible B4 : Les valeurs de configuration*
  - *Bien sensible B5 : Les exécutables WAPT installés sur les postes client*
  - *Bien sensible B6 : L'authentification*
- *Description des hypothèses sur l'environment d'exploitation de WAPT*
  - *Hypothèse H1 : Les Administrateurs et Déployeurs de Paquet WAPT sont formés*
  - *Hypothèse H2 : Les systèmes subjacents à WAPT sont sains*
  - *Hypothèse H3 : Les binaires nécessaires au fonctionnement de WAPT sont intègres*
  - *Hypothèse H4 : Les paquets WAPT sont construits de manière sûre*
  - *Hypothèse H5 : Les Utilisateurs des postes client ne sont pas Administrateurs Locaux*
  - *Hypothèse H6 : Les Administrateurs Locaux des postes client sont formés*
- *Description des menaces pesant sur les biens sensibles WAPT*
  - *Menace M1 : Installation d'un logiciel malveillant par une entité non-autorisée*
  - *Menace M2 : Altération de valeurs de configuration par une entité non-autorisée*
  - *Menace M3 : Accès illégitime par une entité non-autorisée*
  - *Menace M4 : Écoute du réseau par une entité non-autorisée*
  - *Menace M5 : Altération du trafic réseau par une entité non-autorisée (Type* Man In the Middle*)*
- *Description des fonctions de sécurité de WAPT*
  - *Fonction de sécurité F1 : Authentification des contrôles d'accès*
    - *Fonction de sécurité F1A : Authentification d'une machine lors de son enregistrement initial dans la base de données WAPT*
      - *Solution mise en place*
    - *Fonction de sécurité F1B : Vérification des certificats HTTPS du serveur par les clients WAPT*
      - *Solution mise en place*
    - *Fonction de sécurité F1C : Aucun port n'écoute sur les agents WAPT*
      - *Solution mise en place*
    - *Fonction de sécurité F1D : Signature des remontées d'inventaire*
      - *Solution mise en place*
    - *Fonction de sécurité F1E : Vérification des droits avant l'exécution de certaines actions WAPT*
      - *Solution mise en place*
  - *Fonction de sécurité F2 : Protection de l'intégrité du processus d'installation des paquets WAPT*
    - *Fonction de sécurité F2A : Signature des paquets WAPT*
      - *Solution mise en place*
    - *Fonction de sécurité F2B : Signature des attributs du fichier* control

            · *Solution mise en place*
        * *Fonction de sécurité F2C : Restriction d'accès au répertoire d'installation de l'agent WAPT*
        * *Fonction de sécurité F2D : Restriction totale d'accès au répertoire de stockage du couple clé privé / certificat de signature d'inventaire*
    – *Fonction de sécurité F3 : Sécurisation des communications entre les différents composants WAPT*
        * *Fonction de sécurité F3A : Signature des requêtes envoyées aux agents WAPT*
            · *Solution mise en place*
- *Présentation des processus cryptographiques serveur*
- *Présentation des processus cryptographiques serveur*
- *Matrices de couvertures*
    – *Menaces et biens sensibles*
    – *Menaces et fonctions de sécurité*

Sont documentés ici différents principes avancés de sécurité incorporés dans WAPT.

La lecture de cette documentation n'est pas indispensable pour utiliser WAPT ; elle est cependant recommandée pour vous permettre de mieux comprendre certains choix architecturaux.

57.1 Préambule et définitions

Attention: Le service WAPT fonctionne en compte système **privilégié**.

Indication: les sous-composantes **`wapttray`**, **`waptservice`** et **`waptexit`** de l'agent WAPT peuvent être optionnellement désactivées en fonction du contexte d'usage.

57.2 Périmètre à sécuriser

Les éléments à sécuriser qui concernent strictement WAPT sont :

- **Le serveur WAPT** (*waptserver*) ;
- **Les agents WAPT** (*wapt-get*) et ses sous-composantes (*wapttray*, *waptservice* et *waptexit*) ;
- **La console de management** (*waptconsole*) ;
- **Les communications réseaux** entre ces différentes composantes.

En complément des éléments listés ci-dessus, un exploitant de WAPT devra choisir et suivre une méthodologie adaptée au contexte de son *Organisation* pour :

- Assurer un téléchargement sûr de tous les autres fichiers servant à constituer un paquet WAPT.

- Rédiger le script python `setup.py` d'installation d'un paquet WAPT de telle manière à éviter toute faille de sécurité ou de confidentialité exploitable.
- Gérer de manière sûre les clés privées de signature des paquets.
- Gérer de manière sûre les Autorités de certification et de révocation des certificats SSL et HTTPS.

La gestion sûre de ces éléments complémentaires est exclue du périmètre de cette documentation.

57.3 Description des utilisateurs typiques

Les rôles suivants doivent être compris pour évaluer les principes de sécurité présents dans WAPT :

- **Utilisateur**

  Un *Itilisateur* est un individu équipé d'un machine avec l'agent WAPT (**Enterprise** et **Discovery**).

- **Déployeur de Paquet**

  Un *Déployeur de Paquet* est un individu pouvant signer des paquets ne contenant **PAS** de code python (en général les paquets de type *group*, *unit* et *host*) et les charger sur le dépôt principal (**Enterprise**).

- **Développeur de Paquet**

  Un *Développeur de Paquet** est un individu pouvant signer des paquets, qu'ils intègrent ou non du code python et les charger sur le dépôt principal (**Enterprise**);

Note: La distinction entre *Déployeur de Paquet* et *Développeur de Paquet* n'existe que dans la version **Enterprise** de WAPT.

- **SuperAdmin**

  Le *SuperAdmin* est un individu avec tous les droits dans WAPT (**Enterprise** and **Discovery**).

- **Administrateur Local**

  Utilisateur disposant des droits d'administration locaux sur les postes équipés de WAPT (**Enterprise** et **Discovery**).

Note: En fonction du contexte de la documentation et de la version du produit, un *Administrateur* désignera un *Déployeur de Paquet*, un *Développeur de Paquet* ou bien le *SuperAdmin*.

Note: Les *Utilisateurs* membres du groupe de sécurité Active Directory **waptselfservice** sont considérés comme des *Administrateurs Locaux* du point de vue de la sécurité WAPT.

57.4 Description des menaces pesant sur les biens sensibles WAPT

Par définition, un bien sensible est une donnée (ou fonction) jugée comme ayant de la valeur pour un attaquant.

Sa valeur est estimée selon des critères de sécurité (aussi appelés besoins de sécurité) :

- disponibilité ;
- intégrité ;
- confidentialité ;
- authenticité.

Les biens sensibles à protéger sont les suivants :

57.4.1 Bien sensible B1 : Les communications

Les communications entre le serveur central et les agents ainsi que les communications entre la console et le serveur sont un bien sensible et doivent être protégées.

Note: Besoin de sécurité de l'authentification :

- intégrité ;
- confidentialité ;
- authenticité.

57.4.2 Bien sensible B2 : Les données d'inventaire

Les informations sur l'état de déploiement des paquets, ainsi que configuration matérielle et logicielle des postes clients sont un bien sensible et doivent être protégées.

Note: Besoin de sécurité des données d'inventaire :

- intégrité ;
- confidentialité.

57.4.3 Bien sensible B3 : Les journaux d'historique

Les journaux générés par WAPT sur le serveur central et les agents sont un bien sensible et doivent être protégés.

Note: Besoin de sécurité des journaux d'historique :

- disponibilité.

57.4.4 Bien sensible B4 : Les valeurs de configuration

Les valeurs de configuration du serveur (clés du serveur https, configuration accès à la base de données, configuration de l'authentification au serveur) sont un bien sensible et doivent être protégés.

Note: Besoin de sécurité des valeurs de configuration :

- intégrité ;
- confidentialité.

57.4.5 Bien sensible B5 : Les exécutables WAPT installés sur les postes client

Les exécutables WAPT installés sur les postes client managés (contenu du répertoire wapt incluant les binaires, les dll, les fichiers de configuration et la base de données) sont un bien sensible et doivent être protégés.

Note: Besoin de sécurité des valeurs de configuration :

- intégrité.

57.4.6 Bien sensible B6 : L'authentification

Les données d'authentification à la console d'administration ainsi que les données d'authentification des agents sur le serveur (clé publique de chaque agent WAPT) sont un bien sensible et doivent être protégées.

Note: Besoin de sécurité de l'authentification

- intégrité ;
- confidentialité.

57.5 Description des hypothèses sur l'environment d'exploitation de WAPT

Par définition, les hypothèses sont des déclarations portant sur le contexte d'emploi de WAPT ou de son environnement.

Les hypothèses suivantes sur l'environnement d'exploitation de WAPT doivent être considérées :

57.5.1 Hypothèse H1 : Les Administrateurs et Déployeurs de Paquet WAPT sont formés

Les *Administrateurs* et les *Développeurs de Paquets* sont formés à l'usage de WAPT. En particulier, ils doivent s'assurer que leurs identifiants et clés de sécurité restent secrets.

57.5.2 Hypothèse H2 : Les systèmes subjacents à WAPT sont sains

Les systèmes d'exploitation sur lesquels les agents WAPT s'exécutent mettent en oeuvre des mécanismes de protection adéquats (confinement, contrôle d'accès, etc.) paramétrés et configurés selon les bonnes pratiques.

Les systèmes d'exploitation sont à jour des correctifs en vigueur au moment de l'installation, ils sont sains et exempts de virus, chevaux de Troie, etc.

57.5.3 Hypothèse H3 : Les binaires nécessaires au fonctionnement de WAPT sont intègres

Toutes les bibliothèques et tous les outils nécessaires à l'installation de WAPT sont considérés comme sûrs.

57.5.4 Hypothèse H4 : Les paquets WAPT sont construits de manière sûre

Il est de la responsabilité de l'*Administrateur* de s'assurer que les fichiers destinés à être intégrés dans des paquets WAPT proviennent de sources sûres et sont en particuliers exempts de virus, chevaux de Troie, etc.

Les Administrateurs sont également responsables d'écrire et d'incorporer des scripts `setup.py` sûrs dans les paquets WAPT.

57.5.5 Hypothèse H5 : Les Utilisateurs des postes client ne sont pas Administrateurs Locaux

Un *Utilisateur* n'a pas les droits d'administration de son poste de travail. Sinon l'*Utilisateur* est considéré comme un *Administrateur Local*.

En particulier, l'*Utilisateur* n'a pas les droits d'écriture dans le répertoire d'installation du client WAPT.

57.5.6 Hypothèse H6 : Les Administrateurs Locaux des postes client sont formés

L'*Administrateur Local* d'un poste client doit être formé à l'exploitation de WAPT, ou à défaut ne pas modifier les fichiers d'installation se trouvant dans le dossier d'installation de WAPT.

57.6 Description des menaces pesant sur les biens sensibles WAPT

Par définition, une menace est une action ou un événement susceptible de porter préjudice à la sécurité globale de la machine équipée de WAPT.

Les agents menaçants à considérer pour l'évaluation de sécurité sont les suivants :

- **Entités non-autorisées** : il s'agit d'un attaquant humain ou d'une entité qui interagit avec WAPT mais qui ne dispose pas d'un accès légitime à celui-ci.

Note: Les *Administrateurs* et les *Administrateurs Locaux* ne sont pas considérés comme des attaquants.

Les menaces qui portent sur les biens sensibles WAPT définis ci-dessus sont les suivantes :

57.6.1 Menace M1 : Installation d'un logiciel malveillant par une entité non-autorisée

Cette menace correspond à un attaquant qui parviendrait à utiliser une composante de l'agent WAPT pour installer une application malveillante de façon pérenne, ou pour désinstaller ou désactiver une composante de sécurité du poste sur lequel l'agent WAPT est installé.

57.6.2 Menace M2 : Altération de valeurs de configuration par une entité non-autorisée

Cette menace correspond à un attaquant qui parviendrait à modifier ou à supprimer le paramétrage d'un élément de WAPT défini par un *Administrateur* légitime de WAPT.

57.6.3 Menace M3 : Accès illégitime par une entité non-autorisée

Cette menace correspond à un attaquant qui parviendrait à récupérer les données d'authentification d'un *Administrateur*, à contourner le mécanisme d'authentification de manière à accéder ou à altérer un bien sensible stocké sur le serveur. Elle correspond également à un attaquant qui parviendrait à se faire passer pour un agent WAPT.

57.6.4 Menace M4 : Écoute du réseau par une entité non-autorisée

Cette menace correspond à un attaquant qui parviendrait à intercepter et prendre connaissance des communications réseaux entre les agents et le serveur hébergeant WAPT.

57.6.5 Menace M5 : Altération du trafic réseau par une entité non-autorisée (Type *Man In the Middle*)

Cette menace correspond à un attaquant qui parviendrait à modifier les communications réseaux entre les agents et le serveur hébergeant WAPT ou les communications réseau entre la console et le serveur WAPT.

57.7 Description des fonctions de sécurité de WAPT

Par définition, les fonctions de sécurité sont l'ensemble des mesures techniques et mécanismes mis en œuvre pour protéger de façon proportionnée les biens sensibles contre les menaces identifiées.

57.7.1 Fonction de sécurité F1 : Authentification des contrôles d'accès

Fonction de sécurité F1A : Authentification d'une machine lors de son enregistrement initial dans la base de données WAPT

Nouveau dans la version 1.5.

Note: Les risques évités sont :

- L'inscription d'une machine illégitime dans la base de données.
- Une attaque par déni de service par surcharge de la base de données.
- Eviter l'enregistrement d'un inventaire falsifié dans la base de données.

Solution mise en place

Pour exister dans la base de données et ainsi apparaître dans la console WAPT, une machine doit s'enregistrer auprès du serveur WAPT avec une commande **`register`**.

La commande **`register`** peut être exécutée automatiquement lors de l'installation ou de la mise à jour de l'agent WAPT si la machine est correctement enregistrée avec un compte machine Kerberos dans le domaine Active Directory de l'*Organisation*.

Si la machine ne présente pas au serveur WAPT un ticket kerberos valide, alors la commande **`register`** échoue ;

Lors de l'enregistrement, le poste client génère une paire de clés RSA dans le répertoire privé local et envoie au serveur une demande de signature de certificat avec le ticket kerberos.

Si le ticket Kerberos est valide, le Serveur WAPT enregistre le client dans sa base de données, signe le CSR, stocke le certificat dans sa base de données et le renvoie au client.

Si un appareil est déjà enregistré, il peut se ré-enregistrer sans ticket kerberos en utilisant le certificat signé de son serveur actuel.

Si l'authentification kerberos échoue ou n'est pas disponible, le serveur envoie un statut non autorisé, et le client peut demander à l'administrateur local d'entrer les informations d'identification d'un compte du serveur WAPT ayant le privilège *admin* ou le privilège *register_host*.

L'authentification du compte est effectuée côté serveur en utilisant les mécanismes *admin*, *passwd* ou *ldap*.

Note: La méthode avec Kerberos assume que le serveur Active Directory répond au moment du **`register`**.

Fonction de sécurité F1B : Vérification des certificats HTTPS du serveur par les clients WAPT

Nouveau dans la version 1.5.

Note: Les risques évités (notamment MITM) sont :

- L'envoi d'informations sensibles à un serveur WAPT illégitime et non-autorisé.
- La récupération d'informations sensibles par une entité non-autorisée.
- L'affichage d'informations falsifiées dans la console de l'*Administrateur*.

- Une mauvaise date est envoyée lors d'une requête HEAD (demande de modification de date de fichier) empêchant les futures mise à jours.
- Envoyer le mot de passe de la console à un serveur WAPT illégitime et non-autorisé.

Solution mise en place

Pour fonctionner correctement en version sécurisée :

- Une option de vérification du certificat HTTPS serveur est introduite dans le fichier `C:\Program Files (x86)\wapt\wapt-get.ini` des agents WAPT qui **force la vérification du certificat serveur par les agents WAPT** .
- Une option de vérification du certificat HTTPS serveur est introduite dans le fichier `C:\Program Files (x86)\wapt\wapt-get.ini` des agents WAPT qui **force la vérification du certificat serveur par les agents WAPT**.

L'implémentation technique peut être basée sur deux méthodes :

- Utiliser un utilitaire de vérification de certificat implémenté dans la configuration du service **`Nginx`** du serveur WAPT ; cette méthode est généralement fournie par une *Autorité de Certification* validée pour votre réseau.
- Utiliser la méthode d'"*épinglage de certificat* qui consiste a fournir à l'agent WAPT une liste de certificats de confiance qui sera stockée et maintenue dans le dossier `C:\Program Files (x86)\wapt\ssl\server` .

Fonction de sécurité F1C : Aucun port n'écoute sur les agents WAPT

Nouveau dans la version 1.5.

Note: Les risques évités sont :

- Une entité non-autorisée utilise un port ouvert à mauvais escient.

Solution mise en place

Les connexions vers le serveur WAPT sont exclusivement initiées pas les clients, et les différentes actions instantanées (**`update`** / **`upgrade`** / **`install`** ...) passent au travers d'une connexion permanente par une Websocket initiée par l'agent WAPT.

Note: si HTTPS est activé, l'agent vérifie que la Websocket s'établit bien avec le bon serveur.

Fonction de sécurité F1D : Signature des remontées d'inventaire

Nouveau dans la version 1.3.12.13.

Note: Les risques évités sont :

- Une entité non-autorisée envoie un inventaire falsifié d'une machine existante dans la base de données WAPT.

Solution mise en place

- Au premier **register**, chaque machine crée un couple clé privée / certificat public dans le répertoire `C:\Program Files (x86)\wapt\private` accessible en lecture uniquement aux *Administrateurs Locaux*. Une fois la machine enregistrée, la clé publique est envoyée au serveur WAPT.
- Lors d'une mise à jour de l'inventaire, le nouvel inventaire est envoyé signé avec la clé privée de la machine et déchiffré par la clé publique enregistrée dans la base de données.
- Le serveur refusera de valider tout inventaire signé avec une mauvaise clé.

Fonction de sécurité F1E : Vérification des droits avant l'exécution de certaines actions WAPT

Note: Les risques évités sont :

- Eviter l'exécution de tâches sensibles par des entités non-autorisées.

Solution mise en place

Les *Utilisateurs* interagissent avec WAPT au travers des interfaces WAPT (**wapt-get** en ligne de commande, **wapttray**, **waptexit**, **waptselfservice**).

Les interfaces peuvent ensuite déléguer l'exécution des tâches souhaitées au service WAPT local fonctionnant en compte système.

Les actions qui enclenchent des modifications listées ci-dessous ne nécessitent pas d'authentification auprès du service WAPT :

- `wapt-get update` (mettre à jour la liste des paquets disponibles).
- `wapt-get upgrade` (lancer l'installation des mises à jour en attente).
- `wapt-get download-upgrade` (télécharger les mises à jour en attente).
- `wapt-get clean` (supprimer des paquets restés en cache après installation).
- stopper n'importe quelle tâche WAPT en cours.
- stopper / relancer le service WAPT.

Les autres actions nécessitent que l'"*Utilisateur* s'authentifie et que son compte appartienne au groupe de sécurité Active Directory **waptselfservice** ou que l'"*Utilisateur* soit *Administrateur Local*, exemple d'action :

- `wapt-get install` : ordonner à l'agent WAPT d'installer un paquet WAPT marqué **MISSING** sur la machine.
- `wapt-get remove` : ordonner à l'agent WAPT de supprimer un paquet WAPT.
- `wapt-get forget` : ordonner à l'agent WAPT d'oublier l'existence d'un paquet WAPT installé sur la machine sans le désinstaller.

57.7.2 Fonction de sécurité F2 : Protection de l'intégrité du processus d'installation des paquets WAPT

Fonction de sécurité F2A : Signature des paquets WAPT

Note: Les risques évités sont :

- Pour éviter qu'une entité non-autorisée modifie le contenu ou le comportement d'un paquet WAPT.

Solution mise en place

- Quand un *Administrateur* ou un *Développeur de Paquet* construit un paquet WAPT, un fichier `WAPTmanifest.sha256` est créé qui liste les sommes de contrôle de tous les fichiers du paquet.
- Un fichier `signature.sha256` **chiffré** avec la clé privée est ensuite créé dans le dossier `WAPT`, il contient la somme de contrôle du fichier `WAPTmanifest.sha256`.
- L'ensemble est archivé avec l'extension *`.wapt`*.
- Quand un agent WAPT télécharge un paquet WAPT, l'agent vérifie que le fichier `signature.sha256` a été signé avec la clé privée qui correspond au certificat présent dans le dossier `WAPT`.
- L'agent WAPT vérifie ensuite que le certificat (ou la chaîne de certificat) `certificate.crt` a bien été signé avec une clé privée correspondant à un des certificats présents dans le dossier `C:\Program Files (x86)\wapt\ssl`.
- L'agent WAPT fait ensuite la somme de contrôle de tous les fichiers du paquet (excepté les fichiers `signature.sha256` et `certificate.crt`) et vérifie que cela correspond au fichier `WAPTmanifest.sha256` contenu dans le paquet.
- Si l'une de ces étapes n'est pas validée, alors cela signifie qu'un fichier a été modifié/ ajouté/ supprimé. Alors, l'exécution du **`setup.py`** est annulée.
- Le paquet en défaut est ensuite supprimé du cache local ; l'évènement est rapporté dans les log de l'agent.

Fonction de sécurité F2B : Signature des attributs du fichier *control*

Nouveau dans la version 1.4.

Note: Les risques évités sont :

- Une entité non-autorisée modifie des dépendances WAPT sur la machine en falsifiant le fichier `https://waptserver/wapt/Packages`.

Solution mise en place

Lors de la signature d'un paquet WAPT, les attributs sensibles du paquet sont listés dans l'attribut **signed_attributes**.

Note: Exemple d'une liste *signed_attributes* :

package, *version*, *architecture*, *section*, *priority*, *maintainer*, *description*, *depends*, *conflicts*, *maturity*, *locale*, *min_os_version*, *max_os_version*, *min_wapt_version*, *sources*, *installed_size*, *signer*, *signer_fingerprint*, *signature_date*, *signed_attributes*,

---

Les attributs listés dans *signed_attributes* sont signés avec la clé privée de l'*Administrateur* et la signature est stockée dans l'attribut *signature* du fichier `control`.

Le certificat associé à cette clé privée est stocké dans le fichier `WAPT\certificate.crt` à l'intérieur du paquet WAPT.

Sur le serveur WAPT, lors de l'opération **`wapt-scanpackages`** (déclenchée par un ajout ou suppression de paquet), l'index `Packages` des paquets est régénéré.

Le serveur WAPT extrait de chaque paquet le certificat du signataire et l'ajoute dans le fichier ZIP `Packages`, dans le répertoire `ssl`. Chaque certificat est nommé avec sa fingerprint encodée en hexadécimal.

Lorsque le client WAPT effectue un **`update`** (mise à jour des paquets disponibles), il télécharge le fichier index `Packages`, qui contient à la fois les attributs signés de tous les paquets et les certificats des signataires.

Si le certificat du signataire des attributs d'un paquet est approuvé (ce qui signifie que ce certificat est signé par une *Autorité de Certification* ou que le certificat lui-méme est de confiance), **ET** que le certificat du signataire peut vérifier la signature des attributs, le paquet est ajouté à l'index des paquets disponibles, sinon il est ignoré.

Fonction de sécurité F2C : Restriction d'accès au répertoire d'installation de l'agent WAPT

Note: Les risques évités sont :

- Une entité non-autorisée modifie le comportement de l'agent WAPT.

Le répertoire d'installation `C:\Program Files (x86)\wapt` est accessible en lecture et modification :

- Aux *Administrateurs Locaux* à travers un accès local au répertoire d'installation de l'agent WAPT.
- Aux *Administrateurs* à travers le mécanisme de déploiement des mises à jour de l'agent WAPT.

Ni les *Développeurs de Paquets*, ni les *Utilisateurs* n'ont d'accès en écriture au répertoire d'installation de l'agent WAPT.

Les restrictions d'accès au dossier `wapt` reposent sur le mécanisme ACL standard du système d'exploitation et sont appliquées pendant l'installation de l'agent WAPT.

Fonction de sécurité F2D : Restriction totale d'accès au répertoire de stockage du couple clé privé / certificat de signature d'inventaire

---

Note: Les risques évités sont :

- Une entité non-autorisée falsifie une remontée d'inventaire.
- Une entité non-autorisée usurpe l'identité d'une machine avec WAPT.

Aucun droit d'accès au répertoire `C:\Program Files (x86)\wapt\private` n'est accordé à aucun *Utilisateur*, quel qu'il soit. Seul l'agent WAPT a accès en lecture et écriture à ce répertoire.

Note: Le stockage du couple clé privée / certificat découle d'un choix technique qui consiste à dire que la machine détient seule toutes les informations qui la concernent.

57.7.3 Fonction de sécurité F3 : Sécurisation des communications entre les différents composants WAPT

Fonction de sécurité F3A : Signature des requêtes envoyées aux agents WAPT

Nouveau dans la version 1.5.

Note: Les risques évités sont :

- Une entité non-autorisée envoie des requêtes falsifiées aux agents WAPT.

Solution mise en place

Les commandes suivantes, entre autres, sont signées par la Console WAPT avant d'être envoyées aux agents WAPT ciblés via le Serveur WAPT et les Websockets :

- `wapt-get install` : demande à l'agent WAPT d'installer un paquet WAPT marqué **MISSING** sur la machine.
- `wapt-get remove` : demander à l'agent WAPT de supprimer un paquet WAPT.
- **`trigger package forget`** : demande à l'agent WAPT d'oublier l'existence d'un paquet WAPT précédemment installé sans supprimer le logiciel ou la configuration.
- `wapt-get update-status` : demander à l'Agent WAPT de renvoyer l'état de son inventaire actuel au Serveur WAPT.
- `wapt-get upgrade`: demander à l'Agent WAPT d'exécuter les paquets marqués **NEED UPGRADE**.
- **`trigger host update`** : demander à l'agent WAPT de mettre à jour la liste des paquets disponibles, et de vérifier si le client doit installer, mettre à niveau ou supprimer les paquets WAPT, en fonction de l'arbre des dépendances des paquets WAPT.

Tous les attributs des demandes d'action immédiate sont signés :

- l'*UUID* du poste ;
- l'action (ex: **`install`**) ;

- les arguments (ex: tis-firefox) ;
- l'horodatage des requêtes.

Le certificate associé à la signature est également passé :

- A la réception d'une requête par l'agent WAPT, il vérifie que la requête est correctement signée.
- L'agent vérifie ensuite que la date fournie en argument ne dépasse pas une minute de décalage.
- Enfin, l'agent vérifiera enfin que le certificat associé est autorisé à lancer des commandes.

57.8 Présentation des processus cryptographiques serveur

L'authentification kerberos est :

- *admin* : le nom d'utilisateur et la dérivation du hachage pbkdf2-sha256 du mot de passe sont stockés dans la configuration du Serveur WAPT. Il s'agit du principal mécanisme d'authentification de l'Administrateur WAPT.
- *passwd* : noms d'utilisateurs secondaires supplémentaires et hash des mots de passe sont stockés dans un fichier de style `htpasswd` côté serveur.
- *ldap* : si le hash passwd n'est pas défini dans le fichier `htpasswd`, un serveur ldap peut être déclaré et utilisé comme mécanisme d'authentification pour les comptes secondaires du serveur wapt. Les comptes ldap valides sont définis par un DN de base ldap et une liste ldap de groupes.
- *session* : lors de l'authentification initiale à l'aide d'un des mécanismes *admin*, *passwd* ou *ldap*, un cookie de session peut être utilisé comme substitut à l'authentification ultérieure du serveur. Le cookie de session a une durée de vie par défaut de 12h.

57.9 Présentation des processus cryptographiques serveur

- Après une authentification réussie, le serveur stocke les privilèges associés dans les sessions des clients.
- Dans le contexte de la cible d'évaluation actuelle, seul le privilège *admin* est évalué.
- Le privilège *admin* est un alias pour tous les privilèges. Une fois authentifié, l'utilisateur a accès à tous les points d'extrémité disponibles du Serveur WAPT.
- S'il n'est pas authentifié, l'Utilisateur a toujours accès au dépôt des paquets, qui est public par conception.

57.10 Matrices de couvertures

57.10.1 Menaces et biens sensibles

La matrice suivante présente la couverture des menaces sur les biens sensibles (les lettres **D**, **I**, **C** et **A** représentent respectivement les besoins de **Disponibilité**, **Intégrité**, **Confidentialité** et **Authenticité**) :

Table1: Couverture des biens sensibles par les menaces

	B1. Communications	B2. Données d'inventaire	B3. Logs	B4. Configuration	B5. Postes clients	B6. Authentification
M1. Installation de logiciel mlaveillant	I,C				I	
M2. Altération de configuration				I		
M3. Accès illégitime		I,C	D	I,C	I	I,C
M4. Ecoute réseau	C	C				
M5. Altération réseau	I,A	I,A				

57.10.2 Menaces et fonctions de sécurité

La matrice suivante présente la couverture des menaces par les fonctions de sécurité :

Table2: Couverture des menaces par les fonctions de sécurité

	F1. Authentification des contrôles d'accès	F2. Protection des données	F3. Communications sécurisées	F4. Signature des paquets
M1. Installation de logiciel mlaveillant	✓	✓		✓
M2. Altération de configuration	✓	✓		
M3. Accès illégitime	✓			
M4. Ecoute réseau			✓	
M5. Altération réseau			✓	✓

CHAPTER 58

Présentation des processus cryptographiques

Date	avr. 17, 2023
Rédacteur	Hubert TOUVET, Vincent Cardon
Applicable pour WAPT	>= 2.3.0.13180
Version du document	2.3.0.0-0
Hash Git	368951c0f29223af79a7ad6f74d01726029c12fa

- *Répertoires et fichiers référencés dans ce document*
- *Définition des Acteurs*
- *Synthèse des modules crypto mis en oeuvre par la solution WAPT*
- *Types d'infrastructures PKI / CA dans une configuration WAPT standard*
- *Gestion des clés et des certificat de l'Administrateur*
  - *Validité du certificat de l'Administrateur*
  - *Autoriser le certificat de l'Administrateur à signer un paquet*
- *Gérer les clés et certificats du Client WAPT*
  - *Émission initiale et mise à jour du certificat du client WAPT*
  - *Déployer les certificats d'autorité pour vérifier les paquets et les actions sur les clients*
  - *Déployer les certificats d'autorité pour la communication HTTPS entre les clients WAPT et le serveur WAPT*
- *Communications HTTPS entre les clients WAPT et les dépôts WAPT*
  - *Déployer des certificats d'autorité*
  - *Communications Websockets entre les clients WAPT et le serveur WAPT*
- *Communications entre la console WAPT et le serveur WAPT*

  - *Déployer des certificats d'autorité*
  - *Déployer des certificats d'autorité pour vérifier les paquets importés dans le dépôt local*
- *Processus de signature d'un paquet WAPT*
  - *Paramètres initiaux*
  - *Signature des attributs du fichier control*
  - *Signature des fichiers du paquet*
- *Vérifier la signature des attributs d'un paquet*
- *Vérifier la signature d'un paquet*
- *Signature d'une action immédiate*
  - *Processus de signature pour des actions immédiates*
  - *Vérifier la signature d'une action immédiate*
- *Vérification du téléchargement complet d'un paquet*

Les processus cryptographiques sont utilisés dans les activités suivantes :

- Signature et vérification des **fichiers contenus dans un paquet**.
- Signature et vérification des **attributs d'un paquet**.
- Signature et vérification des **actions immédiates** sur les clients WAPT.
- Signature des inventaires et **statut des clients WAPT**.
- Authentification de la connexion Websockets des clients WAPT sur le Serveur WAPT.
- Communication HTTPS entre les clients WAPT et le Serveur WAPT.
- Communications HTTPS entre la console WAPT et le serveur WAPT.
- Communications HTTPS entre les clients WAPT et les dépôts WAPT.

58.1 Répertoires et fichiers référencés dans ce document

- <WAPT> : répertoire d'installation de WAPT. Par défaut `%Program Files (x86)%\WAPT`.
- <WAPT>\wapt-get.ini : fichier de configuration du client WAPT (**wapt-get** et **waptservice**).
- <WAPT>\ssl : répertoire par défaut pour les certificats de confiance pour signer les paquets et les actions.
- <WAPT>\ssl\server : répertoire par défaut pour stocker les certificats https du serveur WAPT (épinglage de certificat).
- <WAPT>\private : répertoire par défaut pour les certificats permettant de signer l'inventaire et les connexions Websocket.
- %LOCALAPPDATA%\waptconsole\waptconsole.ini : fichier de configuration de la console WAPT et des actions de développement de l'outil **wapt-get**.
- %aAPPDATA%\waptconsole\ssl : répertoire par défaut pour les certificats de confiance pour l'import de paquets WAPT depuis un dépôt externe (c.à.d. les *modèles de paquets*).

58.2 Définition des Acteurs

- **Organisation**

  Une Organisation est le périmètre de responsabilité dans lequel est exploitée la solution WAPT.

- **Autorité de Certification**

  Un certificat d'Autorité est l'entité qui détient les clés qui ont signé les certificats des *Développeurs de Paquets*, des *Déployeurs de Paquets* et des serveurs HTTPS.

- **Administrateurs**

  Les Administrateurs sont en possession d'une clé RSA personnelle et d'un certificat signé par l'"*Autorité de Certification* de l'"*Organisation* ; ils ont aussi un identifiant et un mot de passe pour accéder à la console WAPT.

- **Agent WAPT**

  Les clients WAPT sont l'ensemble des appareils que les *Administrateurs* de l'"*Organisation* peuvent gérer avec WAPT. Les clients **peuvent être ou non un membre** du dmaine Active Directory de l'"*Organisation*.

- **Serveur WAPT**

  Le serveur WAPT est un serveur Linux / Nginx / PostgreSQL de l'"*Organisation* qui gère l'inventaire et le status des Postes clients WAPT.

  Par défaut, le serveur WAPT joue également le rôle de dépôt WAPT interne. Le serveur WAPT a un compte ordinateur dans l'Active Directory de l'"*Organisation*.

- **Dépôts WAPT internes**

  Les dépôt internet WAPT sont un ou plusieurs serveur Linux / Nginx qui diffusent aux Postes clients WAPT en HTTPS des paquets WAPT signés.

- **Dépôts WAPT externes**

  Les dépôts WAPT externe sont des dépôts WAPT publics que les *Développeurs de Paquets* peuvent utiliser pour importer des paquets conçus par d'autres *Organisations*, sous condition d'en vérifier l'adéquation aux normes internes de sûreté et de sécurité ;

- **Serveur Active Directory**

  Le serveur Active Directory gérant le domaine AD de l'"*Organisation*;

58.3 Synthèse des modules crypto mis en oeuvre par la solution WAPT

L'Agent WAPT, le Serveur WAPT et le Console WAPT utilisent à la fois du code Python interprété et du code compilé Lazarus / FPC.

Le code Lazarus / FPC utilise le framework mORMot (>= 2.0.4383) pour la gestion du protocole https, la gestion du client kerberos, les opérations sur les certificats X509, et les opérations cryptographiques (Hash, RSA).

Le framework mORMot est lui-même configuré et lié pour utiliser la bibliothèque **`OpenSSL 1.1.1s 1 Nov 2022`** pour les sockets TLS, les opérations RSA asymétriques (génération de clé, chiffrement, déchiffrement, signature, vérification), et les opérations de certificats X509.

Le code Python (Agent WAPT, Serveur WAPT, paquets WAPT) est lié aux mêmes bibliothèques **`OpenSSL 1.1.1s 1 Nov 2022`** et utilise les modules suivants :

Du côté de l'Agent WAPT :

- les modules python **cryptographie==3.3.2** et **pyOpenSSL==20.0.1** liés à **openssl 1.1.1s** : utilisés pour toutes les opérations cryptographiques RSA, les générations de certificats X509 et les vérifications de signatures dans l'agent WAPT en python.
- **winkerberos==0.8.0** (agent Windows) / **kerberos==1.3.1** (agent Linux) et **requests-kerberos==0.12.0** : utilisés pour authentifier le client WAPT lors de son premier enregistrement avec le Serveur WAPT.
- **certifi==2021.5.30** : utilisé comme base pour les certificats des Autorités Racine.

Du coté du Serveur WAPT :

- Module ssl **Python** 3.8.16 lié à **openssl 1.1.1s**.
- **cryptography==3.3.2** et **pyOpenSSL==20.0.1** liés sur **openssl 1.1.1s** : utilisés pour toutes les opérations cryptographiques RSA telles que les générations de clés, les générations de certificats X509 et les vérifications de signatures.

Sur le Serveur WAPT, **nginx/1.18.0** est configuré pour servir les paquets par https, gérer les demandes https de l'API du Server WAPT, gérer les authentifications Kerberos des clients et la vérification des certificats des clients. Le serveur nginx est configuré pour TLS1.2, cipher "EECDH+AESGCM:EDH+AESGCM:AES256+EECDH:AES256+EDH".

58.4 Types d'infrastructures PKI / CA dans une configuration WAPT standard

Il y a trois types de PKI / CA dans une configuration WAPT standard

Table1: Types de PKI / CA dans une configuration WAPT

Type	Utilisation	Origine et enregistrement	Renouvellement	Révocation
Certificat de transport HTTPS	L'objectif de ce certificat est de sécuriser la communication entre l'Agent WAPT et le Serveur WAPT et pour authentifier le Serveur WAPT.	Le certificat est délivré par une *Autorité de certification* de confiance pour l'ordinateur client (la *Autorité de Certification* doit être intégrée dans le Certificate Store local), et la clé privée https doit être stockée sur le serveur http **nginx**.	Le certificat doit être renouvelé comme tous les certificats https gérés par l'*Organisation*.	Le certificat doit être révoqué comme tous les certificats https gérés par l'*Organisation*.
Le certificat de l'Administrateur / Packager	L'objectif de ce certificat est de signer les paquets et les messages.	Ce certificat est émis par l'*Autorité de certification* de l'*Organisation*, et doit être stocké dans le répertoire `<WAPT>\ssl` de chaque Agent WAPT qui doit faire confiance à ce certificat.	Le certificat doit être renouvelé selon la procédure standard de l'*Autorité de certification* de l'*Organisation*.	La CRL doit être publiée via http et accessible par le serveur WAPT, et l'attribut du point de distribution de la CRL doit être défini dans le certificat. Le Serveur WAPT redistribue la CRL à l'Agent WAPT.
Le certificat du client de l'Agent WAPT	L'objectif de ce certificat est d'authentifier l'Agent WAPT auprès du serveur WAPT, de signer les données et enfin de chiffrer éventuellement les données que l'Agent WAPT envoie au Serveur WAPT. Ce certificat est purement technique pour identifier un client WAPT et est créé par un processus CSR initié par le client lors de l'enregistrement de l'Agent WAPT auprès du Serveur WAPT. La clé privée est déposée sur le client dans `<WAPT>\ssl\private`, et la clé publique est déposée sur le Serveur WAPT.	Le *Autorité de certification* est géré en interne par le Serveur WAPT et est utilisé uniquement pour l'authentification des clients avec le Serveur WAPT.	Le renouvellement se fait lors de l'enregistrement ou bien lorsque la vérification du client échoue.	La révocation est gérée en interne par le Serveur WAPT qui configure une CRL pour le service **Nginx**.

58.5 Gestion des clés et des certificat de l'Administrateur

Les paquets et actions de l'*Administrateur* sont signés pour n'autoriser que les Administrateurs de confiance à intervenir sur les postes.

L'*Administrateur* de la solution WAPT a en sa possession :

- Une clé privée *RSA* de 2048 bits chiffrée par l'algorithme aes-256-cbc.
- Un certificat *X509* signé par une *Autorité de Certification* approuvée par l'*Organisation*.

Note: Le processus d'émission de ces clés, la signature du certificat, la distribution et la révocation sont à la charge de l'*Organisation* utilisant WAPT et sortent donc du périmètre fonctionnel de WAPT.

Cependant, pour facilement tester la solution, WAPT propose une fonction pour générer une clé RSA et un certificat X509 :

- La clé RSA générée est de 2048 bits, chiffrée par l'algorithme aes-256-cbc et encodée en format PEM avec l'extension *.pem*.
- Le certificat est soit autosigné, soit signé par une Autorité de Confiancedont on a à disposition la clé et le certificat en format PEM.
- Si le certificat est autosigné, son attribut *KeyUsage* comporte le flag *keyCertSign*.
- Si l'*Administrateur* est habilité par l'*Organisation* à signer des paquets contenant du code python (présence du fichier `setup.py`), l'attribut du certificat *extendedKeyUsage* comporte le flag **CodeSigning**.
- Le certificat *X509* est encodé et remis à l'*Administrateur* en format PEM avec l'extension *.crt* .

58.5.1 Validité du certificat de l'Administrateur

L'Agent WAPT fait confiance à tous les certificats X509 non expirés situés dans le répertoire <wapt>\ssl de l'agent WAPT.

Si un fichier *.crt* encodé en format PEM contenu dans le répertoire `<wapt>\ssl` de l'Agent WAPT contient plusieurs certificats, seul le premier est reconnu.

l'agent WAPT vérifie uniquement les dates de validité (attributs `notValidBefore`/ `notValidAfter`). Le certificat est valide si (`Now` >= `notValidBefore` et `Now` <= `notValidAfter`").

58.5.2 Autoriser le certificat de l'Administrateur à signer un paquet

Le certificat utilisé par la console WAPT pour signer les paquets et actions est défini avec le paramètre *personal_certificate_path* de la section `[global]` du fichier `%LOCALAPPDATA%\waptconsole\waptconsole.ini`.

WAPT demande à l'*Administrateur* son mot de passe pour permettre de rechercher la clé privée (encodée au format PEM) correspondant au certificat parmi les fichiers `.pem` du répertoire contenant les certificats.

Lors de la signature de paquet, WAPT refusera le certificat si le paquet contient un fichier `setup.py` et que le certificat n'est pas de type *CodeSigning*.

58.6 Gérer les clés et certificats du Client WAPT

Le client WAPT (**`waptservice`**) utilise des clés RSA et un certificat X509 pour interagir avec le Serveur WAPT.

Le certificat du client WAPT est utilisé dans les situations suivantes :

- Pour contrôler l'accès aux paquets *hosts* et aux autres dépôts de paquets lorsque l'authentification par certificat client est activée dans la configuration du serveur NGINX.
- Lors de la mise à jour du statut du poste sur le serveur. (**`update_server_status`**) : **signature des informations**.
- Lors de la connexion Websocket du poste vers le serveur (**`waptservice`**) : **signature de l'UUID du poste**.

58.6.1 Émission initiale et mise à jour du certificat du client WAPT

- A l'issu du processus d'installation de l'agent WAPT sur le Poste client, l'agent WAPT s'enregistre automatiquement auprès du serveur WAPT en émettant une requête https authentifiée par Kerberos qui utilise le TGT du compte machine.

  L'agent WAPT utilise les API kerberos de Windows en s'appuyant sur les modules python **kerberos-sspi** et **requests-kerberos** .

Note: Cette procédure fonctionne si et seulement si le poste client est joint au domaine Windows pour lequel le Serveur WAPT est configuré.

Si la clé et les certificats n'ont pas encore été générés, ou s'ils ne correspondent pas au *FQDN* actuel de la machine, l'agent WAPT génère une clé RSA et un certificat X509 autosigné avec les paramètres suivants :

- La clé est de type RSA 2048 bits encodée en PEM et stockée dans le fichier <WAPT>\private\<device UID>.pem.
- Le certificat généré a les attributs suivants :
  - Sujet.COMMON_NAME = <device UID>.
  - *Subject.ORGANIZATIONAL_UNIT_NAME* = nom de l'*Organisation* enregistré la base de registre du client Windows.
  - *SubjectAlternativeName.DNSName* = <device UID>.
  - *BasicConstraint.CA* = True.
  - Validity = 10 years.
  - Serialnumber = random.
- Le certificat est sauvegardé dans le fichier <WAPT>\private\<device UID>.crt.

  Note: Seul le compte machine et les *Administrateurs Locaux* ont accès au répertoire <WAPT>\private car des ACL spécifiques sont appliquées à l'installation de l'agent WAPT sur le poste.

- L'inventaire ou les mises à jour de statuts du client sont envoyés au serveur WAPT par requête HTTPS POST.
- On authentifie la requête https POST en ajoutant deux headers http spécifiques :
- *X-Signature* :
  - Encodage en JSON des informations BLOB d'inventaire ou de status.
  - Signature du *json* avec la clé privée du client WAPT : hachage *sha256* et padding *PKCS#1 v1.5*.
  - encodage de la signature en *base64*.
- *X-Signer* : *Subject.COMMON_NAME* du client WAPT.
- Après avoir initialement authentifié le client WAPT avec kerberos, le Serveur WAPT reçoit la demande de signature de certificat envoyée par le client, signe un certificat client approprié et le stocke dans la table *hosts* de son inventaire au format *PEM* (colonne *host_certificate*).
- Le certificat client nouvellement signé est envoyé au client.
- Le certificat est sauvegardé dans le fichier <WAPT>\private\<device UID>.crt.

Note: Si le poste client WAPT est renommé, la clé privée et le certificat sont recréés.

Lors de la tentative de mise à jour de status du client vers le serveur, la requête POST sera refusée, car la machine est enregistrée dans la base de données avec un autre certificat.

La machine tentera alors de se ré-enregistrer (**register**) avec une authentification kerberos ; ainsi le nouveau certificat sera enregistré dans la base de données.

58.6.2 Déployer les certificats d'autorité pour vérifier les paquets et les actions sur les clients

Les certificats formatté en PEM sont stockés dans des fichiers avec des extensions *.crt* or *.pem* dans le répertoire défini avec le paramètre public_certs_dir dans la fichier <WAPT>\wapt-get.ini. Ils sont réputé pour être des **certificats de confiance**.

Ce paramètre public_certs_dir est initialisé par défaut à <WAPT>\ssl.

Le déploiement de ces certificats d'autorité est effectué lors de l'installation initiale de l'agent WAPT par l'installeur.

Depuis la console, l'*Administrateur* compile un installeur personnalisé en vue de son déploiement par *GPO* sur les postes clients.

La Console WAPT incorpore dans cet installeur les certificats présents dans le répertoire <WAPT>\ssl du poste depuis lequel l'installeur est compilé.

L'*Administrateur* doit s'assurer d'enregistrer dans <WAPT>\ssl uniquement les certificats d'autorité nécessaires avant de lancer la compilation de l'agent.

Le déploiement ou la mise à jour de certificats de l'*Autorité de Certification* pour la vérification des paquets et actions peuvent être également assurés à postériori par une GPO Active Directory ou par un paquet WAPT.

58.6.3 Déployer les certificats d'autorité pour la communication HTTPS entre les clients WAPT et le serveur WAPT

Le service WAPT ainsi que l'outil en ligne de commande **wapt-get** communiquent avec le serveur WAPT pour envoyer l'inventaire (**register**) et le statut de déploiement des paquets (**update-status**).

Ces deux types de connexions vérifient le certificat https du serveur.

Paramètre verify_cert de la section [global] du fichier <WAPT>\wapt-get.ini :

- verify_cert = True ou 1

  Cette méthode ne fonctionnera bien que si le Serveur WAPT HTTPS est configuré pour renvoyer son certificat et les certificats intermédiaires à l'initialisation de communication TLS.

- *verify_cert* = <chemin vers fichier .pem>

  Cette méthode vérifie le certificat HTTPS du serveur WAPT en utilisant le bundle de certificats indiqué. Tous les certificats de CA intermédiaires et root doivent être rassemblés dans un fichier au format *.pem* ;

- verify_cert = False ou 0

  Cette méthode ne vérifie pas le certificat HTTPS du serveur WAPT.

Conventionnellement, on stocke le bundle de l'*Autorité de Certification* approuvé dans le répertoire <WAPT>\ssl\server.

La Console WAPT comporte une fonction pour faciliter la récupération initiale de la chaîne de certificats du serveur et pour la stocker au format *.pem* dans le fichier <WAPT>\ssl\server\<FQDN serveur>.pem.

Il est de la responsabilité de l'*Administrateur* de s'assurer que la chaîne ainsi récupérée est authentique.

Lors de la compilation de l'installeur de l'agent WAPT, les certificats ou le bundle de certificats sont intégrés dans l'installeur.

Lors du déploiement de l'installeur sur les clients WAPT, le bundle est copié dans <WAPT>\ssl\server et le paramètre verify_cert de la section [global] du fichier <WAPT>\wapt-get.ini est renseigné pour désigner le bundle.

58.7 Communications HTTPS entre les clients WAPT et les dépôts WAPT

58.7.1 Déployer des certificats d'autorité

Les connexions HTTPS de l'agent WAPT vers le dépôt principal utilisent les mêmes méthodes que les communications entre l'agent WAPT et le serveur WAPT.

L'agent WAPT utilise le même bundle de certificats pour communiquer en HTTPS avec le dépôt principal, avec le serveur WAPT, et avec les dépôts secondaires.

La connexion https est mise en œuvre par les modules python **requests**, **urllib3** et **ssl**.

Le certificat transmis par le serveur HTTPS du dépôt est vérifié par le module **urllib3.contrib.pysopenssl.PyOpenSSLContext** et **urllib3.util.ssl_wrap_socket**.

58.7.2 Communications Websockets entre les clients WAPT et le serveur WAPT

Pour permettre des actions immédiates sur les clients WAPT, le service WAPT déployé sur les clients tente d'établir et de maintenir une connexion Websocket vers le serveur WAPT.

Cette connexion s'effectue sur un connexion chiffrée avec le protocole TLS et utilise côté client le même bundle de certificat que la connexion HTTPS Client vers Serveur WAPT.

58.8 Communications entre la console WAPT et le serveur WAPT

58.8.1 Déployer des certificats d'autorité

Le paramètre verify_cert de la section [global] du fichier %LOCALAPPDATA%\waptconsole\waptconsole.ini peut avoir plusieurs valeurs :

- verify_cert = True ou 1

  Cette méthode ne fonctionnera bien que si le Serveur WAPT HTTPS est configuré pour renvoyer son certificat et les certificats intermédiaires à l'initialisation de communication TLS.

- *verify_cert* = <chemin vers fichier .pem>

  Cette méthode vérifie le certificat HTTPS du serveur WAPT en utilisant le bundle de certificats indiqué. Tous les certificats de CA intermédiaires et root **DOIVENT** être rassemblés dans un fichier au format *.pem*.

- verify_cert = False ou 0

  Cette méthode ne vérifie pas le certificat HTTPS du serveur WAPT.

Conventionnellement, on stocke le bundle de l'"*Autorité de Certification* approuvé dans le répertoire <WAPT>\ssl\server.

La Console WAPT comporte une fonction pour faciliter la récupération initiale de la chaîne de certificats du serveur et la stocker au format *.pem* dans le fichier <WAPT>\ssl\server\<FQDN serveur>.

Il est de la responsabilité de l'"*Administrateur* de s'assurer que la chaîne ainsi récupérée est authentique.

Il est également possible de récupérer la chaîne de certificats du serveur et de renseigner le paramètre `verify_cert` avec la commande :command:`wapt-get enable-check-certificate`.

58.8.2 Déployer des certificats d'autorité pour vérifier les paquets importés dans le dépôt local

Dans la console WAPT / onglet *Dépôt privé*, un bouton *Importer depuis internet* permet de télécharger un paquet depuis un dépôt externe dont l'URL est fournie par le paramètre `repo-url` de la section `[wapt_templates]` du fichier `%LOCALAPPDATA%\waptconsole\waptconsole.ini`.

Une case à cocher *Vérifier la signature de paquet* permet de s'assurer que le paquet est signé avec un certificat provenant d'une Autorité de confiance.

Les certificats d'autorité présents dans le répertoire désigné par le paramètre `public_certs_dir` de la section `[wapt_templates]` du fichier `%LOCALAPPDATA%\waptconsole\waptconsole.ini` sont réputés de confiance.

Si le paramètre n'est pas mentionné explicitement, il est initialisé à `%APPDATA%\waptconsole\ssl`.

Ce répertoire n'est pas automatiquement alimenté par WAPT. Il incombe à l'*Administrateur* de copier / coller dans celui-ci les fichiers `PEM` d'autres *Administrateurs* de confiance ou les certificats d'Autorités de Certification de confiance.

Les certificats d'autorité sont encodés en format PEM et stockés dans des fichiers avec l'extension *.pem* ou *.crt*. On peut stocker plusieurs certificats dans chaque fichier `.crt` ou `.pem`.

Il n'est pas nécessaire d'avoir la chaîne complète de certificats, WAPT acceptera le signataire d'un paquet à partir du moment que :

- Le certificat du paquet est également présent dans le répertoire `public_certs_dir`. Le test d'égalité est fait avec l'empreinte du certificat ;
- Le certificat de l'Autorité ayant signé le certificat du paquet est présent dans le répertoire `public_certs_dir`. La recherche est faite avec soit l'attribut `issuer_subject_hash`, soit l'attribut `authority_key_identifier` du certificat. La signature du certificat est vérifiée par la classe **`x509.verification.CertificateVerificationContext`** ;

58.9 Processus de signature d'un paquet WAPT

Le processus de signature du paquet est lancé lors des actions suivantes :

- Action **`wapt-get.exe build-upload <directory>`**.
- Action **`wapt-get.exe sign-package <path-to-package-file.wapt>`**.
- Sauvegarder un paquet *host* dans la Console WAPT.
- Editer ou sauvegarder un paquet dans la console WAPT.
- Importer un paquet depuis un dépôt externe.
- Créer un paquet avec l'assistant de configuration.

58.9.1 Paramètres initiaux

- Fichier ZIP du paquet WAPT.
- Clé privée RSA du signataire encodée en format `.pem` et chiffrée (par l'algorithme *aes-256-cbc* de OpenSSL si la clé a été créée dans la console WAPT).
- Certificat *X509* du signataire correspondant à la clé privée.
- Si le paquet à signer contient un fichier `setup.py`, le certificat *X509* **DOIT** avoir l'extension *advanced Key Usage* : *codeSigning (1.3.6.1.5.5.7.3.3)* ;

58.9.2 Signature des attributs du fichier control

Le fichier `control` d'un paquet décrit les métadonnées du paquet, en particulier son nom, sa version, ses dépendances et ses conflits. C'est la fiche d'identité du paquet.

Ces métadonnées sont primitivement utilisées par l'agent WAPT pour déterminer si un paquet doit être mis à jour, et quels autres paquets doivent être installés ou désinstallés préalablement.

Ces informations sont donc signées pour garantir aux Postes client leur intégrité et leur authenticité.

Etapes du processus :

- Les attributs `signed_attributes`, `signer`, `signature_date`, `signer_fingerprint` sont ajoutés à la structure du fichier `control`:
  - `signed_attributes` : liste séparée par des virgules des noms des attributs pris en compte dans la signature ;
  - `signer` : Nom commun du détenteur du certificat pour information ;
  - `signature_date` : date et heure en cours (UTC) sous la forme "%Y-%m-%dT%H:%M:%S" ;
  - `signer_fingerprint`: : empreinte `sha256` encodée en hexadécimal du certificat du signataire obtenue avec la fonction **`fingerprint`** incluse dans la classe **`cryptography.x509.Certificate`**.
- Les attributs signés de la structure de contrôle sont encodés en JSON, sans espace ni saut de ligne, et triés par ordre alphabétique.
- Le JSON BLOB résultant est signé avec un hachage *sha256* et un remplissage *PKCS#1 v1.5*.
- La signature est encodée en base64 et stockée dans le JSON dans l'attribut `signature` du fichier `control`.

58.9.3 Signature des fichiers du paquet

- Les attributs du fichier control sont signés et sérialisés en *JSON*. Le résultat est stocké dans le fichier <WAPT>\control du paquet WAPT.
- Le certificat X509 sérialisé PEM du détenteur du certificat est stocké dans le fichier <WAPT>\certificate.crt du paquet WAPT.
- Les empreintes *sha256* de tous les fichiers contenus dans le paquet WAPT sont encodées en hexadécimal et stockées sous la forme d'une liste JSON [(nom du fichier, hachage),] dans le fichier <WAPT>\manifest.sha256 du paquet WAPT.
- Le contenu du fichier <WAPT>\manifest.sha256 est signé avec la clé privée de l'*Administrateur* (clé RAS 2048 bits), avec un hachage *sha256* et un remplissage *PKCS#1 v1.5* :
  - La procédure de signature fait appel à la fonction sign de la classe **`cryptography.rsa.RSAPrivateKey.signer`**.
  - **`cryptography.rsa.RSAPrivateKey.signer`** repose sur les fonctions OpenSSL de **`EVP_DigestSignInit`**.

- La signature est encodée en base64 et stockée dans le fichier <WAPT>\signature.sha256 du paquet WAPT.

58.10 Vérifier la signature des attributs d'un paquet

La vérification a lieu :

- Lors de la mise à jour du fichier d'"index des paquets disponibles sur le client WAPT à partir de l'index Packages du dépôt.
- Lorsqu'une signature de paquet est vérifiée (installation, téléchargement) lorsqu'elle n'est pas en mode *développement*, c'est-à-dire si l'installation se fait à partir d'un fichier ZIP et non d'un répertoire de développement.

La vérification consiste à :

- Lire les attributs du fichier control depuis le fichier <WAPT>\control du ZIP du paquet.
- Récupérer le certificat X509 du signataire depuis le fichier <WAPT>\certificate.crt du ZIP du paquet.
- Décoder l'attribut signature du control depuis le format base64.
- Construire une structure JSON avec les attributs devant être signés (tels que définis dans la classe **PackageEntry**).
- Vérifier si la clé publique du certificat du titulaire peut vérifier le hachage de la liste structurée des attributs JSON et la signature du fichier control, en utilisant le hachage *sha256* et le remplissage *PKCS#1 v1.5*.
- Vérifier si le certificat est de confiance (soit présent en tant que tel dans les certificats de confiance, soit signé par une *Autorité de Certification* de confiance).

Dans le cas où nous devons vérifier les attributs sans avoir le paquet WAPT à disposition, nous récupérons la liste des certificats des détenteurs potentiels de certificats à partir du fichier d'index Packages sur le dépôt WAPT. Les certificats sont nommés ssl/<hexadecimal formated certificate fingerprint>.crt.

Un attribut de la structure control du paquet indique l'empreinte du certificat du signataire du fichier control.

58.11 Vérifier la signature d'un paquet

La vérification a lieu :

- Lors de l'installation d'un paquet sur un Poste client.
- Lors de l'édition d'un paquet existant.
- Lors de l'import d'un paquet depuis un dépôt externe (si option cochée dans la console).

La vérification consiste à :

- Récupérer le certificat X509 du signataire depuis le fichier <WAPT>\certificate.crt du ZIP du paquet.
- Vérifier que le certificat a été signé par une autorité de confiance dont le certificat est présent dans le fichier ssl du client WAPT.
- Vérifier la signature du fichier <WAPT>\manifest.sha256 avec la clé publique.

58.12 Signature d'une action immédiate

Depuis la console, les *Administrateurs* peut déclencher des actions directes sur le client WAPT, s'il est connecté au serveur par le mode Websockets.

La console WAPT signe ces actions avec la clé et le certificat de l'*Administrateur* avant de les envoyer au serveur WAPT en utilisant une requête HTTPS POST ; la requête est ensuite transmise aux clients WAPT ciblés.

Les actions possibles sont :

- `trigger_host_update`.
- `trigger_host_upgrade`.
- `trigger_install_packages`.
- `trigger_remove_packages`.
- `trigger_forget_packages`.
- `trigger_cancel_all_tasks`.
- `trigger_host_register`.
- `start_waptexit`.
- `show_message`.
- `trigger_host_update_server_status`.
- `trigger_change_description`.
- `trigger_waptservicerestart`.
- `unregister_computer`.
- `trigger_gpupdate`.
- `trigger_waptwua_scan`.
- `trigger_waptwua_download`.
- `trigger_waptwua_install`.
- `trigger_waptwua_uninstall`.
- `trigger_host_audit`.
- `trigger_audit_packages`.
- `trigger_cleanmgr`.
- `trigger_host_reboot`.
- `trigger_host_shutdown`.
- `trigger_session_setup`.
- `run_wol`.
- `get_tasks_status`.

58.12.1 Processus de signature pour des actions immédiates

- L'action est définie par son nom et des attributs dépendants de l'action. Les attributs sont *uuid*, *action*, *force*, *notify_server*, et *packages* (pour les actions impliquant une liste de paquets).
- Les attributs `signed_attributes`, `signer`, `signature_date`, `signer_certificate` sont ajoutés à la structure de l'action:
  - `signed_attributes` : liste des noms des attributs qui sont signés.
  - `signer` : Subject.COMMON_NAME du titulaire du certificat.
  - `signature_date` : date et heure en cours (UTC) sous la forme "%Y-%m-%dT%H:%M:%S".
  - `signer_certificate` : certificat *X509* du titulaire encodé en base64.
- La structure est encodée en JSON.
- La signature du JSON est calculée à partir de la clé privée RSA du `signer` en utilisant un algorithme de hachage *sha256* et un remplissage *PKCS1 v1.5*.
- La signature est encodée en base64 et stockée dans le JSON dans l'attribut `signature`.

58.12.2 Vérifier la signature d'une action immédiate

Depuis la console, les *Administrateurs* peut déclencher des actions directes sur le client WAPT, s'il est connecté au serveur par le mode Websockets.

Les actions sont encodées en JSON, signées avec la clé et le certificat de l'"*Administrateur* et relayées vers le client WAPT visé par le serveur WAPT.

L'action **`get_tasks_status`** ne demande pas d'authentification SSL.

Sur réception d'un évènement par la connexion Websocket du client WAPT :

- Le certificat X509 du signataire de l'action est extrait du json (format PEM).
- Le client WAPT teste si le certificat est un certificat de confiance, c'est-à-dire présent dans `<WAPT>\ssl` ou signé par une autorité de confiance (certificat de l'autorité présent dans `<WAPT>\ssl`).
- Le client WAPT teste si le certificat peut vérifier la signature présente dans la structure JSON de l'action ce qui consiste en:
  - Extraire la signature encodée en base64 dans le json depuis l'attribut `signature` dans le fichier JSON.
  - Extraire la date de signature formatée sous la forme "%Y-%m-%dT%H:%M:%S" depuis l'attribut `signature_date`.
  - Vérifier que la date de signature n'est pas trop ancienne ou dans le futur de plus de 10 minutes.
  - Reconstruire une représentation JSON des attributs de l'action.
  - Vérifier que la clé publique du certificat peut vérifier le JSON avec la signature en utilisant un algorithme de hachage *sha256* et un remplissage *PKCS1 v1.5*.

58.13 Vérification du téléchargement complet d'un paquet

Pour chaque paquet, une somme *md5* du paquet est calculée et disponible dans l'index `Packages` du dépôt.

Lors de l'installation d'un paquet, le client vérifie si le paquet est déjà disponible localement dans le répertoire `<WAPT>\cache`.

Si le fichier est présent, sa somme *md5* est comparée avec la somme *md5* présente dans l'index. Si elles diffèrent, le paquet en cache local est effacé.

Important: Cette somme md5 ne sert qu'à s'assurer qu'un paquet a été téléchargé complètement.

La vérification de la signature du paquet sera utilisé à la place de la somme md5et d'être effectivement assuré de l'intégrité et de l'authenticité du paquet.

CHAPTER 59

Appliquer les meilleures pratiques au packaging de logiciels

Note: _benwa est un administrateur système et il a autorisé Tranquil IT à republier son excellente diatribe sur reddit Developers, you can make sysadmins happier.

59.1 Variables d'environnement

- Les variables d'environnement existent depuis le DOS. Elles peuvent vous faciliter la vie (et la mienne).

59.2 Répertoires des programmes

- Tous les systèmes n'utilisent pas `C:\` comme lecteur principal. Certaines entreprises utilisent la redirection de dossiers, et déplacent le dossier Documents. Certains endroits dans le monde ne parlent pas anglais et leurs noms de répertoires reflètent cela. **Utilisez ces variables d'environnement pour que vos programmes fonctionnent tout simplement**:
  - `%SystemDrive%` est le lecteur où se trouve `%SystemRoot%`. Vous n'avez probablement pas besoin de le savoir ;
  - Le système d'exploitation Windows est situé dans le répertoire `%SystemRoot%`. Ne vous en souciez pas. Laissez le répertoire Windows tranquille ;
  - `%ProgramFiles%` est l'endroit où vous devez placer vos fichiers de programme, de préférence dans une structure `Company\Program` ;
  - `%ProgramFiles(x86)%` est l'endroit où vous devez placer vos fichiers de programme 32 bits. Veuillez les mettre à jour pour le 64 bits. Le 32-bit ne sera plus supporté dans l'avenir, et les entreprises attendront que vous vous organisiez pour bien plus longtemps que nécessaire ;
  - `ProgramData%` est l'endroit où vous devez stocker les données qui ne sont pas spécifiques à l'utilisateur, mais qui doivent quand même être écrites par les utilisateurs (les utilisateurs n'ont pas non plus d'accès en écriture à ce dossier).

Votre programme ne devrait pas nécessiter de droits d'administrateur pour s'exécuter, car vous ne devriez pas nous faire écrire dans le répertoire `%ProgramFiles%`. Aussi, ne mettez pas d'exécutables dans ce répertoire.

- `%Temp%` est l'endroit où vous pouvez traiter des données temporaires. Placez ces données dans un nom de dossier unique (peut-être un GUID généré) afin de ne pas provoquer d'incompatibilité avec un autre programme. Windows fera même le nettoyage à votre place. Ne placez pas de données temporaires dans les dossiers `%ProgramData%` ou `%ProgramFiles%` ;
- `%AppData%` vous permet de sauvegarder les paramètres de l'utilisateur qui exécute votre programme. C'est un endroit fantastique qui peut être synchronisé avec un serveur et être utilisé pour migrer rapidement et facilement un utilisateur vers une nouvelle machine et conserver tous les paramètres de ses programmes. Ne mettez pas de fichiers géants ou éphémères ici.

  Vous pourriez être à l'origine d'une connexion très lente si vous mettez les mauvais éléments ici et qu'une machine doit les synchroniser. **NE METTEZ PAS VOS FICHIERS DE PROGRAMMES ICI**. C'est l'entreprise qui décide quels logiciels sont autorisés à fonctionner, ce n'est pas à vous de décider, ni aux utilisateurs qui ne savent peut-être pas comment l'environnement de leur entreprise est configuré ;
- `LocalAppData%` permet de placer des fichiers plus volumineux spécifiques à un utilisateur ou à un ordinateur. Par exemple, personne n'a besoin de synchroniser un cache de vignettes. Elles ne seront pas transférées lorsqu'un utilisateur migrera vers une nouvelle machine, ou se connectera à une nouvelle station VDI, ou à un nouveau serveur de terminal. **NE METTEZ PAS VOS FICHIERS DE PROGRAMME ICI NON PLUS** ;

Note: De plus en plus d'éditeurs de logiciels proposent des versions *portables* de leurs logiciels qui s'installent et s'exécutent à partir de `%AppData%` ou de `%LocalAppData%`. Votre objectif est de permettre aux utilisateurs d'installer des logiciels même s'ils ne sont pas Administrateurs Locaux et vous commercialisez cela comme une fonctionnalité, bien qu'il s'agisse plutôt d'un NOGO de sécurité. Pire encore, vous avez tendance à rendre difficile de trouver le bon *MSI* qui permettrait à vos clients d'installer correctement votre logiciel dans `%ProgramFiles%`. Faites en sorte qu'il soit facile de trouver votre *MSI* qui s'installera dans les `%ProgramFiles%`, de cette façon vous ferez en sorte que les politiques de restriction des logiciels et de verrouillage des applications de vos clients fonctionnent bien et que leurs administrateurs système soient satisfaits.

Vous pouvez aussi bien obtenir ces chemins de répertoires par des appels API si vous n'utilisez pas ou ne pouvez pas utiliser de variables d'environnement.

59.3 Logs

- Utilisez le Windows Event Log pour la journalisation. Il gérera la rotation pour vous et un sysadmin peut transférer ces journaux ou faire ce qu'il faut. Vous pouvez même créer votre propre petite zone juste pour votre programme.

59.4 Codes d'erreur

- Utilisez les codes d'erreur standard lorsque vous quittez votre programme.

59.5 Impression

- Utilisez l'"API d'impression Windows et n'utilisez pas l'impression directe dans votre programme.

59.6 Distribution

- Distribuez votre programme en MSI. C'est le standard pour les fichiers d'installation de Windows (même si Microsoft ne l'utilise pas toujours lui-même).
- Signez vos fichiers d'installation et vos exécutables. C'est ainsi que nous savons que votre MSI est valide et que nous pouvons le mettre sur une liste blanche dans AppLocker ou équivalent.

Note: Applocker et Software Restriction Policies peuvent être très efficaces et la **gestion de ces stratégies peut être rendue plus simple avec WAPT**.

59.7 Mises à jour

- Vous souhaitez que votre programme se mette à jour ? C'est possible si l'entreprise est d'accord. Vous pouvez créer une tâche ou un service programmé qui s'exécute en mode élevé pour permettre cela sans accorder de droits d'administrateur à l'utilisateur. J'aime la façon dont Chrome Enterprise le fait : il donne une GPO pour définir les paramètres de mise à jour, la version maximale à laquelle elle va se mettre à jour (disons 81.* pour permettre toutes les mises à jour mineures automatiquement et les versions majeures sont manuelles), et un service. Ils ont également une GPO pour empêcher les installations lancées par les utilisateurs ;

Note: WAPT est conçu pour les entreprises qui ne permettent pas à leurs utilisateurs d'exécuter des mises à jour logicielles, c'est la politique souvent choisie par les grandes entreprises consciencieuses vis à vis de la sécurité.

59.8 Numéros de version

- Utilisez le versionnage semantique (doit aller dans la propriété de version dans le fichier d'installation et dans la liste Ajout/Suppression de programmes, pas dans le titre de l'application) et ayez un changelog. Vous pouvez également mettre à disposition en téléchargement votre installateur à un endroit prévisible pour permettre l'automatisation. Un chemin de mise à jour publié est également utile ;

Note: Si vous appliquez cette pratique, alors vous rendrez les administrateurs système qui déploient vos mises à jour logicielles en utilisant la fonction *WAPT def_update()* **très heureux** !

59.9 GPO

- Les modèles ADMX sont des trucs très moches ;

Note: Nous sommes tout à fait d'accord avec vous _benwa sur ce point chez Tranquil IT. Si les développeurs conseillent à leurs clients d'utiliser des GPO pour déployer leur logiciel ou leur système ou les paramètres des utilisateurs, alors, **ils doivent apprendre que les GPO ne sont pas fiables**.

Au lieu de cela, packagez vos logiciels, votre système et vos configurations utilisateur en utilisant WAPT. Un fichier `setup.py` est beaucoup plus facile qu'un fichier *`xml`* pour les administrateurs système qui doivent le vérifier avant de le déployer.

Les paquets WAPT peuvent être appliqués récursivement à des arbres d'Unités Organisationnelles, de sorte que votre paquet WAPT se comportera en production exactement comme le ferait une GPO, **juste beaucoup plus facilement**.

59.10 Dongles de licences

- Les dongles de licence USB sont un péché. Utilisez une licence de logiciel ordinaire ou une licence activée par réseau. Je suis sûr qu'il y plein de systèmes de gestion des licences sur le marché pour que vous n'ayez pas à réinventer la roue ;

Note: Vous pouvez faire en sorte que votre logiciel accepte une clé de licence comme paramètre dans votre exécutable *`msi`*.

WAPT peut être utilisé pour attribuer des clés de licence à des postes de travail individuels lors de l'installation en utilisant une méthode *qui garantit que la clé de licence ne peut pas être lue pendant le transport*.

Ensuite, si vous voulez que votre logiciel appelle chez vous pour vérifier la validité de la licence, faites en sorte que votre méthode fonctionne avec des *serveur mandataires*.

59.11 Fonctionnement en réseau

- N'utilisez pas ce fichu champ de saisie IPv4 personnalisé. Utilisez des FDQN. L'IPv6 existe depuis 1998 et fonctionnera avec votre logiciel si vous lui donnez une chance ;
- Le pare-feu Windows (je ne peux pas vraiment en dire plus sur les pare-feu tiers) va rester actif. Sachez faire la différence entre une règle entrante et sortante. Le plus souvent, votre serveur aura besoin de règles entrantes. La plupart du temps, vos clients n'auront même pas besoin de règles sortantes. Configurez-les au moment de l'installation, et non du lancement. Utilisez des groupes de pare-feu pour faciliter le filtrage. N'utilisez pas de règles quelconques si vous pouvez l'éviter. L'objectif n'est pas de faire fonctionner le système, mais de le faire fonctionner en toute sécurité. Si vous n'utilisez pas de numéros de version dans votre chemin d'installation, vous n'aurez peut-être même pas à refaire ces règles après chaque mise à niveau ;
- Les serveurs mandataires sont bons pour l'hygiène et les serveurs mandataires sont maintenant une caractéristique de sécurité par défaut non seulement dans les environnements informatiques des entreprises, mais aussi sur les petits réseaux. En rendant votre logiciel non compatible avec les proxies, les administrateurs réseau de vos clients devront établir et maintenir des règles spéciales pour leurs pares-feu, et cela rien que pour vos beaux yeux. Il est facile de coder votre logiciel pour qu'il fonctionne avec des serveurs mandataires, alors faites-le !

59.12 PDFs

- Ne livrez pas un logiciel qui nécessite d'autoriser le fonctionnement de javascript dans les lecteurs de PDF. La logique métier doit être exécutée avant la sortie au format PDF, pas après.

Note: Le *PDF* est le format de fichier que les gens utilisent par défaut pour échanger des documents. Les lecteurs PDF sont destinés à afficher des documents, et non à exécuter des programmes non signés.

CHAPTER 60

Stratégie de sortie des mises à jour de WAPT

Les mises à jour WAPT ne sortent pas selon un calendrier fixe.

Au lieu de cela, Tranquil IT sortira une nouvelle version majeure de WAPT lorsque de nouvelles mises à jour fonctionnelles majeures seront intégrées au cœur du produit.

Tranquil IT publiera des versions mineures intermédiaires de WAPT entre les versions majeures afin de corriger les défauts de fonctionnement et de sécurité.

60.1 Délai de publication entre les versions Enterprise et Discovery

Une nouvelle version majeure sera disponible en une RC1 (Release Candidate #1) comme **Enterprise** et cette même version sortira en **Discovery**. Avant de la sortir, la version Enterprise aura fait l'objet de tests internes approfondis avec des clients ayant l'"**Insider Program** pour s'assurer qu'aucune régression ne s'immisce dans le cœur de WAPT.

La version Enterprise passera par plusieurs RCs et la version finale de disponibilité générale Entreprise sera alors disponible entre 4 et 8 semaines après la sortie de la première version de nos clients avec un **Program Insider**.

Ce délai apportera plusieurs bénéfice au processus de sortie de la version Enterprise :

- Accorder plus de temps pour effectuer des tests approfondis des nouvelles fonctionnalités Enterprise tout en évitant les régressions.
- Il permet à Tranquil IT de travailler avec un petit groupe de clients d'entreprise sélectionnés pour s'assurer que les procédures de mise à niveau fonctionnent sans problème. En guise de récompense, ce groupe de clients sélectionnés bénéficie d'un accès direct aux développeurs et à l'équipe d'assistance de Tranquil IT, ce qui leur permet de s'entraîner et d'apprendre les nouvelles fonctionnalités WAPT avant qu'elles ne soient disponibles pour tout le monde.
- Donner un peu de temps au forum et à la liste de diffusion pour indexer les questions et les réponses qui seront éventuellement incluses dans la documentation officielle.
- Permettre à l'équipe de documentation de s'appuyer sur une base fonctionnelle figée pour ainsi documenter de manière fiable les fonctionnalités nouvelles ou améliorées.

- Donner à l'équipe de traduction le délai nécessaire pour mettre à jour les traductions.
- Permettre à l'équipe de communication et marketing de s'appuyer sur une base fonctionnelle figée pour ainsi rétro-planifier les annonces, les podcasts vidéo et la promotion générale de WAPT.

CHAPTER 61

Bulletin de sécurité

61.1 WAPT-2021-01 : CVE-2021-38608

- Brief : Une autorisation non sécurisée permet à un utilisateur s'exécutant en tant qu'invité d'élever ses privilèges.
- Annoncé : 13 août 2021.
- Impact : **Haut**.
- Produits : WAPT Enterprise & Community.
- Versions impactées : WAPT Enterprise < 2.0.0.9450, WAPT Enterprise < 1.8.2.7373 et WAPT Community < 1.8.2.7373.
- Description : Une autorisation non sécurisée permet aux utilisateurs du système d'exploitation invité d'élever leurs privilèges via l'agent WAPT.
- Rapporteur : Anass ANNOUR de l'équipe d'évaluation des risques ORM/ITT&AC, BNPParibas.
- Publié CVE : https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2021-38608.

CHAPTER 62

Changelog

62.1 WAPT-2.3 Serie

62.1.1 WAPT-2.3.0.13516 (2023-02-23)

hash : 69968974

This is a bugfix release for WAPT 2.3.

Attention: When upgrading from WAPT 2.2.3 to WAPT 2.3, when installing the new **waptsetup.exe** 2.3, if the **waptagent.exe** 2.2.3 had previously been installed ON the management machine ABOVE the **waptsetup.exe** 2.2.3, then the org certificate located in `wapt\ssl` directory of the agent belonged to the **waptagent.exe** 2.2.3 InnoSetup installation instead of being a local file, and was removed during upgrade to **waptsetup.exe** 2.3, which handles certificate deployment differently.

Now, in the case a **waptagent.exe** has been installed above a waptsetup.exe install, the certificates in `wapt\ssl` will be preserved during upgrade. This should happen only on the managemnent machine that is used to rebuild the agent if the agent has been re-installed above the **waptsetup.exe** install.

Note: The RHEL9 repository are how signed with a sha256 key/digest

WAPT Agent

- [IMP] waptsetup.exe : backup <wapt>\ssl*.crt before upgrading and restore after install
- [UPD] when building waptagent, check that there is at least one trusted cert for packages and actions
- [UPD] be more relax on waptagent setup naming: if setup exename « starts » with waptagent, assume we can safely use the configuration inside when running silently
- [IMP] waptsetup: don't ask innososetup to close applications using Microsoft Windows RestartManager. Use specific process name instead.

WAPT Console

- [FIX] fix zip64 for big packages (>2GB) not handled properly in waptconsole
- [FIX] waptconsole build waptagent certificate issue when both CA and personal cert+CA files are selected

WAPT Server

- [FIX] Debian : fix logrotate on wapt server

62.1.2 WAPT-2.3.0.13505 (2023-02-13)

hash : c7fcb3a7

This is a bugfix release for WAPT 2.3, and has been signed with a new code signing certificate to replace the expired one.

Attention: All the previous version of the 2.3 branch have an issue with the creation of the waptagent.exe due to a expiring code signing certificate. If you need to create a new WAPT Agent, please upgrade to this version.

The error message that you will get is « Error while creating waptagent.exe: Checking hashes of executables on server against Tranquil IT certificate has failed. Please check if waptbinaries.sha256 has not been altered. »

Message in French : « Erreur lors de la création du waptagent.exe : La vérification de la signature Tranquil IT des hashs de contrôle sur le serveur a échoué. Vérifier que le waptbinaries.sha256 n'a pas été altéré sur le serveur. »

WAPT core

- [FIX] better handling of filename with ".." and "~" in zip filenames. No need to be paranoid if ".." and "~" are in the middle of the name
- [FIX] waptservice only_if_no_process_running not taken in account when auto upgrade with waptupdate_task_period is enabled.
- [UPD] waptservice / core: include packages with install status == error when checking for conflicting packages to remove.
- [FIX] remote user waptmessage encoding issue
- [FIX] waptconsole waptpackage manifest add support for file with non ascii chars.
- [IMP] read Packages index from disk: use mormot function to potentially avoid lock conflicts
- [FIX] remove or forget packages with spaces in package name. fix RemoveDuplicates when there are spaces in data items.

- [FIX] closing WAPT Self for Linux/MacOSX
- [FIX] waptdeploy : update certificate pinning with new code signing certificate
- [FIX] waptcrypto : takes into account signature_date when checking certificate expiration date vs timestamping time.
- [SEC] update openssl binaries to 1.1.1t

WAPT Server

- [FIX] waptdeploy on server location: <repodir>/waptagent/waptdeploy.exe
- [SEC] add server_tokens off to avoid giving nginx server version to non authenticated clients
- [SEC] delete waptversion in /ping to avoid giving waptserver version to non authenticated clients
- [IMP] add view acl for get_storage_used_by_kbs

WAPT WADS

- [FIX] check volume letters before diskpart closes
- [IMP] waiting network for wgetwads Closes
- [IMP] install waptagent at end pressed debian
- [IMP] not force login in ipxescript if login already in ipxescript (for leave the possibility of forcing the language before)
- [IMP] add keymap on menu register
- [IMP] add login in pxe for linux deploy
- [IMP] delete double login wads

62.1.3 WAPT-2.3.0.13470 (2023-01-26)

hash : 4cc5fc06

This is a bugfix release for WAPT 2.3, and add support for Red Hat Enteprise Linux 9 and derivatives (both as server and agent)

WAPT Core

- [FIX] fix waptdeploy.exe unable to read setup exe version, requiring the use of force flag in GPO

WAPT Agent

- [FIX] fix datetime display for software inventory on Redhat and derivatives
- [IMP] better support for Red Hat os version numbering in inventory and tags
- [NEW] add el9 waptagent and waptserver support

WAPT Server

- [IMP] simplify web interface displayed version values to avoid misunderstanding
- [UPD] waptserver autocreate console ldap authenticated users if default_ldap_users_acls config is not empty
- [FIX] fix update_hosts_sid_table connexion leaks (to update the reachable column before calling query in reporting tab)
- [NEW] add el9 waptagent and waptserver support

WAPT Console

- [FIX] fix package maturity action default value if none chosen
- [FIX] fix grayed out host packages actions in Discovery mode
- [UPD] Strip comments in pem encoded certificates to reduce size and try to fit into the 32kb limit of stuffed exe.
- [IMP] adding possibility to cancel configuration package creation

WAPT WADS

- [IMP] add support for keyboard selection in ipxe
- [FIX] fix template windows 11 wads
- [UPD] wads: wait 30s for an ip address if dhcp is slow to respond or waiting for 802.1x vlan switch
- [FIX] fix wads regression where a computer could connect to waptserver instead of local secondary repo
- [IMP] Upload iso. Deleting file if wrong hash after upload
- [FIX] fix copy cert in winpe for wads
- [FIX] fix waptdeploy unable to read setup exe version, requiring the use of force flag

62.1.4 WAPT-2.3.0.13438 (2023-01-19)

hash : 8e580896

This is a bugfix release for WAPT 2.3. Those are mainly fixes and improvements to smooth the upgrade process from older WAPT versions.

WAPT Core

- [FIX] waptcore: keep install status of previous package if new package upgrade status is ERROR
- [FIX] Don't forced install packages which could't not be installed properly last time (to avoid install loop) a better approach could be to define a maximum retries count and an increasing delay between retries.

WAPT Console

- [FIX] fix verify **waptsetup.exe** and **waptdeploy.exe** hash when creating waptupgrade
- [UPD] set all search timer to default (300ms)
- [FIX] waptconsole display correct icon on Linux
- [UPD] waptconsole: propose to add a licence right after login if none on server.
- [FIX] waptconsole: fix some tab orders in forms
- [FIX] waptconsole package wizard: change layout for compatibility with linux.
- [FIX] waptconsole: quick fix for external repos settings if none is currently defined in waptconsole ini settings. Autoregister `wapt-templates`.
- [FIX] waptsetup: don't create a shortcut for the waptconsole to replicate behavior from older waptsetup…
- [NEW] waptagent for Windows can be generated on Linux waptconsole
- [REF] Improved djoin support
- [NEW] waptconsole: better support for dark mode on Linux / MacOS

WAPT Agent

- [IMP] macOS: use **sw_vers -productVersion** for mac os version
- [FIX] windows: waptwua client: fix issue when main repo url ends with a slash
- [FIX] fix **wapt-signpackage** compatibility error : removes mds argument
- [FIX] fix waptupgrade package for centos
- [FIX] fix application version on MacOsx
- [FIX] switch `DisableSkipWindowsUpdates` to `waptwua` section
- [NEW] Add ForceUnsigned for add drivers in winpe
- [FIX] add defaultInterpreterPath for vscode support
- [FIX] waptexit self-kill if machine has been started for too much time

WAPT WADS

- [IMP] wads: removing mounted drive letters before diskpart for better support of machine without any installed OS
- [NEW] Add script compile_ipxe.py to integrate waptserver url directly in ipxe binary
- [FIX] fix acl wads_admin on upload_winpe
- [FIX] wads: fix wads skip_login_wads and acl

WAPT Server

- [FIX] waptserver: don't try to convert jsonb boolean to raw boolean as it fails for postgresql <= 10
- [FIX] better support for postgres upgrade for Debian / Ubuntu in **postconf.py**
- [FIX] waptserver: path to **waptdeploy** on windows server to fix link
- [FIX] during upgrade, run **/opt/wapt/wapt-scanpackages.sh** when run **postconf.py**
- [NEW] waptserver: new option to set nginx port from waptserver.ini

62.1.5 WAPT-2.3.0.13356 (2023-01-10)

hash : fd590589

This is the first release of WAPT 2.3. This release does not have any new big feature, but brings a ton of little bugfixes and improvements to make WAPT usage more lean and smooth.

What's New?

- 1000+ bugfixes
- Less issues with false positive with antivirus software when deploying a new agent: WAPT Agents do not need to be rebuilt. The WAPT Agent is based on **waptsetup.exe** with certificate and configuration stored in the certificate signature of the file. The signature of the file is not altered.
- WAPT Agent on Linux and macOS: improved workflow for installing and updating the WAPT Agent.
- Improved Websocket connexion. Disconnects and reconnects have be made more robust.
- Improved support on macOS.
- Improved support on Linux.
- Update of WAPT external components.
- Tech Preview : WAPT Console support on Linux (Debian and derivatives, Redhat and derivatives)
- Tech Preview : WAPT Console support on macOS (Mojave and above).

Upgrade details

WAPT Server 2.3 needs PostgreSQL 10 or above. Please be sur to have the correct version running, especially if your server is running Debian and has been upgraded from Stretch:

- If the WAPT Server is running on Debian or Ubuntu, if you have upgrade from Debian Stretch to Buster to Bullseye, please check that the running instance of PostgreSQL has been upgraded when the OS has been upgraded;
- If you are on Redhat 7, upgrade is taken care of in the postconf script, and it should upgrade from 9.6 to 14;
- If the WAPT Server is running on Redhat 8 or derivative, then the DB is already in a good version;
- If the WAPT Server is running on Windows the DB upgrade is done during the upgrade from 9.6 to 14.

WAPT Core

- [SEC] When checking exe certificate, first check that the signature is OK.
- [SEC] when stuffing waptsetup.exe, check that waptsetup.exe downloaded from wapt server is properly signed by Tranquil IT.
- [FIX] Fixed handling properly utf8 chars in certificate subject.
- [FIX] Small improvement for wapt-get build-waptagent. Do not ask the server password twice.
- [FIX] Fixed stuffed legacy waptagent build: be sure to have a deterministic binary result when stuffing in waptconsole or server side.
- [IMP] remove client library dependency for command line progress bar.
- [SEC] waptpython 3.8.16 is now compiled with the isolated mode flag at true by default (Python -I)
- [REF] Removed unused functions.
- [REF] Removed unused headers.
- [IMP] waptservice: fix setting loglevel for specific components do not log WS listening too often. Fixed some action's « created_by » attributes which were not not set.
- [FIX] Windows setuphelpers: missing service_list in _all__.
- [FIX] **`wapt-get`**: moved *LoadOpenSSLFromPythonLib* to get proper path for *RegWaptBaseDir* on Linux.
- [NEW] Added armhf as a valid package architecture.
- [FIX] Fixed `scan_package` issue when there are packages without `package_uuid`. Packages table was growing at each `scan_packages`.
- [IMP] **`wapt-get`**: Added some help for `build-waptagent` and `add-config` / `reset-config`/ `set-config` *-from-url*.
- [IMP] `wapt-get reset-config-from-url`: removes dynamic configs from `conf.d` too.
- [IMP] Re-include empty folders in zipped WAPT packages.
- [FIX] Update for zip empty folder entries.
- [FIX] When checking files and directories from package manifest, create empty directories from the `manifest` file if thet do not exist yet.
- [UPD] `wapt-get update-package-sources`: Implicit transparent import of all functions from `packagesdevhelpers.py`.
- [FIX] Do not audit packages with `install_status` <> "OK".
- [SEC] waptpackage: Cleanup removed multiple MD type. We use only sha256 now.
- [NEW] waptconsole: Stuff `waptsetup` with *`json`* config for embedding into *waptupgrade* package.
- [FIX] waptpackage signature issue if the WAPT package is created from scratch with null attributes (ex. max_os_version). If signed, these null attributes are written to control file as sempty string, this breaks the signature control. So we initialize all default signed attributes to empty string instead of null.
- [UPD] `wapt-get create-waptagent`: Use *`json`* embedded config stuffed into certificate zone of executable signature.
- [FIX] Fixed regression in python _sign_control (encoding issue).
- [UPG] Upgraded python to 3.8.16.
- [IMP] `waptutils.py` cleanup and small fix in `user_is_member_of`.
- [REF] waptserver: Cleanup code with **`pyflakes`**.

- [IMP] Allow *none* loglevel.
- [NEW] Introduced `wapt-get reset-config-from-url`.
- [FIX] Fixed json_load_file() by adding encoding option. Default is « utf-8 ».
- [IMP] waptguihelper: Introduced StayOnTop argument for input_dialog() and grid_dialog()
- [FIX] Fixed `wapt-get add-config-from-url` in pure Pascal. The hash is retrieved from the filename if present, or as second parameter of command line.
- [REF] wapt python core: Removed sha1 compatibility with wapt 1.3 packages signatures.
- [FIX] Shows the proper logged user after login.
- [IMP] Fallback other method for get domain in get_hostname.
- [REF] `jsonconfig` data embedded in setup exe.
- [FIX] Default value for check verify cert.
- [UPD] Introduced uwaptjsonconfig (port of json config from python to FPC (FreePascal Compiler)).
- [UPD] **`wapt-get`**: Added a command to list the initial configs available on server (in `wapt/conf.d`).
- [UPD] waptsetuputil: Added UnzipConfigFromExe.
- [FIX] Removed global variable for PopupEnterprise, check Licensing after closing the window.
- [IMP] buildlib: Do not remove unittest from python lib when creating the build environment.
- [FIX] `remove_file()` was unable to remove symlinks.
- [FIX] wapt core: Regression on uuid retrieval from WMI. "System_Information" key is an array.
- [NEW] wapt core: added « wapt_temp_dir » `wapt-get.ini` parameter to specify the directory wher packages are unzipped at installation (for wyse terminal).
- [REF] Introduced packagesdevhelpers python module to remove helpers useful only for « packages source update » and reduce import time of setuphelpers.
- [IMP] windows_version() now getting the correct UBR (Update Build Revision) shown with « winver » command, adding windows_version_full in hardware inventory
- [IMP] waptguihelper: help improved for grid_dialog - also, introduced an (optional) Text parameter.
- [FIX] waptpackage: trim attributes value in `control` data. ("all" was retrieved as "all ").
- [IMP] twaptpackage: Always set architecture and priority default.
- [UPD] Refactored SSLCABundle usage.
- [FIX] Fixed waptpackage build issue when sourceroot includes the ending path separator. Fixed self service package building. Fixed version incbuild result.
- [FIX] Fixed issue with in path in zipped files created with `CreateRecursiveZip`.
- [FIX] Fixed file not found when calling `GetServerCertificate`.
- [FIX] Fixed editing zipped package inplace (hosts packages).
- [FIX] Added call to mormot2 `RegisterOpenssl` for Access violation in **`waptlicences`**.
- [IMP] Grid editor: Show which column is currently focused even if grid has not the focus.
- [IMP] Use UTC (Coordinated Universal Time) time for expiration check of ACLs.

- [UPD] wapt core: use datetime in UTC for `audit_data`.
- [IMP] wapt core: allow usage of an environment variable *waptbasedir* to specify the location of root *waptbasedir*.
- [IMP] Default grid order set to descending signature date.
- [FIX] Allow ~ character in WAPT package names (for spaces in Organizational Units packages).
- [FIX] waptcrypto: Fixed certificate filename attribute not set when loading a certificate chain.
- [UPD] Refactored `SSLCABundle` usage.
- [FIX] Fixed using particular characters in passwords.
- [FIX] Fixed waptcore: Fixed the type for dynamic configuration.
- [FIX] copytree2 `replace_at_next_reboot`.
- [REF] Moved all the dynamic json config functions into the WAPT class to take in account the actual agent settings (specially directories).
- [UPD] Created a full version 1.2.3.rev-hash into file `wapt/version-full`.

WAPT Agent

- [FIX] force create random uuid if bios uuid is not correct.
- [FIX] Do not check `wsusscn2.cab` if not Enterprise.
- [IMP] add host_capabilities inventory.
- [IMP] Better JSON format (Human Readable) for Audit Data.
- [FIX] Use parameter `IncludeCA` on `ListSOCertificatesFromFolder`.
- [FIX] Fixed translation issues in graphical components.
- [FIX] Fixed last version, checks the minimal OS version
- [FIX] edit waptwua if `install_delay` has value.
- [IMP] When uninstalling the WAPT Agent, stop the **`waptservice`** only if the service exists.
- [FIX] Popping wrong license message on new installation.
- [FIX] waptservice socketio: Force get new ws params in case of connection error like when config is updated.
- [FIX] Fixed add new rule missing import for `isenterprise`.
- [NEW] Added disk drives to host overview template.
- [IMP] Reduced size of host *`json`* inventory data. Do not send host configurations data if not changed. Do not send audit_data headers if no data. Fixed last audit data that was always sent.
- [IMP] Improved local waptservice auth feedback.
- [REF] Refactored waptservice code.
- [FIX] Enable custom CA file for websockets certificate checking.
- [FIX] Fixed WAPT Agent `websockets_verify_cert`: error reading setting from *`ini`* file. Reset socketioclient to None when connection error to force recreating the object with new TLS (Transport Layer Security) settings.
- [IMP] waptdeploy: Use only registry wapt_is1 install location to get the WAPT base directory.
- [IMP] waptdeploy: Do not check **`wapt-get`** working condition.

- [FIX] Fixed waptdeloy argument parsing.
- [UPD] waptsetup: Removed distribution of **innosetup** as it is no longer needed.
- [NEW] waptdeploy: Check that the WAPT Agent installer and **wapt-get.exe** are digitally signed by Tranquil IT.
- [FIX] waptdeploy wapt basedir guessing. Hardened waptdeploy RunTask.
- [FIX] Fixed `wapt-get build-waptagent`: empty configuration name.
- [ADD] Check all rules signatures before doing anything else.
- [IMP] The agent version is obtained from the *exe*, not the server.
- [FIX] waptsetup auto json config: should accept `waptsetup-1.2.3_<confname>_<confhash>.exe`.
- [FIX] Fixed remote WakeOnLAN.
- [IMP] waptservice: Do not include *PrinterPaperNames*, *PaperSizesSupported* and self_service_rules in inventory sent to the WAPT Server.
- [FIX] waptexit: If unable to get licences from waptservice, assume *is_enterprise* is False.
- [FIX] wapt-get: Set password callbacks after reloading config.
- [FIX] Shortened the upgrade scheduled task argument, as it is limited to 256 chars.
- [FIX] Stuffed waptsetup: Append waptwua settings to *json*.
- [FIX] waptserver socketio: Host does not register / reconnect by itself when deleted from the WAPT Server.
- [NEW] waptsetup.exe : If waptagent.exe is named, and only one config is embedded, take the first available config for the name of the configuartion to install instead of hardcoded « default ».
- [IMP] waptservice: Can start right after install even if no `wapt-get.ini`.
- [NEW] Added *nopassword* to config wizard for `service_auth_type`.
- [UPD] Added `wapt-get reset-config-from-url` and `set-config-from-url` json configuration.
- [FIX] Do not delete the files if the signature has failed.
- [IMP] waptsetup: Display a summary of embedded stuffed json configurations. Removed *use dynamic configuration* task.
- [FIX] waptserver: Fixed WakeOnLAN issue when no broadcast address exists in inventory.
- [FIX] `remove_user_appx` was not initialized from setuphelpers.
- [UPD] waptsetup: ApplyJsonConfigToIniFile when a *json* configuration is stuffed instead of `conf.d` dynamic configuration.
- [IMP] waptsetup: Do not update `wapt-get.ini` when using dynamic *json* configuration.
- [UPD] waptservice socketio: Do not require connection params update / reconnection try if there is no authorization token. When `allow_unauthenticated_connect` = **True** on the WAPT Server, the WAPT agents should be able to connect without getting a token.
- [FIX] waptself: Fixed next page button unavailable on last page.
- [UPD] waptexit: Add `waptexit_disable_skip_windows_updates` parameter in `wapt-get.ini` file and commandline argument to disable the checkbox for skipping Windows Updates.
- [UPD] wapt-get: Return ExitCode <> 0 when an exception is raised Added **ping --service** command to check waptservice accessibility from waptsetup.
- [UPD] waptself: Display details of WAPT package on top of packages list to avoid reframes.
- [UPD] Enable `waptservice_allow_all_packages` only for *nopassword* `service_auth_type`.

- [NEW] Added a waptservice parameter `waptservice_allow_all_packages` which allow all user to install / remove all packages as if they were part of the waptselfservice group.
- [NEW] If a *json* configuration is provided in waptsetup as stuffed data in certicode certificate area, use it for initial configuration.
- [FIX] Improved error message and wait cursor when waptselfservice is starting.
- [FIX] Fixed selfservice missing common module for `self_service_rules` when using the *nopassword* argument with the WAPT Enterprise version.
- [FIX] Changed Icon for *Add Dependencies* → *Trashcan* to *Plus*.
- [IMP] User is now informed when self service can not get a token (service not started).
- [FIX] Remove double slahs in url *//Packages*.
- [NEW] Added Ubuntu22 in waptsetup package.
- [FIX] Fixed waptmessage ambiguous "-s" option (use stdout and set window size), replaced by -c for init console.
- [FIX] Fixed tasks list on host.
- [FIX] Normalized view (lowercase) in grid for *target_os* from control part.
- [FIX] Fixed execution of waptmessage in file instead of base64 (to avoid too long command line).
- [FIX] Use cached trusted signer certificates store instead of recreating it each time.
- [FIX] Fixed signed_attributes written as string list (instead of python form) and signer is the signer certificate *Common Name*.
- [IMP] use **`--not-interactive`** with register if the installation runs in silent mode.
- [FIX] waptservice: Do not ignore broadcast for `WaptUpdateServerStatus` to avoid the WAPT Tray sticking upon sending data to the WAPT Server.
- [FIX] Fixed unable to synchronize remote repository.
- [IMP] waptmessage: No autosize if a size is specified on the command line.
- [FIX] Fixed no hash in clipboard, added missing helper for `add-config-from-url` in wapt-get.
- [IMP] Limit access right to Administrators to log directory (in case non public stuff gets written to logs).
- [IMP] `install_scheduling` work if not in PENDING_UPDATES status.
- [FIX] Fixed waptexit compilation: Removed specific `WaptIniFilename` function.
- [FIX] Fixed waptmessage unable to load sqlite.
- [IMP] Updated waptwua status to "NEED-SCAN" on hosts when `download_wsusscan` is triggered and `wsusscn2.cab` file is downloaded.
- [NEW] wapt core: Added `as_dict` and descending parameters to `Wapt.read_audit_data_set`.
- [IMP] Do not take care anymore of maturity for version when it is compared to WAPT store version.
- [FIX] Fixed configuration package template `setup_package_template_conf.py`.
- [FIX] Fixed waptservice configuration: Set the `configs_dir` relative to `wapt-get.ini` full path.
- [FIX] Fixed waptservice "start_waptexit" with arguments Faulty arguments boolean value decoding.
- [FIX] Fixed bad arguments sent to waptservice triggering upgrades with `only_priorities` and `only_if_not_process_running`.
- [FIX] Fixed `Wapt.write_audit_data_if_changed`: Write data if previous data has expired.

- [FIX] Updated the template of dynamic *json* configuration packages to match new location and naming of *json* configuration related functions.
- [NEW] Option `include_potentially_superseded_updates` in configuration wizard.
- [FIX] Fixed waptservice: Be sure to dynamically revert to default setting when a key is removed from `wapt-get.ini`.
- [FIX] Fixed waptservice: Make sure we have a random `secret_key` for local waptservice session.
- [NEW] WAPTWUA superseded support.
- [IMP] **`wapt-get edit`** now opens `update_package.py` too.
- [UPD] Added a *NEED-SCAN* waptwua.status, updated when `Wapt.update()` is called.
- [FIX] Fixed waptself: Set focus on search when opening.
- [IMP] Ignore history for waptwua status.
- [FIX] Fixed **`wapt-get update-package-sources`**: Handle properly relative path to package sources.
- [FIX] Fixed **`wapt-get update-package-sources`**: use `devdirupdate_package.py` to call `update_package()` hook if this file exists. Else use `setup.py`.
- [IMP] wapttray: Launch external **`waptself`** and **`waptconsole`** with OpenDocument instead of windows only ShellExecuteW.
- [FIX] Workaround fix when **`pyscripter`** is put as editor for packages. `params_vscod_list` fixed when space in parameters. Reupdated description.
- [IMP] **`wapt-get edit`** now opens `changelog.txt`, VSCod* now opens `control` file too. **`wapt-get edit`** can now be run as user with VSCod* updating `wapt_sources_edit()` description.
- [UPD] Changed default log path to `wapt/log` if writable.
- [UPD] Same logging initialization code for all UI executables with `waptcommon.InitLoggingFromCommandLine`.
- [IMP] waptservice waptself: localauth with file token (ie. nopassword). Handles local groups.

WAPT Console

- [FIX] display an explicit error message if a new host package can not be saved on the WAPT Server because of acl.
- [IMP] Process application messages when performing file hash/zip actions.
- [FIX] Fixed waptconsole copy cert to `wapt/ssl`: handle properly spaces in target directory name.
- [FIX] Place cursor at end of line instead of point of click in textareas.
- [ADD] Popup Menu with Copy and Copy as JSon for Audit TreeView.
- [FIX] Fixed proper images on actions buttons.
- [FIX] Fixed OU icon when OU name contains an empty character.
- [FIX] Fixed Out of bound error : add verification on condition check in specific cases.
- [FIX] Fixed missing error message on secondary repositories.
- [IMP] Improve usability of copying new certificate in `<WAPT>\ssl` directory
- [FIX] Fixed icon on action `ActWUAGetUnusedKB`.
- [FIX] Fixed actions caption on toolbar in Windows Update.
- [FIX] Fixed removing ability to personalize toolbuttons on ISO, configs, and drivers in *OS Deployment*.

- [FIX] Fixed popup menus on toolbar in *OS Deployment*.
- [FIX] Fixed actions on toolbar in *Software Inventory*.
- [NEW] waptconsole / waptserver: Added a specific ACL for `update_audit_data`.
- [UPD] Increasing softwares max count limit in *Software Inventory* from 5000 to 20000.
- [FIX] Fixed locking some actions on non Enterprise versions.
- [FIX] Fixed waptconsole package zip build: `CreateRecursiveZip`.
- [IMP] cleanup of HTML templates on waptservice. Removed unused js.
- [IMP] Showing icons for *target_os*.
- [FIX] Fixed waptconsole TX509Store: when intermediate certificates are provided in user *.pem* certificate file, only trust the first one.
- [FIX] Fixed waptconsole waptcrypto: implement `TX509Store.GetCertificatesChainFromFingerprint`. Fixed self signed certificates are always trusted when checking the WAPT package.
- [FIX] Fixed waptconsole: when signing packages, make sure we end with LF only (n unix style) `control` files.
- [IMP] Basic POC (Proof of Concept) for Auto Completion on Reporting Queries.
- [FIX] Fixed viewing TechPreview Features does not take care of display preferences.
- [FIX] Fixed the downloaded packages have now the chosen maturity.
- [IMP] Show `*.cmd` files in post install script selector.
- [NEW] Upload a default *json* configuration on the WAPT Server when building **waptagent.exe**. Fixed **waptsetup.exe** stuffing on the WAPT Server when uploading a *json* configuration.
- [FIX] Fixed the button Type for update package warning.
- [ADD] Confirm button before Update from the WAPT store.
- [FIX] Fixed waptconsole update from the WAPT store Introduced `StripPrefix` in `TPackageRequest` to allow searching in the repository on package name without prefix.
- [FIX] Include `min_os_version` and `max_os_version` in WAPT package identification to check which WAPT package is newest.
- [FIX] When building customized waptsetup, sometimes missing trusted certificate.
- [FIX] Fixed the copy of `wapt-get.ini` if there is no `waptconsole.ini`.
- [NEW] Menu item for restoring toolbars to default.
- [FIX] Fixed actions on toolbar in *WAPT Development* and *OS Deployment* forms.
- [FIX] Fixed removing certificates in create waptsetup [NEW] function for listing certificates from folder.
- [FIX] Fixed buttons links with actions on WSUS.
- [FIX] Fixed encoding problem for WSUS.
- [IMP] Removed GUI interface for the Update from the store action.
- [ADD] Added a warning message before updating a WAPT package.
- [ADD] Updated from the store button in private repository done.
- [IMP] Added Updated part for the Store Update Action.

- [IMP] Update from the store button (visual part).
- [FIX] Fixed regression on creating new *wuagroup* package.
- [UPD] waptconsole *build agent -> named with version*, config and hash instead of **waptagent.exe/**.
- [FIX] Fixed __pycache__ included in zipped package when built from waptconsole.
- [ADD] reporting: Added Unique save for each query.
- [FIX] Fixed SQL query editor: any query can be edited at any time, without erasing the others.
- [FIX] Fixed SQL query editor: if queries are already created and registered and have the same name, you can edit both without overwriting the other one.
- [IMP] Use system font for html viewers.
- [IMP] Allow package wizard without installer path.
- [NEW] Added « keys » mustache helper for html templates.
- [IMP] waptconsole: Do not try to ping servers before login dialog.
- [FIX] Fixed enabling build and upload if all information are set / pre configuration in case of portable app if an executable is found.
- [UPD] waptconsole Cyberwatch integration. Added Values mustache helper to format dict as list for Cyberwatch html report template. Added styled Cyberwatch example audit template.
- [IMP] Addied listening to ipv6 only if ipv6 is available.
- [FIX] Fixed waptconsole crash if custom column with empty size cell.
- [IMP] Added a warning when no DNS record is found (Remote repository).
- [FIX] Fixed call if app is currently closing (login cancelled).
- [IMP] Opening configuration by double-clicking on grid.
- [IMP] Package wizard for portable apps.
- [IMP] waptconsole, display bytes size in human readable format in grid.
- [FIX] Fixed OU options that are now available if the user is currently focusing the *OU* grid.
- [IMP] Improved asking credentials on http error 401.
- [FIX] Fixed waptconsole: random timeout error when running commands from the WAPT Console.
- [FIX] Fixed WAPT package creation for OUs (Organizational Units).
- [ADD] Link to the official documentation for the Config Package Wizard.
- [IMP] Proper restore of GUI when WindowState is maximized. Prevent flickering if starting maximized.
- [IMP] Improved warning before deleting a valid licence.
- [FIX] Fixed waptconsole regression: import packages. Check the signature even if it is disabled in remote repository settings.
- [FIX] Fixed waptconsole regression on additional private repositories listed in the repositories tab, even if not defined in `repositories` setting in `waptconsole.ini`.
- [FIX] Fixed waptconsole: private key password is not asked again if a matching key can not be found or decrypted.
- [REF] waptserver model upgrade: removed unused database migration steps.
- [UPD] waptserversetup: avoid automatic restart when installing MSVC (Microsoft Visual C++) 2022.

- [FIX] Fixed error editing same OU package in one session.
- [ADD] ACL Edit Repo on Index for secondary repos
- [FIX] Fixed missing editing ACL *Edit Repo*.
- [FIX] Fixed waptconsole access violation when checking unzipped package signature.
- [FIX] Fixed waptonsole multiple update of hosts corrupt packages depends grid display.
- [IMP] **waptself**, **wapt-get**, **waptexit**, **wapttray**: kill check threads on close, even on linux to speed up application shutdown.
- [UPD] waptconsole: lazy loading of DMPython. Removed python source scripter tab on main form. Moved to (inactive) uvispysources. Removed debug panel on main form removed unused `uvissearchpackage`. Added some euristic icons on audit and reporting grids depending on well known values (OK, ERROR etc…).
- [IMP] Improved the interpretation of checkbox states due to label description.
- [IMP] Improved search when importing queries.
- [FIX] Fixed host configuration package that are not editable right after creating them.
- [FIX] Fixed waptconsole pkcs12 export and email in X509 certificates.
- [IMP] Removed Python dependency in the WAPT Console.
- [UPD] waptconsole: Added popup menu to Json hardware treeview.
- [IMP] Improved reporting import, now select all queries by default + some code improvement
- [IMP] Improved enabling or disabling ACL by double click.
- [FIX] Fixed waptconsole: html audit templates. Bad search order.
- [FIX] Fixed waptconsole: actions categories fixes and updates. Hide unused categories from toolbars customization.
- [FIX] Fixed waptconsole: empty success message for some actions. Updated translations.
- [FIX] Fixed waptconsole get agents installers: fixed MISSING -> OK status.
- [UPD] Fixed waptconsole: Added Edit html template popup menu action.
- [FIX] Fixed no logo resizing if smaller size.
- [UPD] Load html templates for `host_overview` and `host_audit` from user's `appdata` directory if it exists, else from `wapt`.
- [REF] waptconsole: Refactored `TFrmHtmlViewer` to lookup templates either in user templates directory (`%APPDATA%waptconsoletemplates`) or in default `wapt` installation directory (`%WAPTBASEDIR%templates`).
- [UPD] waptconsole: Improved drag & drop of columns into GridHosts.
- [FIX] Fixed blocking action editing WSUS package if no Enterprise licence is active.
- [FIX] Fixed waptconsole drag & drop audit values.
- [FIX] Fixed waptconsole regression when signing *unit* package or modyfing stripped down WAPT packages.
- [IMP] waptconsole: Load AD Groups in thread.
- [FIX] Fixed waptconsole compilation without *USE_WAPTPACKAGE* flag.
- [REF] waptconsole: Introduced an interface for uwaptpackage TWaptPackage WIP: fix compilation when *USE_WAPTPACKAGE* is defined TODO: implement IX509Store
- [FIX] Fixed waptconsole: fixed host overview layout if no html template.

- [UPD] waptconsole: host details layout changes: introduced html templates based overview if `templateshost_overview.html` file exists (mustache template).
- [FIX] Fixed waptconsole *sendmessage* gui splitter.
- [IMP] waptconsole: check that downloaded waptsetup version is the same or newer than that of the WAPT Server.
- [FIX] Fixed autosearch in `ttissearch` component.
- [NEW] waptconsole: Added a popumenu *copy to clipboard as json for audit data*.
- [IMP] waptconsole: allow drag & drop of a audit *`json`* value subkey from value tree explorer.
- [NEW] waptconsole: displays audit history and WIP audit data explorer (treeview + html template).
- [FIX] Fixed reporting queries grid layout not saved properly.
- [UPD] GUI Vis ACL: zebra colored lines and added possibility to change user password from one button (same action like in right click on user).
- [FIX] Fixed avoiding exception if no user was selected before adding ACL rights.
- [FIX] Fixed trigger downloads when triggering updates from the WAPT Console (missing import).
- [UPD] Updated icons on windows update status for WUA.
- [FIX] Fixed waptconsole check external repository version timeout exception raised in frontend.
- [FIX] Fixed waptconsole multiserver: fixed can't trigger action on servers other than main WAPT Server.
- [FIX] Fixed waptconsole: Avoid error message of no `repo_url` for last used package template section.
- [FIX] Fixed modifying a password if old password was empty.
- [ADD] Hide / show all columns in grids.
- [NEW] new option `check_package_version` in `waptconsole.ini`.
- [UPD] waptconsole reporting: Added a quick search filtering zone for the query result.
- [FIX] Fixed wrong message when no admin rights and the WAPT Agent needs to be upgraded or is not present.
- [UPD] Host menu for upgrading hosts part.
- [REF] waptconsole multiserver: Refactored `TriggerActionOnHosts` to send multiples actions to the right servers.
- [FIX] Fixed waptconsole: use ROOT in addition to CA windows system certificates stores when building **`winpe`** with `verify_cert` = **True**.
- [UPD] Deleted host popup.
- [NEW] Possibility to download WAPT packages when asking hosts for updates.
- [UPD] `trigger_host_update` adding possibility to download the WAPT package after update.
- [FIX] Fixed waptconsole: The WAPT Console crashed when checking newest packages if wapt-templates repository is protected with an encrypted client key.
- [FIX] Fixed saving configuration when new configuration was created.
- [FIX] Fixed saving language parameter.
- [FIX] Fixed waptconsole: access violation when access to external repository is blocked or needs a proxy.
- [FIX] Fixed waptconsole multiserver regression: unable to edit a WAPT package which was just edited.
- [FIX] Fixed waptconsole edit conf package: Do not close if error when uploading to the WAPT Server.

- [FIX] patched `setup_package_template_cert.py.tmpl`.
- [FIX] Fixed not adding « cn » in OU.
- [FIX] Fixed layout on Windows Update part.
- [FIX] Fixed the flow layout.
- [IMP] waptconsole: WIP multiserver. Mostly works for hosts, but not for packages management.
- [FIX] Fixed waptconsole: re-enable dataexport to *.csv* for grids.
- [NEW] Explicit hint on number version when the WAPT package is not up to date (GridPackages).
- [REF] Refactored private key password handling. Added a callback to clear cached key password in case of decrypt error in http client. Stores client https authentication key password in same storage as package private keys.
- [REF] WIP for multiserver console. WaptCookieManager takes in accounts the domain. TODO: send allowed session cookies for cross domain auth. Lazy loading of waptserver instance. Loads list of servers in `DMWaptConsole.ReloadConfigFile`. All sections with a `wapt_server` key are taken in account. Shares the WaptServerSession across all waptserver connections.
- [FIX] Fixed bad port for **veyon**.

WAPT Server

- [SEC] Windows: waptserversetup.exe windows: do not reenable acl inheritance on wapt root folder.
- [SEC] Send minimal information on /ping api call.
- [IMP] Set session cookie to 3 days
- [IMP] waptserversetup: Check if there is a CRITICAL log entry during `winsetup.py` and exit with an exitcode 1000 if it is the case.
- [IMP] waptserver: Do not automatically create users in wapt database when user logs in with kerberos (self-service case).
- [FIX] Fixed waptserverinstall windows: regression unable to install on new windows machine if wapt was not already installed.
- [REF] Server python code cleanup.
- [IMP] wapttasks: use environment variable on linux to pass `config` file path.
- [NEW] waptserver: reduced lifetime of session cookie to default 12h. `session_lifetime` can be changed in `waptserver.ini` using `session_lifetime` seconds parameter.
- [UPD] Updated to python 3.8.16 for all supported operating systems.
- [FIX] Fixed stuffed setup exe naming on the WAPT Server.
- [NEW] new parameter `list_subnet_skip_login_wads`.
- [FIX] Fixed waptserver: shorten SQL columns aliases for long `get_hosts` *json* queries.
- [SEC] Upgraded **werkzeug** 2.0.2 -> 2.1.1 for PYSEC-2022-203.
- [NEW] waptservice websocket: Enabled certificate checking on websockets.
- [IMP] waptserver: Added index on `computer_ad_ou`.
- [FIX] Fixed waptserver: by default, do not create stuffed `waptsetup` when a dynamic config is uploaded.
- [FIX] Fixed waptserversetup: if installService, configure the local service to reach newly installed server. Propose to start the WAPT Console right after for demo mode.
- [NEW] `model.py`: Added `upgrade-db` action and **`--overwrite-version=1.2.3`** option to force the replay of upgrade db.

- [FIX] Fixed waptserver **nginx** config, there can be spaces in path. quotes include.
- [NEW] Be sure to not start the WAPT Server if the database structure can not be upgraded properly.
- [NEW] If licences *json* data is empty, assume an empty list.
- [IMP] Getting storage used by KBs.
- [NEW] 22H2 build numbers in WindowsVersions class.
- [NEW] Added `hosts_sid` endpoint routing to uwsgi in nginx configuration templates.
- [FIX] Fixed **wapt-get build-waptagent**: create **waptagent.exe** link on the WAPT Server.
- [FIX] Fixed waptserver: ignore null bytes in audit data string values.
- [FIX] Fixed waptserver: allow access to agent download without client certificate auth.
- [FIX] Fixed waptserver model: remove references to unused HostExtData table.
- [FIX] Fixed waptserver multiinstance with uwsgi: takes in account `application_root` for interprocess `get_ws_connections` /api/v3/hosts_sid calls.
- [UPD] Added waptserver /api/v3/update_hosts_sid_table endpoint to fill the HostWebsocket table with the in memory ws_connections for reporting purpose.
- [UPD] Changed the path of the untouched **waptsetup.exe** on the WAPT Server: moved to the `wapt/waptagent` folder to be consistent with other agents location Same for **waptdeploy.exe**.
- [DEL] waptserver: Removed « enable_store » setting.
- [UPD] waptconsole multiserver: display unreachable servers.
- [FIX] Fixed waptserversetup: Reinclude waptwua even if service is not installed to allow **wapt-get** usage.
- [FIX] Fixed waptconsole multiserver dynamic config: bad server url for checking https certificate.
- [FIX] Fixed waptconsole multiserver: Do not include a server at startup if it is not pingable.
- [UPD] waptserversetup windows: Removed some additional unused files when waptservice is not installed.
- [UPD] waptconsole multi servers: Do not try to update / merge repo if `repo_url` is empty.
- [IMP] waptserver / waptservice websockets: When registering host, return an authentication token in response, so that websockets can connect without additional roundtrip.
- [IMP] `allow_unauthenticated_registration` is now like `use_kerberos`.
- [FIX] Postconf, current config is now autoselected.
- [UPD] waptsetup waptserversetup: Sign the installers and uninstallers using embedded iscc logic.
- [UPD] waptserver db: Changed the primary key of tables *HostPackagesStatus*, *HostExtData*, *Packages*, *HostSoftwares*, *HostGroups*, *HostWebsocket*, *HostAuditData*, *ReportingSnapshots*, *HostWsus*, *LogsAPI* to bigint.
- [UPD] waptserversetup: Check that the user is a LOCAL computer user and not a domain user.
- [FIX] Fixed waptserversetup: postgresql upgrade. Try to fix ACLs on data directory.
- [FIX] Added a conflict on apache2 in the Linux WAPT Server package to avoid old install leftovers.
- [REF] Removed `enterprise_common.py`.
- [UPD] Upgrade **nginx** on Windows.
- [UPD] Upgraded DB to **postgresql v14** for windows.

- [UPG] upgraded **postgresql** 9.6 to **v14** on CentOS7.
- [FIX] Fixed waptserver: Fixed sid map sharing in uwsgi mode (missing imports).
- [FIX] Fixed waptserver websocket: Be sure to not clear a SID which would be newer than current disconnect event. Not sure if disconnect / reconnect are always synchronous.
- [FIX] Fixed waptserver: Improved message when triggering action.
- [IMP] Added HTST (HTTP Strict Transport Security) header to nginx template.
- [FIX] Fixed waptserver `update_hosts_audit_data`: Updated values with same global key (host_id,value_id).
- [FIX] Added **trigger_host_action** ACL on /api/v3/connected_wol_relays (used by /api/v3/trigger_wakeonlan).
- [IMP] waptserver websocket auth: Put host certificates in cache.
- [UPD] waptserver websocket: Do not cache UUID twice.
- [REF] waptserver websockets: use a global in memory dictionary to hold the host uuid -> SID of connected host to avoid Database insert or updates.
- [FIX] Fixed server regression for custom *json* fields ValueError: too many values to unpack (expected 3).
- [IMP] waptserver: WIP endpoint `update_hosts_audit_data` to bulk insert hosts related data.
- [IMP] waptserver: update api/v3/get_agents_info to match the online `wapt_agent_list.json`.
- [FIX] Fixed glpi sync: simplified `glpi_upload_hosts.py` script.
- [FIX] Fixed waptserver huey tasks: `licences_list` not properly initialized when not using default `waptserver.ini`.
- [FIX] Fixed waptserver audit table structure upgrade: typo
- [FIX] Fixed avoiding GET method limits on `hosts_for_wua`.
- [FIX] Fixed waptserver unable to delete some hosts when CRL is enabled be tolerant if the host certificate is not issued by this server's CA.
- [FIX] Fixed waptconsole multiserver: Computers identified by fqdn uuid are not displayed properly in the grid.
- [UPD] Remove references to **waptsetup-tis.exe** -> renamed to **waptsetup.exe**.
- [FIX] Fixed `update_server_status` with dynamic configuration.
- [IMP] Include **waptsetup.exe** in **waptserversetup.exe**.

WADS

- [FIX] Clear WADS stdout before and after **diskpart** to avoid broken stdout.
- [IMP] Check whether `winpe.wim` and `7z.exe` files exist when creating the WADS WinPE.
- [FIX] Added missing "/" in `wgetwads` error messages.
- [IMP] WADS: Added session login type and acl.
- [IMP] WADS: Login to server only one time instead of for each request.
- [IMP] WADS: Added flags: unchecked for wads login on Windows Server.
- [IMP] Use of latest mormot function for `WgetWads` to fix DNS check.
- [IMP] Improved error messages for WADS and WGETWADS.
- [IMP] Added option *wads* in Windows Server installer.

- [IMP] `get_wads_secondary_repo` –> follow protocol of the server connection.
- [FIX] Fixed `list_subnet_skip_login_wads` read configuration.
- [IMP] WinPE creation key
- [REF] Remove useless code on `get_wads_config` (Login WADS).
- [IMP] WgetWads does not require python to work.
- [FIX] Be more indulgent on *`json`* rules for WADS.
- [FIX] Fixed WADS working when no logging required.
- [ADD] Login in IPXE, more tests needed.
- [IMP] Proper way to secure `wads_get_config`.
- [ADD] Login on WADS register host and get wads configuration.
- [NEW] include hostname in `debian.ipxe` for OS deployment.
- [FIX] Fixed `djoin` with given domainuser parameter.
- [IMP] Added back support GET method on /api/v3/get_wads_config.
- [NEW] Added asset tag in HostOSDeploy.
- [IMP] Ask for a new hostname when starting to deploy if hostname equals to "autoregister".
- [IMP] Improved filtering keyboard faster + french translation in *Make WinPE*.
- [FIX] Fixed missing glob import in WADS `get_iso_config`.
- [NEW] Adding drivers in WinPE from WADS drivers.
- [IMP] Improved feedback when the `djoin` fails (already existing machine).
- [WADS] <Value> format in *`XML`* was incorrect and not complete for password definition.
- [IMP] Last error message added for failed `djoin`.
- [FIX] Fixed uninstall **`wapttftpserver`** when uninstalling **`waptserver`**.
- [IMP] Improved file upload with hash check wads *`iso`* files listed from the WAPT Server even if not saved in the WAPT Console.
- [NEW] Added customized WinPE export to zip file.
- [IMP] Improved showing the error message on upload failure.
- [IMP] Improved applying default configuration on wads host if no configuration has been set.
- [IMP] ISO download dialog box.
- [IMP] WADS: WinPE now pinging WAPT Server. Selected language keyboard layout will be available directly in a new cmd.
- [IMP] WADS: XML no longer disable UAC by default.
- [FIX] Fixed `mac_address` not returned with iPXE.
- [ADD] Added `ipxe_script_jinja_path` and two templates.
- [UPD] Added file type filters for loading the post-install script.
- [FIX] Restored a progression bar when uploading the *`ISO`* and the *`winpe`* files.
- [IMP] kill **`wapttftpserver`** and uninstall the service before installing it.
- [ADD] Added Windows 11 unattend *`XML`* template files.

- [IMP] Improved searching WADS data (hosts, isos, driver bundles, configurations).
- [FIX] Added tftp **firewalld** port opening.
- [IMP] Avoid creating WinPE on Windows partition + some ACL added.
- [UPD] Renamed drivers bundle filenames to sha256(filename).
- [ADD] Added a template for Debian.
- [UPD] *GridConfigDeploy* showing the platform now.
- [FIX] Fixed saving bundle names.
- [NEW] Injecting a:abbr:*OEM (Original Equipment Manufacturer)* key by **slmgr** command.
- [FIX] Fixed SELinux rules for wads.
- [FIX] Potential fix for (over 10 joins for djoin by a standard user on MSAD).
- [UPD] WADS grayed when windows update repository is selected.
- [UPD] Possibility to select an *iso* file even if not Windows.
- [FIX] Fixed waptconsole `uploadWinPE`: regression in upload progress bar and incomplete zip.
- [FIX] Fixed wads to include non CA certificates for WinPE build.
- [IMP] Added `ipxe_script` in DeployConfig table.

WAPT Agent MacOS

- [UPD] Delete old *pkg* if available in *pkg* list.
- [NEW] Added fake menu for macOS for letting user to quit the app from the *MainMenu*.
- [FIX] Improved support for macOS *MenuBar*.
- [FIX] Added WAPT Console *.app* plist file for macOS X.
- [FIX] Fixed some macOS X model and serial number reports.
- [FIX] Fixed macOS X `local_groups` key in `host_info`.
- [FIX] Updated mormot2 for **gssapi** on macOS X.
- [NEW] support WADS security, Network masks.
- [FIX] Fixed `installed_softwares` on MacOS.
- [NEW] Added timestamping to *pkg* signing.
- [FIX] Fixed getting agent version in `get_wads_config`.
- [NEW] Added entitlements file for macOS signing.
- [IMP] Force light UI when DarkMode is active on macOS.
- [FIX] Fixed opening maximized self service on macOS
- [FIX] Fixed loading hosts on macOS when *more options* in inventory is checked.
- [IMP] Better handle on input (utf8 convertion).
- [IMP] macOS: Updated build script to handle binary file signing and better debugging.
- [IMP] Patched `dmidecode` info for macOS.

- [FIX] Fixed macOS core `get_hostname` return binary string instead of str -> update_status loop.
- [IMP] Use `system_profiler_info` for `dmi_info` on macOS X.
- [REF] `plistlib.readPlistFromBytes` deprecation fix.
- [FIX] Fixed core macOS: use UUID from `system_profiler_info` instead of `dmidecode`.
- [FIX] Fixed duplicated macOS code in setuphelpers for `get_hostname()`.
- [IMP] Improved mounting content for *.pkg*, *.mpkg*, *.app* only if file is not symbolic.
- [NEW] Added the WAPT Console to Linux and macOS gui distribution.
- [IMP] Fixed keyword and name with `installed_softwares` in macOS and Linux.
- [FIX] Fixed register for macOS.
- [FIX] Fixed custom waptmessage logo linux.
- [FIX] Fixed **harakiri** on non Windows kills all running processes.
- [FIX] Fixed restart waptservice for macOS.
- [IMP] Silently attach *dmg* file.
- [FIX] Fixed `get_file_type` in macOS.

WAPT Agent Linux

- [FIX] Fixed logrotate on RedHat8 for waptserver and wapttasks.
- [IMP] **wapt-get.bin**: Improved python traceback format with proper line endings on non Windows.
- [IMP] Improve support for dark mode on WAPT Console on Linux
- [IMP] Replaced in `/usr/bin/` **wapt-get.sh** by **wapt-get.bin**.
- [IMP] Added Ubuntu and CentOS icons.
- [IMP] Added icons in *ImportPackages* window.
- [FIX] Fixed `user_local_appdata` for Linux.
- [IMP] waptagent Debian package: removed system python3 dependency.
- [IMP] Avoid loop in checkbox events on search inventory especially on operating systems other than Windows.
- [IMP] Added `PYTHONNOUSERSITE` = **True** to all *.sh* scripts to avoid spoiling PYTHONPATH with locally installed libraries in user home directory.
- [UPD] Disable compression on unix WAPT agent bundle (each package is itself already compressed).
- [NEW] Added the WAPT Console to Linux and MacOS gui distribution.
- [FIX] Fixed *configpackage* wizard and main form layouts for Linux.
- [UPD] Updated virtualtreeview for Linux visual grid lines improvements.
- [IMP] Fixed keyword and name with `installed_softwares` in macOS and Linux.
- [FIX] Fixed **harakiri** on non Windows kills all running processes.
- [ADD] Added snap software inventory.
- [FIX] Fixed waptservice linux restart Linux: AttributeError: `WaptServiceRestart` object has no attribute *logger*.

- [NEW] Linux OS deployment.
- [FIX] Added **`firewalld`** rule on RedHat based server for **`wapttftpserver`**.

62.1.6 WAPT-2.3.0.13334 RC3 (2023-01-06)

hash : a06031bd

This is the third release candidate of WAPT 2.3.

This is a release candidate for testing that is not intended for production.

This changelog lists the fixes sinces WAPT 2.3 RC2.

WAPT Core

- [SEC] When checking exe certificate, first check that the signature is OK.
- [SEC] when stuffing waptsetup.exe, check that waptsetup.exe downloaded from wapt server is properly signed by Tranquil IT.
- [FIX] Fixed handling properly utf8 chars in certificate subject.
- [FIX] Small improvement for wapt-get build-waptagent. Do not ask the server password twice.
- [FIX] Fixed stuffed legacy waptagent build: be sure to have a deterministic binary result when stuffing in waptconsole or server side.
- [IMP] remove client library dependency for command line progress bar.

WAPT Agent

- [FIX] force create random uuid if bios uuid is not correct.
- [FIX] Do not check `wsusscn2.cab` if not Enterprise.

WAPT Server

- [SEC] Windows: waptserversetup.exe windows: do not reenable acl inheritance on wapt root folder.
- [SEC] Send minimal information on /ping api call.
- [IMP] Set session cookie to 3 days

WAPT Console

- [FIX] display an explicit error message if a new host package can not be saved on the WAPT Server because of acl.
- [IMP] Process application messages when performing file hash/zip actions.
- [FIX] Fixed waptconsole copy cert to `wapt/ssl`: handle properly spaces in target directory name.
- [FIX] Place cursor at end of line instead of point of click in textareas.

WADS

- [FIX] Clear WADS stdout before and after **diskpart** to avoid broken stdout.
- [IMP] Check whether `winpe.wim` and `7z.exe` files exist when creating the WADS WinPE.
- [FIX] Added missing "/" in `wgetwads` error messages.

WAPT Linux

- [FIX] Fixed logrotate on RedHat8 for waptserver and wapttasks.
- [IMP] **wapt-get.bin**: Improved python traceback format with proper line endings on non Windows.
- [IMP] Improve support for dark mode on WAPT Console on Linux

62.1.7 WAPT-2.3.0.13301 RC2 (2023-01-04)

hash: a2af0e8d

What's New?

This is second release candidate of WAPT 2.3. This is second release candidate of WAPT 2.3.

This is a release candidate for testing that is not intended for production.

This changelog lists the fixes sinces WAPT 2.3 RC1.

Note : for security reasons in waptpython, Python isolated mode is now enabled by default (Python -I). If you are using the waptpython Python environment outside of WAPT, please be sure to check for the corresponding Python documentation.

WAPT Core

- [SEC] waptpython 3.8.16 is now compiled with the isolated mode flag at true by default (Python -I)

WAPT Console

- [ADD] Popup Menu with Copy and Copy as JSon for Audit TreeView.
- [FIX] Fixed proper images on actions buttons.
- [FIX] Fixed OU icon when OU name contains an empty character.
- [FIX] Fixed Out of bound error : add verification on condition check in specific cases.
- [FIX] Fixed missing error message on secondary repositories.
- [IMP] Improve usability of copying new certificate in `<WAPT>\ssl` directory

WAPT Agent

- [IMP] add host_capabilities inventory.
- [IMP] Better JSON format (Human Readable) for Audit Data.
- [FIX] Use parameter `IncludeCA` on `ListSOCertificatesFromFolder`.
- [FIX] Fixed translation issues in graphical components.
- [FIX] Fixed last version, checks the minimal OS version
- [FIX] edit waptwua if `install_delay` has value.

WADS

- [IMP] WADS: Added session login type and acl.
- [IMP] WADS: Login to server only one time instead of for each request.
- [IMP] WADS: Added flags: unchecked for wads login on Windows Server.
- [IMP] Use of latest mormot function for `WgetWads` to fix DNS check.
- [IMP] Improved error messages for WADS and WGETWADS.
- [IMP] Added option *wads* in Windows Server installer.
- [IMP] `get_wads_secondary_repo` –> follow protocol of the server connection.
- [FIX] Fixed `list_subnet_skip_login_wads` read configuration.
- [IMP] WinPE creation key

WAPT Linux Agent

- [IMP] Replaced in `/usr/bin/` **`wapt-get.sh`** by **`wapt-get.bin`**.
- [IMP] Added Ubuntu and CentOS icons.
- [IMP] Added icons in *ImportPackages* window.

62.1.8 WAPT-2.3.0.13239 RC1 (2022-12-21)

hash: 675d861e

What's New?

- 1000+ bugfixes
- Less issues with false positive with antivirus software when deploying a new agent: WAPT Agents do not need to be rebuilt. The WAPT Agent is based on **`waptsetup.exe`** with certificate and configuration stored in the certificate signature of the file. The signature of the file is not altered.
- WAPT Agent on Linux and macOS: improved workflow for installing and updating the WAPT Agent.
- Improved Websocket connexion. Disconnects and reconnects have be made more robust.

- Improved support on macOS.
- Improved support on Linux.
- Update of WAPT external components.

Tech Preview

- WAPT Console support on Linux (Debian and derivatives, Redhat and derivatives).
- WAPT Console support on macOS (Mojave and above).

WAPT Core

- [REF] Removed unused functions.
- [REF] Removed unused headers.
- [IMP] waptservice: fix setting loglevel for specific components do not log WS listening too often. Fixed some action's « created_by » attributes which were not not set.
- [FIX] Windows setuphelpers: missing service_list in _all__.
- [FIX] **`wapt-get`**: moved *LoadOpenSSLFromPythonLib* to get proper path for *RegWaptBaseDir* on Linux.
- [NEW] Added armhf as a valid package architecture.
- [FIX] Fixed `scan_package` issue when there are packages without `package_uuid`. Packages table was growing at each `scan_packages`.
- [IMP] **`wapt-get`**: Added some help for `build-waptagent` and `add-config` / `reset-config`/ `set-config` *-from-url*.
- [IMP] `wapt-get reset-config-from-url`: removes dynamic configs from `conf.d` too.
- [IMP] Re-include empty folders in zipped WAPT packages.
- [FIX] Update for zip empty folder entries.
- [FIX] When checking files and directories from package manifest, create empty directories from the `manifest` file if thet do not exist yet.
- [UPD] `wapt-get update-package-sources`: Implicit transparent import of all functions from `packagesdevhelpers.py`.
- [FIX] Do not audit packages with `install_status` <> "OK".
- [SEC] waptpackage: Cleanup removed multiple MD type. We use only sha256 now.
- [NEW] waptconsole: Stuff `waptsetup` with *`json`* config for embedding into *waptupgrade* package.
- [FIX] waptpackage signature issue if the WAPT package is created from scratch with null attributes (ex. max_os_version). If signed, these null attributes are written to control file as sempty string, this breaks the signature control. So we initialize all default signed attributes to empty string instead of null.
- [UPD] `wapt-get create-waptagent`: Use *`json`* embedded config stuffed into certificate zone of executable signature.
- [FIX] Fixed regression in python _sign_control (encoding issue).
- [UPG] Upgraded python to 3.8.16.
- [IMP] `waptutils.py` cleanup and small fix in `user_is_member_of`.
- [REF] waptserver: Cleanup code with **`pyflakes`**.

- [IMP] Allow *none* loglevel.
- [NEW] Introduced `wapt-get reset-config-from-url`.
- [FIX] Fixed json_load_file() by adding encoding option. Default is « utf-8 ».
- [IMP] waptguihelper: Introduced StayOnTop argument for input_dialog() and grid_dialog()
- [FIX] Fixed `wapt-get add-config-from-url` in pure Pascal. The hash is retrieved from the filename if present, or as second parameter of command line.
- [REF] wapt python core: Removed sha1 compatibility with wapt 1.3 packages signatures.
- [FIX] Shows the proper logged user after login.
- [IMP] Fallback other method for get domain in get_hostname.
- [REF] `jsonconfig` data embedded in setup exe.
- [FIX] Default value for check verify cert.
- [UPD] Introduced uwaptjsonconfig (port of json config from python to FPC).
- [UPD] **`wapt-get`**: Added a command to list the initial configs available on server (in `wapt/conf.d`).
- [UPD] waptsetuputil: Added UnzipConfigFromExe.
- [FIX] Removed global variable for PopupEnterprise, check Licensing after closing the window.
- [IMP] buildlib: Do not remove unittest from python lib when creating the build environment.
- [FIX] `remove_file()` was unable to remove symlinks.
- [FIX] wapt core: Regression on uuid retrieval from WMI. "System_Information" key is an array.
- [NEW] wapt core: added « wapt_temp_dir » `wapt-get.ini` parameter to specify the directory wher packages are unzipped at installation (for wyse terminal).
- [REF] Introduced packagesdevhelpers python module to remove helpers useful only for « packages source update » and reduce import time of setuphelpers.
- [IMP] windows_version() now getting the correct UBR (Update Build Revision) shown with « winver » command, adding windows_version_full in hardware inventory
- [IMP] waptguihelper: help improved for grid_dialog - also, introduced an (optional) Text parameter.
- [FIX] waptpackage: trim attributes value in `control` data. ("all" was retrieved as "all ").
- [IMP] twaptpackage: Always set architecture and priority default.
- [UPD] Refactored SSLCABundle usage.
- [FIX] Fixed waptpackage build issue when sourceroot includes the ending path separator. Fixed self service package building. Fixed version incbuild result.
- [FIX] Fixed issue with in path in zipped files created with `CreateRecursiveZip`.
- [FIX] Fixed file not found when calling `GetServerCertificate`.
- [FIX] Fixed editing zipped package inplace (hosts packages).
- [FIX] Added call to mormot2 `RegisterOpenssl` for Access violation in **`waptlicences`**.
- [IMP] Grid editor: Show which column is currently focused even if grid has not the focus.
- [IMP] Use UTC time for expiration check of ACLs.

- [UPD] wapt core: use datetime in UTC for `audit_data`.
- [IMP] wapt core: allow usage of an environment variable *waptbasedir* to specify the location of root *waptbasedir*.
- [IMP] Default grid order set to descending signature date.
- [FIX] Allow ~ character in WAPT package names (for spaces in Organizational Units packages).
- [FIX] waptcrypto: Fixed certificate filename attribute not set when loading a certificate chain.
- [UPD] Refactored `SSLCABundle` usage.
- [FIX] Fixed using particular characters in passwords.
- [FIX] Fixed waptcore: Fixed the type for dynamic configuration.
- [FIX] copytree2 `replace_at_next_reboot`.
- [REF] Moved all the dynamic json config functions into the WAPT class to take in account the actual agent settings (specially directories).
- [UPD] Created a full version 1.2.3.rev-hash into file `wapt/version-full`.

WAPT Agent

- [IMP] When uninstalling the WAPT Agent, stop the **`waptservice`** only if the service exists.
- [FIX] Popping wrong license message on new installation.
- [FIX] waptservice socketio: Force get new ws params in case of connection error like when config is updated.
- [FIX] Fixed add new rule missing import for `isenterprise`.
- [NEW] Added disk drives to host overview template.
- [IMP] Reduced size of host *`json`* inventory data. Do not send host configurations data if not changed. Do not send audit_data headers if no data. Fixed last audit data that was always sent.
- [IMP] Improved local waptservice auth feedback.
- [REF] Refactored waptservice code.
- [FIX] Enable custom CA file for websockets certificate checking.
- [FIX] Fixed WAPT Agent `websockets_verify_cert`: error reading setting from *`ini`* file. Reset socketioclient to None when connection error to force recreating the object with new TLS settings.
- [IMP] waptdeploy: Use only registry wapt_is1 install location to get the WAPT base directory.
- [IMP] waptdeploy: Do not check **`wapt-get`** working condition.
- [FIX] Fixed waptdeloy argument parsing.
- [UPD] waptsetup: Removed distribution of **`innosetup`** as it is no longer needed.
- [NEW] waptdeploy: Check that the WAPT Agent installer and **`wapt-get.exe`** are digitally signed by Tranquil IT.
- [FIX] waptdeploy wapt basedir guessing. Hardened waptdeploy RunTask.
- [FIX] Fixed `wapt-get build-waptagent`: empty configuration name.
- [ADD] Check all rules signatures before doing anything else.
- [IMP] The agent version is obtained from the *`exe`*, not the server.
- [FIX] waptsetup auto json config: should accept `waptsetup-1.2.3_<confname>_<confhash>.exe`.

- [FIX] Fixed remote WakeOnLAN.
- [IMP] waptservice: Do not include *PrinterPaperNames*, *PaperSizesSupported* and self_service_rules in inventory sent to the WAPT Server.
- [FIX] waptexit: If unable to get licences from waptservice, assume *is_enterprise* is False.
- [FIX] wapt-get: Set password callbacks after reloading config.
- [FIX] Shortened the upgrade scheduled task argument, as it is limited to 256 chars.
- [FIX] Stuffed waptsetup: Append waptwua settings to *json*.
- [FIX] waptserver socketio: Host does not register / reconnect by itself when deleted from the WAPT Server.
- [NEW] waptsetup.exe : If waptagent.exe is named, and only one config is embedded, take the first available config for the name of the configuartion to install instead of hardcoded « default ».
- [IMP] waptservice: Can start right after install even if no `wapt-get.ini`.
- [NEW] Added *nopassword* to config wizard for `service_auth_type`.
- [UPD] Added `wapt-get reset-config-from-url` and `set-config-from-url` json configuration.
- [FIX] Do not delete the files if the signature has failed.
- [IMP] waptsetup: Display a summary of embedded stuffed json configurations. Removed *use dynamic configuration* task.
- [FIX] waptserver: Fixed WakeOnLAN issue when no broadcast address exists in inventory.
- [FIX] `remove_user_appx` was not initialized from setuphelpers.
- [UPD] waptsetup: ApplyJsonConfigToIniFile when a *json* configuration is stuffed instead of `conf.d` dynamic configuration.
- [IMP] waptsetup: Do not update `wapt-get.ini` when using dynamic *json* configuration.
- [UPD] waptservice socketio: Do not require connection params update / reconnection try if there is no authorization token. When `allow_unauthenticated_connect` = **True** on the WAPT Server, the WAPT agents should be able to connect without getting a token.
- [FIX] waptself: Fixed next page button unavailable on last page.
- [UPD] waptexit: Add `waptexit_disable_skip_windows_updates` parameter in `wapt-get.ini` file and commandline argument to disable the checkbox for skipping Windows Updates.
- [UPD] wapt-get: Return ExitCode <> 0 when an exception is raised Added **`ping --service`** command to check waptservice accessibility from waptsetup.
- [UPD] waptself: Display details of WAPT package on top of packages list to avoid reframes.
- [UPD] Enable `waptservice_allow_all_packages` only for *nopassword* `service_auth_type`.
- [NEW] Added a waptservice parameter `waptservice_allow_all_packages` which allow all user to install / remove all packages as if they were part of the waptselfservice group.
- [NEW] If a *json* configuration is provided in waptsetup as stuffed data in certicode certificate area, use it for initial configuration.
- [FIX] Improved error message and wait cursor when waptselfservice is starting.
- [FIX] Fixed selfservice missing common module for `self_service_rules` when using the *nopassword* argument with the WAPT Enterprise version.
- [FIX] Changed Icon for *Add Dependencies* → *Trashcan* to *Plus*.
- [IMP] User is now informed when self service can not get a token (service not started).

- [FIX] Remove double slahs in url *//Packages*.
- [NEW] Added Ubuntu22 in waptsetup package.
- [FIX] Fixed waptmessage ambiguous "-s" option (use stdout and set window size), replaced by -c for init console.
- [FIX] Fixed tasks list on host.
- [FIX] Normalized view (lowercase) in grid for *target_os* from control part.
- [FIX] Fixed execution of waptmessage in file instead of base64 (to avoid too long command line).
- [FIX] Use cached trusted signer certificates store instead of recreating it each time.
- [FIX] Fixed signed_attributes written as string list (instead of python form) and signer is the signer certificate *Common Name*.
- [IMP] use `--not-interactive` with register if the installation runs in silent mode.
- [FIX] waptservice: Do not ignore broadcast for `WaptUpdateServerStatus` to avoid the WAPT Tray sticking upon sending data to the WAPT Server.
- [FIX] Fixed unable to synchronize remote repository.
- [IMP] waptmessage: No autosize if a size is specified on the command line.
- [FIX] Fixed no hash in clipboard, added missing helper for `add-config-from-url` in wapt-get.
- [IMP] Limit access right to Administrators to log directory (in case non public stuff gets written to logs).
- [IMP] `install_scheduling` work if not in PENDING_UPDATES status.
- [FIX] Fixed waptexit compilation: Removed specific `WaptIniFilename` function.
- [FIX] Fixed waptmessage unable to load sqlite.
- [IMP] Updated waptwua status to "NEED-SCAN" on hosts when `download_wsusscan` is triggered and `wsusscn2.cab` file is downloaded.
- [NEW] wapt core: Added `as_dict` and descending parameters to `Wapt.read_audit_data_set`.
- [IMP] Do not take care anymore of maturity for version when it is compared to WAPT store version.
- [FIX] Fixed configuration package template `setup_package_template_conf.py`.
- [FIX] Fixed waptservice configuration: Set the `configs_dir` relative to `wapt-get.ini` full path.
- [FIX] Fixed waptservice "start_waptexit" with arguments Faulty arguments boolean value decoding.
- [FIX] Fixed bad arguments sent to waptservice triggering upgrades with `only_priorities` and `only_if_not_process_running`.
- [FIX] Fixed `Wapt.write_audit_data_if_changed`: Write data if previous data has expired.
- [FIX] Updated the template of dynamic *`json`* configuration packages to match new location and naming of *`json`* configuration related functions.
- [NEW] Option `include_potentially_superseded_updates` in configuration wizard.
- [FIX] Fixed waptservice: Be sure to dynamically revert to default setting when a key is removed from `wapt-get.ini`.
- [FIX] Fixed waptservice: Make sure we have a random `secret_key` for local waptservice session.
- [NEW] WAPTWUA superseded support.
- [IMP] **`wapt-get edit`** now opens `update_package.py` too.
- [UPD] Added a *NEED-SCAN* waptwua.status, updated when `Wapt.update()` is called.

- [FIX] Fixed waptself: Set focus on search when opening.
- [IMP] Ignore history for waptwua status.
- [FIX] Fixed **wapt-get update-package-sources**: Handle properly relative path to package sources.
- [FIX] Fixed **wapt-get update-package-sources**: use `devdirupdate_package.py` to call `update_package()` hook if this file exists. Else use `setup.py`.
- [IMP] wapttray: Launch external **waptself** and **waptconsole** with OpenDocument instead of windows only ShellExecuteW.
- [FIX] Workaround fix when **pyscripter** is put as editor for packages. `params_vscod_list` fixed when space in parameters. Reupdated description.
- [IMP] **wapt-get edit** now opens `changelog.txt`, VSCod* now opens `control` file too. **wapt-get edit** can now be run as user with VSCod* updating `wapt_sources_edit()` description.
- [UPD] Changed default log path to `wapt/log` if writable.
- [UPD] Same logging initialization code for all UI executables with `waptcommon.InitLoggingFromCommandLine`.
- [IMP] waptservice waptself: localauth with file token (ie. nopassword). Handles local groups.

WAPT Console

- [FIX] Fixed icon on action `ActWUAGetUnusedKB`.
- [FIX] Fixed actions caption on toolbar in Windows Update.
- [FIX] Fixed removing ability to personalize toolbuttons on ISO, configs, and drivers in *OS Deployment*.
- [FIX] Fixed popup menus on toolbar in *OS Deployment*.
- [FIX] Fixed actions on toolbar in *Software Inventory*.
- [NEW] waptconsole / waptserver: Added a specific ACL for `update_audit_data`.
- [UPD] Increasing softwares max count limit in *Software Inventory* from 5000 to 20000.
- [FIX] Fixed locking some actions on non Enterprise versions.
- [FIX] Fixed waptconsole package zip build: `CreateRecursiveZip`.
- [IMP] cleanup of HTML templates on waptservice. Removed unused js.
- [IMP] Showing icons for *target_os*.
- [FIX] Fixed waptconsole TX509Store: when intermediate certificates are provided in user *.pem* certificate file, only trust the first one.
- [FIX] Fixed waptconsole waptcrypto: implement `TX509Store.GetCertificatesChainFromFingerprint`. Fixed self signed certificates are always trusted when checking the WAPT package.
- [FIX] Fixed waptconsole: when signing packages, make sure we end with LF only (n unix style) `control` files.
- [IMP] Basic POC for Auto Completion on Reporting Queries.
- [FIX] Fixed viewing TechPreview Features does not take care of display preferences.
- [FIX] Fixed the downloaded packages have now the chosen maturity.
- [IMP] Show `*.cmd` files in post install script selector.
- [NEW] Upload a default *json* configuration on the WAPT Server when building **waptagent.exe**. Fixed **waptsetup.exe** stuffing on the WAPT Server when uploading a *json* configuration.

- [FIX] Fixed the button Type for update package warning.
- [ADD] Confirm button before Update from the WAPT store.
- [FIX] Fixed waptconsole update from the WAPT store Introduced `StripPrefix` in `TPackageRequest` to allow searching in the repository on package name without prefix.
- [FIX] Include `min_os_version` and `max_os_version` in WAPT package identification to check which WAPT package is newest.
- [FIX] When building customized waptsetup, sometimes missing trusted certificate.
- [FIX] Fixed the copy of `wapt-get.ini` if there is no `waptconsole.ini`.
- [NEW] Menu item for restoring toolbars to default.
- [FIX] Fixed actions on toolbar in *WAPT Development* and *OS Deployment* forms.
- [FIX] Fixed removing certificates in create waptsetup [NEW] function for listing certificates from folder.
- [FIX] Fixed buttons links with actions on WSUS.
- [FIX] Fixed encoding problem for WSUS.
- [IMP] Removed GUI interface for the Update from the store action.
- [ADD] Added a warning message before updating a WAPT package.
- [ADD] Updated from the store button in private repository done.
- [IMP] Added Updated part for the Store Update Action.
- [IMP] Update from the store button (visual part).
- [FIX] Fixed regression on creating new *wuagroup* package.
- [UPD] waptconsole *build agent -> named with version*, config and hash instead of **`waptagent.exe/`**.
- [FIX] Fixed __pycache__ included in zipped package when built from waptconsole.
- [ADD] reporting: Added Unique save for each query.
- [FIX] Fixed SQL query editor: any query can be edited at any time, without erasing the others.
- [FIX] Fixed SQL query editor: if queries are already created and registered and have the same name, you can edit both without overwriting the other one.
- [IMP] Use system font for html viewers.
- [IMP] Allow package wizard without installer path.
- [NEW] Added « keys » mustache helper for html templates.
- [IMP] waptconsole: Do not try to ping servers before login dialog.
- [FIX] Fixed enabling build and upload if all information are set / pre configuration in case of portable app if an executable is found.
- [UPD] waptconsole Cyberwatch integration. Added Values mustache helper to format dict as list for Cyberwatch html report template. Added styled Cyberwatch example audit template.
- [IMP] Addied listening to ipv6 only if ipv6 is available.
- [FIX] Fixed waptconsole crash if custom column with empty size cell.
- [IMP] Added a warning when no DNS record is found (Remote repository).

- [FIX] Fixed call if app is currently closing (login cancelled).
- [IMP] Opening configuration by double-clicking on grid.
- [IMP] Package wizard for portable apps.
- [IMP] waptconsole, display bytes size in human readable format in grid.
- [FIX] Fixed OU options that are now available if the user is currently focusing the *OU* grid.
- [IMP] Improved asking credentials on http error 401.
- [FIX] Fixed waptconsole: random timeout error when running commands from the WAPT Console.
- [FIX] Fixed WAPT package creation for OUs.
- [ADD] Link to the official documentation for the Config Package Wizard.
- [IMP] Proper restore of GUI when WindowState is maximized. Prevent flickering if starting maximized.
- [IMP] Improved warning before deleting a valid licence.
- [FIX] Fixed waptconsole regression: import packages. Check the signature even if it is disabled in remote repository settings.
- [FIX] Fixed waptconsole regression on additional private repositories listed in the repositories tab, even if not defined in `repositories` setting in `waptconsole.ini`.
- [FIX] Fixed waptconsole: private key password is not asked again if a matching key can not be found or decrypted.
- [REF] waptserver model upgrade: removed unused database migration steps.
- [UPD] waptserversetup: avoid automatic restart when installing MSVC 2022.
- [FIX] Fixed error editing same OU package in one session.
- [ADD] ACL Edit Repo on Index for secondary repos
- [FIX] Fixed missing editing ACL *Edit Repo*.
- [FIX] Fixed waptconsole access violation when checking unzipped package signature.
- [FIX] Fixed waptonsole multiple update of hosts corrupt packages depends grid display.
- [IMP] **`waptself`**, **`wapt-get`**, **`waptexit`**, **`wapttray`**: kill check threads on close, even on linux to speed up application shutdown.
- [UPD] waptconsole: lazy loading of DMPython. Removed python source scripter tab on main form. Moved to (inactive) uvispysources. Removed debug panel on main form removed unused `uvissearchpackage`. Added some euristic icons on audit and reporting grids depending on well known values (OK, ERROR etc…).
- [IMP] Improved the interpretation of checkbox states due to label description.
- [IMP] Improved search when importing queries.
- [FIX] Fixed host configuration package that are not editable right after creating them.
- [FIX] Fixed waptconsole pkcs12 export and email in X509 certificates.
- [IMP] Removed Python dependency in the WAPT Console.
- [UPD] waptconsole: Added popup menu to Json hardware treeview.
- [IMP] Improved reporting import, now select all queries by default + some code improvement
- [IMP] Improved enabling or disabling ACL by double click.
- [FIX] Fixed waptconsole: html audit templates. Bad search order.

- [FIX] Fixed waptconsole: actions categories fixes and updates. Hide unused categories from toolbars customization.
- [FIX] Fixed waptconsole: empty success message for some actions. Updated translations.
- [FIX] Fixed waptconsole get agents installers: fixed MISSING -> OK status.
- [UPD] Fixed waptconsole: Added Edit html template popup menu action.
- [FIX] Fixed no logo resizing if smaller size.
- [UPD] Load html templates for `host_overview` and `host_audit` from user's `appdata` directory if it exists, else from `wapt`.
- [REF] waptconsole: Refactored `TFrmHtmlViewer` to lookup templates either in user templates directory (`%APPDATA%waptconsoletemplates`) or in default `wapt` installation directory (`%WAPTBASEDIR%templates`).
- [UPD] waptconsole: Improved drag & drop of columns into GridHosts.
- [FIX] Fixed blocking action editing WSUS package if no Enterprise licence is active.
- [FIX] Fixed waptconsole drag & drop audit values.
- [FIX] Fixed waptconsole regression when signing *unit* package or modyfing stripped down WAPT packages.
- [IMP] waptconsole: Load AD Groups in thread.
- [FIX] Fixed waptconsole compilation without *USE_WAPTPACKAGE* flag.
- [REF] waptconsole: Introduced an interface for uwaptpackage TWaptPackage WIP: fix compilation when *USE_WAPTPACKAGE* is defined TODO: implement IX509Store
- [FIX] Fixed waptconsole: fixed host overview layout if no html template.
- [UPD] waptconsole: host details layout changes: introduced html templates based overview if `templateshost_overview.html` file exists (mustache template).
- [FIX] Fixed waptconsole *sendmessage* gui splitter.
- [IMP] waptconsole: check that downloaded waptsetup version is the same or newer than that of the WAPT Server.
- [FIX] Fixed autosearch in `ttissearch` component.
- [NEW] waptconsole: Added a popumenu *copy to clipboard as json for audit data*.
- [IMP] waptconsole: allow drag & drop of a audit *`json`* value subkey from value tree explorer.
- [NEW] waptconsole: displays audit history and WIP audit data explorer (treeview + html template).
- [FIX] Fixed reporting queries grid layout not saved properly.
- [UPD] GUI Vis ACL: zebra colored lines and added possibility to change user password from one button (same action like in right click on user).
- [FIX] Fixed avoiding exception if no user was selected before adding ACL rights.
- [FIX] Fixed trigger downloads when triggering updates from the WAPT Console (missing import).
- [UPD] Updated icons on windows update status for WUA.
- [FIX] Fixed waptconsole check external repository version timeout exception raised in frontend.
- [FIX] Fixed waptconsole multiserver: fixed can't trigger action on servers other than main WAPT Server.
- [FIX] Fixed waptconsole: Avoid error message of no `repo_url` for last used package template section.
- [FIX] Fixed modifying a password if old password was empty.
- [ADD] Hide / show all columns in grids.

- [NEW] new option `check_package_version` in `waptconsole.ini`.
- [UPD] waptconsole reporting: Added a quick search filtering zone for the query result.
- [FIX] Fixed wrong message when no admin rights and the WAPT Agent needs to be upgraded or is not present.
- [UPD] Host menu for upgrading hosts part.
- [REF] waptconsole multiserver: Refactored `TriggerActionOnHosts` to send multiples actions to the right servers.
- [FIX] Fixed waptconsole: use ROOT in addition to CA windows system certificates stores when building **winpe** with `verify_cert` = **True**.
- [UPD] Deleted host popup.
- [NEW] Possibility to download WAPT packages when asking hosts for updates.
- [UPD] `trigger_host_update` adding possibility to download the WAPT package after update.
- [FIX] Fixed waptconsole: The WAPT Console crashed when checking newest packages if wapt-templates repository is protected with an encrypted client key.
- [FIX] Fixed saving configuration when new configuration was created.
- [FIX] Fixed saving language parameter.
- [FIX] Fixed waptconsole: access violation when access to external repository is blocked or needs a proxy.
- [FIX] Fixed waptconsole multiserver regression: unable to edit a WAPT package which was just edited.
- [FIX] Fixed waptconsole edit conf package: Do not close if error when uploading to the WAPT Server.
- [FIX] patched `setup_package_template_cert.py.tmpl`.
- [FIX] Fixed not adding « cn » in OU.
- [FIX] Fixed layout on Windows Update part.
- [FIX] Fixed the flow layout.
- [IMP] waptconsole: WIP multiserver. Mostly works for hosts, but not for packages management.
- [FIX] Fixed waptconsole: re-enable dataexport to *.csv* for grids.
- [NEW] Explicit hint on number version when the WAPT package is not up to date (GridPackages).
- [REF] Refactored private key password handling. Added a callback to clear cached key password in case of decrypt error in http client. Stores client https authentication key password in same storage as package private keys.
- [REF] WIP for multiserver console. WaptCookieManager takes in accounts the domain. TODO: send allowed session cookies for cross domain auth. Lazy loading of waptserver instance. Loads list of servers in `DMWaptConsole.ReloadConfigFile`. All sections with a `wapt_server` key are taken in account. Shares the WaptServerSession across all waptserver connections.
- [FIX] Fixed bad port for **veyon**.

WAPT Server

- [IMP] waptserversetup: Check if there is a CRITICAL log entry during `winsetup.py` and exit with an exitcode 1000 if it is the case.
- [IMP] waptserver: Do not automatically create users in wapt database when user logs in with kerberos (self-service case).
- [FIX] Fixed waptserverinstall windows: regression unable to install on new windows machine if wapt was not already installed.
- [REF] Server python code cleanup.
- [IMP] wapttasks: use environment variable on linux to pass `config` file path.
- [NEW] waptserver: reduced lifetime of session cookie to default 12h. `session_lifetime` can be changed in `waptserver.ini` using `session_lifetime` seconds parameter.
- [UPD] Updated to python 3.8.16 for all supported operating systems.
- [FIX] Fixed stuffed setup exe naming on the WAPT Server.
- [NEW] new parameter `list_subnet_skip_login_wads`.
- [FIX] Fixed waptserver: shorten SQL columns aliases for long `get_hosts` *`json`* queries.
- [SEC] Upgraded **`werkzeug`** 2.0.2 -> 2.1.1 for PYSEC-2022-203.
- [NEW] waptservice websocket: Enabled certificate checking on websockets.
- [IMP] waptserver: Added index on `computer_ad_ou`.
- [FIX] Fixed waptserver: by default, do not create stuffed `waptsetup` when a dynamic config is uploaded.
- [FIX] Fixed waptserversetup: if installService, configure the local service to reach newly installed server. Propose to start the WAPT Console right after for demo mode.
- [NEW] `model.py`: Added `upgrade-db` action and **`--overwrite-version=1.2.3`** option to force the replay of upgrade db.
- [FIX] Fixed waptserver **`nginx`** config, there can be spaces in path. quotes include.
- [NEW] Be sure to not start the WAPT Server if the database structure can not be upgraded properly.
- [NEW] If licences *`json`* data is empty, assume an empty list.
- [IMP] Getting storage used by KBs.
- [NEW] 22H2 build numbers in WindowsVersions class.
- [NEW] Added `hosts_sid` endpoint routing to uwsgi in nginx configuration templates.
- [FIX] Fixed **`wapt-get build-waptagent`**: create **`waptagent.exe`** link on the WAPT Server.
- [FIX] Fixed waptserver: ignore null bytes in audit data string values.
- [FIX] Fixed waptserver: allow access to agent download without client certificate auth.
- [FIX] Fixed waptserver model: remove references to unused HostExtData table.
- [FIX] Fixed waptserver multiinstance with uwsgi: takes in account `application_root` for interprocess `get_ws_connections` /api/v3/hosts_sid calls.
- [UPD] Added waptserver /api/v3/update_hosts_sid_table endpoint to fill the HostWebsocket table with the in memory ws_connections for reporting purpose.
- [UPD] Changed the path of the untouched **`waptsetup.exe`** on the WAPT Server: moved to the `wapt/waptagent` folder to be consistent with other agents location Same for **`waptdeploy.exe`**.

- [DEL] waptserver: Removed « enable_store » setting.
- [UPD] waptconsole multiserver: display unreachable servers.
- [FIX] Fixed waptserversetup: Reinclude waptwua even if service is not installed to allow **wapt-get** usage.
- [FIX] Fixed waptconsole multiserver dynamic config: bad server url for checking https certificate.
- [FIX] Fixed waptconsole multiserver: Do not include a server at startup if it is not pingable.
- [UPD] waptserversetup windows: Removed some additional unused files when waptservice is not installed.
- [UPD] waptconsole multi servers: Do not try to update / merge repo if `repo_url` is empty.
- [IMP] waptserver / waptservice websockets: When registering host, return an authentication token in response, so that websockets can connect without additional roundtrip.
- [IMP] `allow_unauthenticated_registration` is now like `use_kerberos`.
- [FIX] Postconf, current config is now autoselected.
- [UPD] waptsetup waptserversetup: Sign the installers and uninstallers using embedded iscc logic.
- [UPD] waptserver db: Changed the primary key of tables *HostPackagesStatus*, *HostExtData*, *Packages*, *HostSoftwares*, *HostGroups*, *HostWebsocket*, *HostAuditData*, *ReportingSnapshots*, *HostWsus*, *LogsAPI* to bigint.
- [UPD] waptserversetup: Check that the user is a LOCAL computer user and not a domain user.
- [FIX] Fixed waptserversetup: postgresql upgrade. Try to fix ACLs on data directory.
- [FIX] Added a conflict on apache2 in the Linux WAPT Server package to avoid old install leftovers.
- [REF] Removed `enterprise_common.py`.
- [UPD] Upgrade **nginx** on Windows.
- [UPD] Upgraded DB to **postgresql v14** for windows.
- [UPG] upgraded **postgresql** 9.6 to **v14** on CentOS7.
- [FIX] Fixed waptserver: Fixed sid map sharing in uwsgi mode (missing imports).
- [FIX] Fixed waptserver websocket: Be sure to not clear a SID which would be newer than current disconnect event. Not sure if disconnect / reconnect are always synchronous.
- [FIX] Fixed waptserver: Improved message when triggering action.
- [IMP] Added HTST header to nginx template.
- [FIX] Fixed waptserver `update_hosts_audit_data`: Updated values with same global key (host_id,value_id).
- [FIX] Added **trigger_host_action** ACL on /api/v3/connected_wol_relays (used by /api/v3/trigger_wakeonlan).
- [IMP] waptserver websocket auth: Put host certificates in cache.
- [UPD] waptserver websocket: Do not cache UUID twice.
- [REF] waptserver websockets: use a global in memory dictionary to hold the host uuid -> SID of connected host to avoid Database insert or updates.
- [FIX] Fixed server regression for custom *json* fields ValueError: too many values to unpack (expected 3).
- [IMP] waptserver: WIP endpoint `update_hosts_audit_data` to bulk insert hosts related data.
- [IMP] waptserver: update api/v3/get_agents_info to match the online `wapt_agent_list.json`.
- [FIX] Fixed glpi sync: simplified `glpi_upload_hosts.py` script.

- [FIX] Fixed waptserver huey tasks: `licences_list` not properly initialized when not using default `waptserver.ini`.
- [FIX] Fixed waptserver audit table structure upgrade: typo
- [FIX] Fixed avoiding GET method limits on `hosts_for_wua`.
- [FIX] Fixed waptserver unable to delete some hosts when CRL is enabled be tolerant if the host certificate is not issued by this server's CA.
- [FIX] Fixed waptconsole multiserver: Computers identified by fqdn uuid are not displayed properly in the grid.
- [UPD] Remove references to **`waptsetup-tis.exe`** -> renamed to **`waptsetup.exe`**.
- [FIX] Fixed `update_server_status` with dynamic configuration.
- [IMP] Include **`waptsetup.exe`** in **`waptserversetup.exe`**.

WADS

- [REF] Remove useless code on `get_wads_config` (Login WADS).
- [IMP] WgetWads does not require python to work.
- [FIX] Be more indulgent on *`json`* rules for WADS.
- [FIX] Fixed WADS working when no logging required.
- [ADD] Login in IPXE, more tests needed.
- [IMP] Proper way to secure `wads_get_config`.
- [ADD] Login on WADS register host and get wads configuration.
- [NEW] include hostname in `debian.ipxe` for OS deployment.
- [FIX] Fixed `djoin` with given domainuser parameter.
- [IMP] Added back support GET method on /api/v3/get_wads_config.
- [NEW] Added asset tag in HostOSDeploy.
- [IMP] Ask for a new hostname when starting to deploy if hostname equals to "autoregister".
- [IMP] Improved filtering keyboard faster + french translation in *Make WinPE*.
- [FIX] Fixed missing glob import in WADS `get_iso_config`.
- [NEW] Adding drivers in WinPE from WADS drivers.
- [IMP] Improved feedback when the `djoin` fails (already existing machine).
- [WADS] <Value> format in *`XML`* was incorrect and not complete for password definition.
- [IMP] Last error message added for failed `djoin`.
- [FIX] Fixed uninstall **`wapttftpserver`** when uninstalling **`waptserver`**.
- [IMP] Improved file upload with hash check wads *`iso`* files listed from the WAPT Server even if not saved in the WAPT Console.
- [NEW] Added customized WinPE export to zip file.
- [IMP] Improved showing the error message on upload failure.
- [IMP] Improved applying default configuration on wads host if no configuration has been set.
- [IMP] ISO download dialog box.

- [IMP] WADS: WinPE now pinging WAPT Server. Selected language keyboard layout will be available directly in a new cmd.
- [IMP] WADS: XML no longer disable UAC by default.
- [FIX] Fixed `mac_address` not returned with iPXE.
- [ADD] Added `ipxe_script_jinja_path` and two templates.
- [UPD] Added file type filters for loading the post-install script.
- [FIX] Restored a progression bar when uploading the *ISO* and the *winpe* files.
- [IMP] kill **wapttftpserver** and uninstall the service before installing it.
- [ADD] Added Windows 11 unattend *XML* template files.
- [IMP] Improved searching WADS data (hosts, isos, driver bundles, configurations).
- [FIX] Added tftp **firewalld** port opening.
- [IMP] Avoid creating WinPE on Windows partition + some ACL added.
- [UPD] Renamed drivers bundle filenames to sha256(filename).
- [ADD] Added a template for Debian.
- [UPD] *GridConfigDeploy* showing the platform now.
- [FIX] Fixed saving bundle names.
- [NEW] Injecting a:abbr:*OEM (Original Equipment Manufacturer)* key by **slmgr** command.
- [FIX] Fixed SELinux rules for wads.
- [FIX] Potential fix for (over 10 joins for djoin by a standard user on MSAD).
- [UPD] WADS grayed when windows update repository is selected.
- [UPD] Possibility to select an *iso* file even if not Windows.
- [FIX] Fixed waptconsole `uploadWinPE`: regression in upload progress bar and incomplete zip.
- [FIX] Fixed wads to include non CA certificates for WinPE build.
- [IMP] Added `ipxe_script` in DeployConfig table.

WAPT Agent MacOS

- [UPD] Delete old *pkg* if available in *pkg* list.
- [NEW] Added fake menu for macOS for letting user to quit the app from the *MainMenu*.
- [FIX] Improved support for macOS *MenuBar*.
- [FIX] Added WAPT Console *.app* plist file for macOS X.
- [FIX] Fixed some macOS X model and serial number reports.
- [FIX] Fixed macOS X `local_groups` key in `host_info`.
- [FIX] Updated mormot2 for **gssapi** on macOS X.
- [NEW] support WADS security, Network masks.
- [FIX] Fixed `installed_softwares` on MacOS.
- [NEW] Added timestamping to *pkg* signing.

- [FIX] Fixed getting agent version in `get_wads_config`.
- [NEW] Added entitlements file for macOS signing.
- [IMP] Force light UI when DarkMode is active on macOS.
- [FIX] Fixed opening maximized self service on macOS
- [FIX] Fixed loading hosts on macOS when *more options* in inventory is checked.
- [IMP] Better handle on input (utf8 convertion).
- [IMP] macOS: Updated build script to handle binary file signing and better debugging.
- [IMP] Patched `dmidecode` info for macOS.
- [FIX] Fixed macOS core `get_hostname` return binary string instead of str -> update_status loop.
- [IMP] Use `system_profiler_info` for `dmi_info` on macOS X.
- [REF] `plistlib.readPlistFromBytes` deprecation fix.
- [FIX] Fixed core macOS: use UUID from `system_profiler_info` instead of `dmidecode`.
- [FIX] Fixed duplicated macOS code in setuphelpers for `get_hostname()`.
- [IMP] Improved mounting content for *`.pkg`*, *`.mpkg`*, *`.app`* only if file is not symbolic.
- [NEW] Added the WAPT Console to Linux and macOS gui distribution.
- [IMP] Fixed keyword and name with `installed_softwares` in macOS and Linux.
- [FIX] Fixed register for macOS.
- [FIX] Fixed custom waptmessage logo linux.
- [FIX] Fixed **`harakiri`** on non Windows kills all running processes.
- [FIX] Fixed restart waptservice for macOS.
- [IMP] Silently attach *`dmg`* file.
- [FIX] Fixed `get_file_type` in macOS.

WAPT Agent Linux

- [FIX] Fixed `user_local_appdata` for Linux.
- [IMP] waptagent Debian package: removed system python3 dependency.
- [IMP] Avoid loop in checkbox events on search inventory especially on operating systems other than Windows.
- [IMP] Added `PYTHONNOUSERSITE` = **True** to all *`.sh`* scripts to avoid spoiling PYTHONPATH with locally installed libraries in user home directory.
- [UPD] Disable compression on unix WAPT agent bundle (each package is itself already compressed).
- [NEW] Added the WAPT Console to Linux and MacOS gui distribution.
- [FIX] Fixed *configpackage* wizard and main form layouts for Linux.
- [UPD] Updated virtualtreeview for Linux visual grid lines improvements.
- [IMP] Fixed keyword and name with `installed_softwares` in macOS and Linux.
- [FIX] Fixed **`harakiri`** on non Windows kills all running processes.

- [ADD] Added snap software inventory.
- [FIX] Fixed waptservice linux restart Linux: AttributeError: `WaptServiceRestart` object has no attribute *logger*.
- [NEW] Linux OS deployment.
- [FIX] Added **`firewalld`** rule on RedHat based server for **`wapttftpserver`**.

62.2 WAPT-2.2 Serie

62.2.1 WAPT-2.2.3.12481 (2022-11-30)

hash: ad3855c9

This is a security release with a few related bugfixes. All WAPT 2.0 versions below 2.2.3.12481 are affected.

Note: if you are using WAPTAgent deployment via GPO, do not forget to update your waptdeploy binary in the definition of the GPO.

WAPT Core

- [SEC] Upgraded **`python`** from 3.8.13 to 3.8.15.
- [SEC] Upgraded **`openssl`** from 1.1.1k to 1.1.1s.
- [SEC] Upgraded WAPT Agent kerberos lib from 1.19.3 to 1.20.1 (Linux / macOS).
- [SEC] Upgraded python modules with CVEs:
  - pylint==2.12.2 -> 2.15.6.
  - ujson==4.0.2 -> 5.5.0.
  - waitress==2.0.0 -> 2.1.2.

WAPT Agent

- **[SEC] waptdeploy.exe: Use only wapt_is1 install location from registry to get the current wapt installation directory.** Do not run **`wapt-get`** to check working condition.
- [FIX] Added fallback method to get domain in `get_hostname`.
- [FIX] Fixed windows, replaced **`wapt-get.exe --hide`** by **`waptpythonw.exe wapt-get.py`** to run **`session-setup`** because **`--hide`** does not actually hide the shell window.
- [FIX] Fixed WakeOnLAN relays.
- [REF] Cleaned up the WAPT Agent `common.py`: removed unused imports.
- [FIX] Fixed waptexit: fix `only_priorities` argument when starting waptexit from service.
- [IMP] MacOS: Updated build script to handle binary file signing and better debugging.

WAPT Console

- [UPD] WADS: Include hostname in template iPXE Debian Linux.
- [IMP] WAPT Console: Do not display empty confirmation messagebox.

WAPT Server

- [FIX] waptserver postconf: Force path when running **psql** command in postconf (linux).

62.2.2 WAPT-2.2.3.12463 (2022-09-29)

hash: fc306143

This release is mainly a bugfix release. The main new feature is tech-preview support for MacOS on Apple M1 architecture.

Note :

- due to EOL and security issue, the PostgreSQL database version has been updated on the WAPT Server for Windows and Redhat7 from version PostgreSQL 9.6.24 to PostgreSQL 14.5. The upgrade will be automatic on Windows during waptserversetup.exe install, and is done during postconf.sh run on Redhat7. Be sure to run the postconf.sh script after upgrading.

WAPT Server

- [UPD] WAPT Server for Redhat7 / Centos7: Upgraded **PostgreSQL** version from 9.6 to 14.5.
- [UPD] WAPT Server for Windows: Upgraded **nginx** to 1.22.0.
- [UPD] WAPT Server for Windows: Upgraded **vcredist** to 2022.
- [UPD] WAPT Server for Windows: Upgraded **PostgreSQL** version from 9.6 to 14.5.
- [FIX] WAPT Server for Windows: Fixed **icacls** for `migrate_pg_db`.
- [FIX] WAPT Server for Windows: Allow install and upgrade with any server admins (does not require to use the local Administrator with RID -500 for installing).
- [UPD] WAPT Server for Windows: waptserversetup: avoid automatic restart when installing MSVC 2022.
- [FIX] Fixed upgrade procedure: migrate data text to *jsonb* only if table hostauditdata in `data_type` text.
- [FIX] Patched `create_default_users` when upgrading from 1.8.2 to 2.2.
- [FIX] Fixed unhandled redirections in TWaptServer wget.
- [FIX] Added `RedirectMax` parameter in WaptServer WGet
- [UPD] Added ubuntu 22.04 in waptagent bundle.
- [FIX] Fixed postconf nginx: bad error string format.

WAPT Console

- [FIX] Fixed host configuration package that were not editable right after creating them.
- [FIX] Fixed error editing same OU package in one session.
- [FIX] Fixed `CleanupPackagesCache` proper unlock even if no assigned package.
- [FIX] Fixed access violation at startup when no server is defined in `waptconsole.ini` file.
- [FIX] Fixed waptconsole: When deleting a package in the *private repo* page, package is still listed until the WAPT Console is restarted, but the package is actually deleted on the WAPT Server.
- [FIX] Fixed waptconsole: Random timeout error when running commands from waptconsole

WAPT Agent

- [FIX] Fixed setuphelpers: reintroduce `running_as_system` for Linux and macOS (uid==0).
- [FIX] Fixed start waptservice only if `wapt-get.ini` configuration exists.
- [FIX] Fixed `remove_file()`: Was unable to remove symlinks.
- [FIX] Reset properly Wapt core settings to default when reloading config from `wapt-get.ini`.
- [FIX] Try to create a minimal `wapt-get.ini` file if it does not exist so that the service can be started without any prior configuration.
- [FIX] Fixed WAPT Agent for macOS: use `system_profiler_info` for `dmi_info` on macOS for support for Apple m1 architecture.
- [FIX] Fixed WAPT Agent for macOS: `plistlib.readPlistFromBytes` deprecation fix.
- [FIX] Fixed WAPT Agent for macOS: core macOS: use UUID from `system_profiler_info` instead of `dmidecode`.
- [FIX] Fixed WAPT Agent for macOS: change postinst script for `launchctl` compatibility.
- [FIX] Fixed WAPT Agent for macOS: macOS core: `get_hostname` returned binary string instead of str -> `update_status` loop.
- [IMP] Fixed WAPT Agent for macOS: Rationalize *pkg* filename.

62.2.3 WAPT-2.2.3.12454-rc2 (2022-09-26)

hash: 64bfc946

This is the second release candidate for WAPT 2.2.3.

The main new feature is tech-preview support for MacOS on Apple M1 architecture. Otherwise it is mainly a bugfix release.

Note :

- due to EOL and security issue, PostgreSQL database version has been updated on WAPT Server for Windows and RedHat7 from version PostgreSQL 9.6.24 to PostgreSQL 14.5. Upgrade will be automatic on Windows during waptserversetup.exe install, and is done during postconf.sh run on Redhat7. Be sure to run the postconf.sh script after upgrade.

Fixes since WAPT-2.2.3-rc1:

WAPT Server for Windows

- [FIX] Fixed **icacls** for migrate_pg_db.

WAPT Agent

- [FIX] Start waptservice only if wapt-get.ini config is exists
- [FIX] Added PYTHONNOUSERSITE = **True** to all *.sh* scripts to avoid spoiling PYTHONPATH with locally installed libraries in user home directory.
- [FIX] Fixed remove_file() that was unable to remove symlinks.
- [FIX] Fixed waptconsole : fix AV at startup when no server is defined in *ini* file.

WAPT Agent for macOS

- [FIX] Use system_profiler_info for dmi_info on macOS for support for Apple m1 architecture.
- [FIX] Fixed plistlib.readPlistFromBytes deprecation.
- [FIX] Fixed core macOS: use uuid from system_profiler_info instead of dmidecode
- [FIX] change postinst script for launchctl compatibility
- [FIX] macOS core get_hostname return binary string instead of str -> update_status loop
- [IMP] rationalize pkg filename

62.2.4 WAPT-2.2.3.12411-rc1 (2022-09-05)

hash: 29e18f23

This is mainly a bugfix release.

Note :

- due to EOL and security issue, PostgreSQL database version has been updated on WAPT Server for Windows and Redhat7 from version PostgreSQL 9.6.24 to PostgreSQL 14.5. Upgrade will be automatic on Windows during waptserversetup.exe install, and is done during postconf.sh run on Redhat7. Be sure to run the postconf.sh script after upgrade.

WAPT Server

- [UPD] WAPT Server for Redhat7 / Centos7 ! upgrade PostgreSQL version from 9.6 to 14.5
- [UPD] WAPT Server for Windows : upgrade nginx to 1.22.0
- [UPD] WAPT Server for Windows : upgrade vcredist to 2022
- [UPD] WAPT Server for Windows : upgrade PostgreSQL version from 9.6 to 14.5
- [FIX] WAPT Server for Windows : allow install and upgrade with any server admins (does not require to use the local Administrator with RID -500 for install)
- [UPD] WAPT Server for Windows : waptserversetup: avoid automatic restart when installing MSVC 2022
- [FIX] fix upgrade procedure : migrate data text to jsonb only if table hostauditdata in data_type text

- [FIX] patch create_default_users when upgrading from 1.8.2 to 2.2
- [FIX] Fix unhandled redirections in TWaptServer wget
- [FIX] Add RedirectMax parameter in WaptServer WGet
- [UPD] added ubuntu 22.04 in waptagent bundle

WAPT Console

- [FIX] host config package are not editable right after creating them.
- [FIX] error editing same OU package in one session
- [FIX] CleanupPackagesCache proper unlock even if no assigned package

WAPT Agent

- [FIX] setuphelpers. reintroduce running_as_system for linux and mac (uid==0)

62.2.5 WAPT-2.2.2.12388 (2022-07-22)

hash: 10e35aa7

This is mainly a bugfix release.

Note:

- There is a change in the wapt the wapt->glpi sync is working, please refer to documentation for upgrade.
- Tech preview: new multiserver console support (connect to multiple wapt server using one console).
- Added support for ubuntu 22.04 amd64.
- `def update_package()` function can now be located in a separate `update_package.py` file. New packages from wapt store will use this new format to make `setup.py` simpler and more readable. Older wapt version are not impacted for package import and package install, but may be impacted if one wants to update directly from the WAPT Console using `update_package` script.

WAPT Deployment Server (WADS)

- [NEW] injecting oem key by slmgr command
- [FIX] fix tftpserver window size handling (bug on Dell uefi bios)
- [FIX] allow djoin with machine in default container CN=computers
- [FIX] improve error message when using standard user on MS AD for djoin.exe when >10 machine quota join has been reached
- [FIX] allow saving / renaming bundle names and check for empty names
- [IMP] add ACL on WADS (before it needed admin level ACL)
- [NEW] add post_install script windows
- [NEW] add ignore_ipxescript and move conf file and ipxescript

- [NEW] Basic Linux OS Deploy support : add Debian ipxe script template
- [NEW] add {{server_url}} {{secondary_repo}} and {{hostname}} in get_wads_config
- [NEW] add mustach templating in ipxescript
- [FIX] waptconsole uploadWinPE : fix regression in upload progress bar and incomplete zip.
- [FIX] add a progression form when uploading ISO and winpe
- [IMP] add wapttftpserver service shutdown in upgrade sequence (throught net stop, not only taskkill)
- [IMP] add tftp firewalld port opening on Redhat

WAPT Console

- [NEW] techpreview: waptconsole reporting multiservers.
- [FIX] Fixed check that downloaded waptsetup version is same or newer than server.
- [NEW] Download from https://wapt.tranquil.it and upload on local waptserver agents for Linux and macOS directly from the WAPT Console.
- [NEW] Added a popupmenu *Copy to clipboard* as `json` for audit data.
- [NEW] Display audit history audit data explorer (treeview + html template) + allow drag/drop of a audit `json` value subkey from value tree explorer.
- [IMP] waptwua: update waptwua status to *NEED-SCAN* on hosts when `download_wsusscan` is triggered and `wsusscn2.cab` file is downloaded.
- [IMP] Package import: Don't take care anymore of maturity for version when it's compared to store version.
- [FIX] Added licence validity check tolerance +1 day.
- [FIX] Fixed trigger downloads when triggering updates from the WAPT Console.
- [FIX] Allow ~ in package names (for spaces in Organizational Unit packages).
- [UPD] Updated icons on windows update status for WUA.
- [NEW] New option `check_package_version` in `waptconsole.ini`.
- [FIX] Fixed saving empty value in Editor for packages.
- [UPD] waptconsole reporting: Added a quick search filtering zone for the query result.
- [FIX] Wrong message when no admin rights and waptagent need upgrade or not present.
- [UPD] When going outside modified rules. A popup will ask to save or not the rules.
- [UPD] Delete host popup.
- [NEW] Added feature to download packages when asking hosts for update.
- [UPD] `trigger_host_update` adding possibility to download the package after update.
- [FIX] Saving language parameter.
- [UPD] Added a *NEED-SCAN* waptwua.status, updated when Wapt.update() is called.
- [FIX] Fixed layout on Windows Update form.
- [NEW] waptconsole: multiserver: manage packages repositories by server.
- [FIX] waptconsole: re-enable dataexport to `csv` for grids.

- [NEW] Explicit hint on number version when the package is not up to date (GridPackages)
- [UPD] waptconsole: Improved drag drop of columns into GridHosts
- [NEW] waptconsole: New Htmlviewer for audit data and Html auditdataview template filename (`wapttemplates`) calculated from section and key, or section.
- [FIX] waptconsole drag/drop audit values.
- [IMP] waptconsole: Load Active Directory Groups in thread.
- [FIX] waptserver: Improved message when triggering action.

WAPT Server

- [FIX] glpi sync: simplified `glpi_upload_hosts.py` script.
- [NEW] techpreview waptserver: endpoint `update_hosts_audit_data` to bulk insert hosts related data (for third party data integration).
- [NEW] Added multiserver endpoint for multiserver WAPT Console.
- [FIX] waptserver update_audit_data fix on_conflicts for value_id.
- [IMP] waptserversetup: take in account wapt_folder parameter in `waptserver.ini` when upgrading a setup.
- [IMP] Use utc time for acls expiration check.
- [FIX] Fixed waptserver unable to delete some hosts when CRL is enabled.
- [IMP] waptserver db install: try to register *`jsquery`* extension to make *`json`* query more powerful for reporting (this is not yet mandatory).
- [IMP] Renamed **`waptsetup-tis.exe`** to **`waptsetup.exe`** on the WAPT Server.
- [IMP] Include **`waptsetup.exe`** in **`waptserversetup.exe`** on Windows.
- [IMP] Download from TIS / upload to the WAPT Server of the installation packages of the WAPT Agents.
- [UPD] Create a full version 1.2.3.rev-hash into file `wapt/version-full`
- [IMP] Added HTST header to nginx template.
- [DEL] Removed direct integration of GLPI sync into WAPT. Now switched to plugin sync
- [FIX] Added **`trigger_host_action`** ACL on /api/v3/connected_wol_relays (used by /api/v3/trigger_wakeonlan)
- [IMP] Force calc_md5 if new filename in server.
- [IMP] Improved websockets performance and reliability. Now websocket ids are stored in memory instead being written in the database.

WAPT Agent

- [FIX] Fixed threading exception in **WAPTExit** and **WAPTTray** that could prevent status updates.
- [NEW] WAPTWUA superseded support. option *include_potentially_superseded_updates* in configuration wizard.
- [NEW] Added snap software inventory.
- [FIX] waptmessage unable to load sqlite on Linux and macOS.
- [FIX] Fixed custom waptmessage logo on Linux.
- [FIX] Fixed waptservice configuration: sets the `configs_dir` relative to `wapt-get.ini` full path.
- [FIX] Fixed waptservice "start_waptexit" with arguments
- [FIX] Fixed bad arguments sent to waptservice triggering upgrades with "only_priorities" and "only_if_not_process_running"
- [FIX] `Wapt.write_audit_data_if_changed`: writes data if previous data has expired.
- [IMP] **`wapt-get add-config-from-url`**: provides a meaningful message when hash is not provided.
- [FIX] Updated the template of dynamic *`json`* configuration packages to match the new location and the naming of *`json`* config related functions.
- [IMP] Improved dynamic configuration handling for the WAPT Agent.
- [FIX] waptservice: ensure a random secret_key for local waptservice session.
- [FIX] **`wapt-get update-package-sources`**: handles properly relative path to package sources.
- [IMP] **`wapt-get edit`** now opens changelog.txt, VSCod* now open `control` file too.
- [UPD] Changed default log path to `wapt/log` if writable.
- [IMP] waptservice waptself: local authentication with file token (ie. nopassword), handling of local groups.
- [NEW] use `--not-interactive` with **`register`** if install run in silent mode and not run update if install service.
- [IMP] waptself, wapt-get, waptexit, wapttray: kill check threads on close, even on linux to speed up application shutdown.
- [FIX] Linux: waptservice restart Linux: AttributeError: "WaptServiceRestart" object has no attribute "logger".
- [IMP] macOS: normalize macos wapt install package name format.
- [FIX] macOS: Fixed registration failing in some cases.
- [IMP] macOS: Added mpkg support.
- [FIX] Fixed no hash in clipboard, added missing helper for `add-config-from-url` in **`wapt-get`**.
- [IMP] Limit access right to admins to log directory (in case non public stuff get written to log)

WAPT Core

- [IMP] Patched with_md5sum in `make_package_filename`.
- [IMP] Added options for update-package-sources.
- [UPD] wapt core: use datetime in UTC for `audit_data`.
- [NEW] wapt core: allow usage of an environment variable « waptbasedir » to specify the location of root waptbasedir.
- [FIX] configuration package template `setup_package_template_conf.py`.

- [IMP] Support for `def update_package` in file `update_package.py` instead of `setup.py` for better readability.
- [UPG] Upgraded **openssl** to 1.1.1o.
- [NEW] core: define path Wapt.configs_dir relative to Wapt.config_filename if the dir Wapt.config_filename..conf.f exists.
- [FIX] Fixed `waptcrypto`: certificate filename attribute was not set when loading a certificate chain.
- [FIX] Fixed new option copytree2 replace_at_next_reboot.
- [FIX] Avoid errors on `get_version_from_binary()` getting params.
- [FIX] Fixed keyword and name with installed_softwares in macOS and Linux.

62.2.6 WAPT-2.2.1.11957 (2022-06-02)

WAPT Deployment Server (WADS)

- [FIX] Fixed **wapttftpserver** restart on Linux.
- [IMP] Added *xml* template for windows 11 deployment.
- [FIX] if `verify_cert` is empty, then `verify_cert = False`.

WAPT Console

- [FIX] CheckLicence => licence is now valid one day before the real beginning.

WAPT Agents

- [FIX] Fixed **harakiri** on Linux.

62.2.7 WAPT-2.2.1.11949 (2022-05-18)

hash: 1b2dfbee

This is a bugfix release.

WAPT Deployment Server (WADS)

- [FIX] Fixed waptconsole: use ROOT in addition to CA windows system certificates stores when building winpe with `verify_cert = True`.
- [FIX] Fixed selinux rules for WADS.
- [FIX] Fixed non ascii character support in passwords.
- [IMP] wgetwads: add more logging data (wget). Disable exe signature certificate as this could be blocking if CRL can not be checked in winpe environment for example.
- [UPD] add a timer to wait for network in WADS.
- [UPD] Update **openssl** to 1.1.1n for WADS.

Other fixes

- [FIX] fix wrong GPO link on waptserver start page
- [FIX] fix some translation messages in console
- [FIX] wrong element order in message in ACL GUI
- [FIX] allow change password if user password has been cleared
- [UPD] update mormot2 for bug in TSynDictionary.AddOrUpdate()
- [UPD] update mormot statics for sqlite to 3.38.5 (required for mormot compatibility)

62.2.8 WAPT-2.2.1.11932 (2022-05-05)

hash: 6522dccb

This is a bugfix release.

WAPT Deployment Server (WADS)

- [FIX] wapttftpserver : better handling of UEFI PXE/TFTP boot
- [FIX] wads now include non CA certificates for winpe build
- [FIX] Not adding « cn » in OU
- [FIX] wapttftpserver : add firewalld rule on redhat based server for wapttftpserver
- [FIX] WADS : improve feed back on upload WinPE
- [FIX] wapttftpserver : kill wapttftpserver and uninstall service before installing it
- [IMP] waptserversetup: add wapttftpserver configuration for windows

WAPT Server

- [FIX] fix typo for rocky support as server
- [FIX] waptservice websocket reconnection: disable by default low level reconnect feature

WAPT Console

- [FIX] fix bad port configuration for veyon remote assistance support
- [FIX] Define default package prefix when creating empty package
- [FIX] patch setup_package_template_cert.py.tmpl
- [FIX] waptconsole: fix access violation when access to external repo is blocked or need a proxy.
- [IMP] package version in bold red if obsolete version compared to external repo for better accessibility

WAPT Agent

- [FIX] waptservice websocket reconnection: disable by default low level reconnect feature
- [FIX] add conf.d to rpm agent installers for the new agent configuration management
- [FIX] macOS: fix get_file_type in macos
- [IMP] macOS: silently attach dmg file
- [IMP] waptwua : improve consistancy between WUA history and WUA status
- [FIX] waptself: bad char case for png file (issue for linux)
- [IMP] add dummy running_on_ac for linux and mac for compatibility
- [FIX] waptutils.user_config_directory() did not work under system account.

WAPT Core

- [IMP] mormot2 static: add 3.38.2 hash
- [IMP] sync htmlviewer with latest github commits from https://github.com/BerndGabriel/HtmlViewer/tree/master
- [IMP] waptguihelper: improved the design for InputDialog form

62.2.9 WAPT-2.2.1.11899 (2022-04-06)

hash: 2d82654e

This is mainly a bugfix release.

A new tftpserver has been introduced and it will ease WADS installation and configuration as it will be directly integrated into WAPT.

WAPT Deployment Server (WADS)

- [NEW] add a wapttftpserver binary on windows and linux to act as a tftp server for WADS
- [FIX] WADS : don't use redirect
- [FIX] WADS : be tolerant if sendstatus can not be sent.
- [IMP] WADS : handle https for drivers (continued)
- [UPD] wads : get windows system certificates for WADS server bundle
- [UPD] implement https verifyCert in wads and wgetwads
- [IMP] add serial_number arg when calling server get_wads_config in wads
- [UPD] waptconsole wads: add audit columns (created/updated) in grids.
- [NEW] Add an action to prepare a host package in WADS OS Deploy grid
- [NEW] wgetwads : use code signing cert of TIS to check signature of json hashes file if no signer_certificate in json file

WAPT Console

- [UPD] OU « All » fixed to not editable on GridOrgUnits
- [FIX] waptconsole: wrong client https key password used for task polling thread.
- [FIX] waptwua packages : ALLOWED status in winupdates grid is kept between form display.
- [FIX] Package creation did not take silent flags in account
- [FIX] memory leak when refreshing packages list
- [FIX] waptconsole packages list: Showing all versions when « Last version only » is not checked
- [FIX] « property not found » in some grids when refreshing data.
- [FIX] running plugins on multiple hosts.
- [FIX] taking in account the platform when lookig for TIS store package version
- [FIX] nested progress notifications in uwaptserverconnection TWaptServer
- [FIX] Disabled pysources check at waptconsole startup.
- [FIX] external repo ini settings dialog when importing.
- [FIX] waptconsole. some ui elements are not disabled when switching to discovery on login.

WAPT Server

- [NEW] add support for postgresql 14 on centos7
- [UPD] wapt windows server: update to nginx 1.20.2
- [IMP] server postinstall : put nginx backups in a different dir than nginx config
- [FIX] waptserver: fix empty error message when trying to activate an existing licence

WAPT Agent

- [NEW] added new waptguihelpers : grid_dialog, filename_dialog, input_dialog, combo_dialog
- [FIX] waptdeploy multiple setupargs raise « Invalid variant operation »
- [FIX] missing root certificates when exporting system store certificates in lazarus app (GetSystemCABundlePath). Must trust CA + ROOT stores
- [FIX] setuphelpers: regression in maintaining backward compatibility for some const which are functions too (programfiles etc..)
- [FIX] be tolerant if uuid can not be regenerated (on linux, dmidecode can't be run as normal user in session-setup)
- [FIX] fix wget waptdeploy.exe waptagent.exe in wads and detect mismatch drivers config
- [FIX] waptagent regression : Revert « [UPD] waptservice : tasks don't notify server by default to avoid too frequent updates of database. »
- [FIX] wapt-get : try to fix get service password on unix.
- [NEW] splitting remove_appx() with new function remove_user_appx() to avoid unexpected behavior
- [NEW] Add restart-waptservice action in wapt-get.py

- [FIX] fix publisher and version in installed_softwares macos
- [FIX] use waptservice to check if is_enterprise in waptexit (avoid direct access to local waptdb) (fix unable to access sqlite db on linux / mac)

WAPT to GPLI connector

- [FIX] glpi fix install_date
- [FIX] regression in glpi export (Softwares)

62.2.10 WAPT-2.2.0.11720 (2022-03-15)

hash: 8e07f388

This is the first release of the 2.2 serie of WAPT.

WAPT Core

- [NEW] Discovery mode for the WAPT Console
  - when checking acls, the licencing status is taken in account to enable or not actions.
  - maximum number of 300 managed hosts in discovery mode.

WAPT Deployment Server (WADS)

- [NEW] tech preview Automated Windows OS deployment called WADS :
  - Using a winpe image (network boot or usb key boot).
  - Shipping wimboot, ipxe.efi, undionly.kpxe, 7z.dll.
  - Added openssl win64 binaries for WADS
  - Added **`wads.exe`** and **`wgetads`** custom binaries in distribution.
  - Added WADS repo option in repo rules.
  - Added a WAPT Console page to list raw registered hosts, upload winpe images, define default config, uplaod drivers bundles.
  - On WAPT Server: added `/var/www/wads/` add a non protected `/wads` in **`nginx`** config.

WAPT Console

- [NEW] add columns in private repo to display newest software version (Tranquil IT effort to parse softwares providers download sites) and newest package version (from Tranquil IT store database).
- [NEW] Dynamic Agent configuration using *`.json`* files stored on the WAPT Server:
  - Added a `last_update_config_fingerprint` local param to keep track of current config.
  - Added "configurations" (merged config overview) data when uploading host status to the WAPT Server.
- [NEW] Dynamic Agent configuration using config packages:

  - Added `templates/setup_package_template_conf.py.tmpl` package template.
  - Added a `wapt/conf.d` directory on the WAPT Agent to hold the installed *`.json`* configuration files.
- [NEW] New in the WAPT Console: added option to show the host WAPT Agent configurations overview.
- [NEW] New in the WAPT Console: option to display a graph of host packages dependencies.
- [NEW] New in the WAPT Console reporting: tabbed interface to displays multiple query results.
- [NEW] New in the WAPT Console: option to filter host inventory based on the result of a SQL query:
  - In reporting, right click on column which represent a host UUID and « choose as Host UUID » abnd save.
  - The query is then available in the combobos « Filter hosts on SQL query » in hosts inventory.
- [NEW] New in the WAPT Console: add a *Tech preview* Tab for packages development workflow:
  - Create from template;
  - Displays `waptdev` directory sources package status;
  - Basic git commands.
- [IMP] Improved the WAPT Console send message : enable use of HTML (copy & paste). HTML Preview.
- [IMP] Do not clear selection on mouse right-click when selecting package names in package edits.
- [IMP] refactored the WAPT Console code to remove most python calls:
  - removed `waptdevutils.py`, removed calls to WaptRemoteRepo, replaced by pure fpc code.
- [UPD] Updated the WAPT Console: merged selected hosts add/remove depends, add/remove conflicts in a single action/form
- [UPD] Updated the WAPT Console update package source: add a checkbox to enable package version increment.
- [UPD] Updated the WAPT Console "plugins" config: warn user if not saved.
- [UPD] Updated the WAPT Console: removed obsolete Add ADS Groups to selected host action.
- [UPD] Updated the WAPT Console action *Refresh Host Inventory* triggers a **`update_server_status`** instead of a full computer register.
- [UPD] Updated the WAPT Console: host additional tools (rdp, vnc, etc) which requires to look for a connected IP are now run in a thread to avoid freezing the UI.
- [UPD] Start of use of mormot2 for X509 and RSA crypto instead of python bindings in the WAPT Console
- [FIX] waptconsole : store executable signature with new key name format (xxx.exe keys)
- [FIX] duplicated panels in initial configuration package wizard.

WAPT Self-Service

- [IMP] waptself: add logger.

WAPT Server

- [IMP] Improved the WAPT Server authentication: try ldap authentication only if `ldap_auth_server` is defined.
- [UPD] Updated the WAPT Server licencing: use **`waptlicences.pyd`** instead of pure python code.
- [UPD] Updated the WAPT Server: add config options `wads_folder` and `agent_folder`.
- [UPD] Updated the WAPT Server: improve GLPI export, add "smodel" on GLPI exports and add "monitors".
- [IMP] force en_US.utf8 locale for linux services.
- [IMP] add /api/v3/latest_installed_package_version.
- [UPD] upgraded jquery to v3.6.0.

WAPT Service

- [NEW] Added `/opt/wapt/wapt-get.bin` to linux distributions.
- [NEW] New in the WAPT service: added a *WaptUnregisterComputer* task and **`unregister_computer`** socketio action.
- [IMP] Improved the WAPT service: improved logger.
- [IMP] Improved the WAPT service and the WAPT Agent take into account the licencing status:
  - Added a `licences` local params to store the current registered licences retrieved from the WAPT Server during the last update.
- [UPD] **`waptcrypto.py`**: made optional the joining of signer certificate when signing claims.
- [UPD] Updated the WAPT Deployment utility: increased timeout from 4s to 15s when pinging the current http WAPT service.
- [UPD] Upgraded **`dmidecode`** to v3.3 on windows.
- [UPD] Updated the WAPT service: do not check battery level for *WaptAuditPackage* task.
- [REF] Installers : merged `wapt.iss` and `common.iss`.
- [FIX] wapttasks: took in account non default config filename.
- [FIX] Fixed the WAPT service: reporting properly the user which created a task (either locally or using websockets).
- [FIX] Fixed the WAPT service: fixed icons in package local webpage.

wapt-get

- [IMP] wapt-get new config actions. Added actions:
  - **`add-config-from-file`**;
  - **`add-config-from-base64`**;
  - **`add-config-from-url`**;

  with parameters:
  - `--not-interactive`: Disables dialog to ask credential users (for batch mode);
  - `--waptbasedir`: Forces a different wapt-base-dir then default dir of `waptutils.py`;
  - `--devmode`: Enables devmode. dbpath is set to memory and certificate/key paths are in `userappdata`;

  - `--json-config-name`: The name of the *.json* file given with the action **json-config-from-file/base64/url**;
  - `--json-config-priority`: The priority of the json file given with the action json-config-from-file/base64/url.
- [UPD] Removed **update-packages** action synonym for **scan-packages**.
- [IMP] wapt-get added **update-status** action in service mode **wapt-get -S update-status**.
- [IMP] Enabled `--CAKeyFilename` and `--CACertFilename` wapt-get options
- [IMP] Added logger for waptguihelper pyd module. if `--loglevel` = `debug` in commandline, logger is activated.
- [IMP] Reporting the `use_repo_rules` flag to the WAPT Server in wapt_status
  - Report `is_enterprise` flag to the WAPT Server
  - Report installed antivirus and monitors in host inventory
- [IMP] Audit loop granularity based on actual installed packages:
  - Added **get_next_audit_datetime()** on Wapt class.
  - `waptaudit_task_period` attribute is now in the Wapt class instead of the WAPT service.
- [UPD] Removed the not functional `--dry-run` wapt-get option.
- [IMP] Improved **register** computer fallback from kerberos to password based authentication:
  - Do not send audit data when registering to limit workload.
- [IMP] Try registering computer if **update_server_status** fails because of authentication.
- [IMP] **waptpython.exe**, **waptpythonw.exe**, and **nssm.exe** are now signed with Tranquil code signing key.
- [NEW] added **pylint** and **black** modules. Added black configuration to **vscode** project template.
- [NEW] Added `setuphelpers.getscreens`.
- [IMP] Improved *SetupHelpers* unzip : new `extract_with_full_paths` argument (default True).
- [NEW] New *SetupHelpers* `listening_sockets()`.
- [IMP] Added `templates/setup_package_template_portable_exe.py.tmpl` and `templates/setup_package_template_portable_zip.py.tmpl` package templates.

Others stuff

- [IMP] Added `windows_version_prettyname` and `windows_version_releaseid` in `host_info`.
- [IMP] Always use **RunAsAdminWait** to copy package certificate to the local WAPT service `waptssl` directory.
- [IMP] Improved the WAPT Console config: stores WAPT Server certificate in `AppUser` folder (`roamingwaptconsolesslserver`).
- [IMP] Reset TLS client key password in the WAPT Console config if connection error.
- [UPD] Retire python `GetPrivateKeyPath`, raise exception if `GetPrivateKey` does not succeed.
- [FIX] Clear cached TLS client key password when validating the the WAPT Console config dialog.
- [IMP] Improve GLPIlpi settings windows.
- [IMP] Clean up the html error page from the WAPT Server when checking the WAPT Server and WAPT repository URL.

- [FIX] Don't reenter the private key password dialog if already asking the user. This issue can be triggered if several therad are using a key, or if cooperative multitasking like TAction messages (OnUpdate) triggers a Get with client side certificate authentication.
- [SEC] Fix `dhparam` on the WAPT Server postconf.
- [FIX] Fix failover on file version with **`remove_outdated_binaries()`**.
- [IMP] Add `asset_tag` to sysinfo api.
- [FIX] `Get_antivirus_info`: test if timestamp attribute exists.
- [IMP] New getscreens function.
- [IMP] Added columns *uuid manufacturer* and *product serialnumber* in database.
- [UPD] Added `mac_addresses` to `LocalSysinfo`.
- [UPD] Expanded LocalSysinfo with uuid, serial_number and sku_number, fixed keys with underscore.
- [IMP] Improved matching of reachable IPs of client using new GetReachableIP from mormot2.
- [UPD] GetReachableIP: connection tests are performed in parallel using mormot GetReachableAddr instead of one after the other to reduce delay when launching IP based command to remote hosts from the WAPT Console.
- [FIX] Take `--config option` in account for wapt-get fpc code.
- [UPD] waptcrypto: implemented `TX509Certificate.CN`, removed `TX509Certificate.DN`.
- [UPD] Updated *SetupHelpers* **`need_install`**: now comparing software versions with 4 members. Assumes that 1.2 == 1.2.0.0 and 1.2.3.4.5 == 1.2.3.4, **`remove_previous_version`**: use version with 4 members.

62.3 WAPT-2.1 Serie

62.3.1 WAPT-2.1.2.10652 (2022-01-10)

hash: 7dd63b61

- [UPD] shorten the default package filename. If `target_os` is alnum, do not include md5sum in the filename. If `target_os` is in tags, do not duplicate it in filename
- [FIX] disable debug data for linux
- [FIX] try to circumvent issue with Trend antivirus blocking the **`WaptTaskManager`**. Looks like the issue is with platform.win32_ver using win32api.GetVersionEx…
- [FIX] Installed softwares invalid conditions
- [FIX] fix local_user and local_group on macOS
- [FIX] removed workaround on 60s delay for websocket disconnect
- [FIX] use CompressGZip instead of CompressZLib on the WAPT Server, compression is GZip
- [FIX] Allow "~" in package filenames
- [FIX] try to not update records in database if data has not changed
- [FIX] Wake on lan relay now equals is remote repository, close #2940
- [FIX] fix group members

- [FIX] return only local and user group (ignore nsswitch)
- [FIX] backported the WAPT Exit utility (improved detailed logging) from 2.2
- [FIX] backport waptlicences py module from 2.2
- [SEC] check that hostname matches https certificate in the WAPT Console http client.
- [FIX] backport uwaptlicencing: allow empty json licencing data
- [FIX] fix WaptHttpPostData
- [FIX] check valid uri in wapthttputils waptwget WaptWget_Try
- [FIX] init LastModifiedDate to "" if not found in THttpResponse
- [FIX] add a 50ms report delay for httpprogressnotification
- isolate wapt python engine: PyFlags:= [pfNoUserSiteDirectory, pfIsolatedFlag];
- [FIX] Fixed *SetupHelpers*: backported changes from 2.2 is_linux64 type_rhel fix installed_softwares for type_redhat upd uninstall_apt with autoremove
- [FIX] `user_appdata` = `user_local_appdata` for unix
- [IMP] introduced get_powershell_str, get_default_app remove_appx
- [IMP] introduce InitLogger for the WAPT Exit utility
- [FIX] Fixed the WAPT Console: generalize the use of a fallback package_uuid in case of old packages without package_uuid field.
- [FIX] Fixed the WAPT Console: use editable dropdown in frmpackagedetails for maturity
- [FIX] backport issue with inc version of some group packages when importing
- [FIX] Disable client side ssl authentication on root WAPT Server url (regression)
- [FIX] isolate from user python env when building binary packages
- [UPD] improved feedback message for license activation on the WAPT Server.
- [UPD] wapt-scanpackages.py: add option -d to disable update of database Packages table.
- [FIX] The -b switch is True by defaut, so there were no way to disable update of database table.
- [UPD] Updated the WAPT Console: be tolerant for old package without package_uuid
- [UPD] strip ending slash in {{data.wapt.hostname}} server template properties to avoid double slashes in templates result
- [UPD] backport openssl build parameter from 2.2
- [FIX] Fixed the WAPT Agent url link in the WAPT Server index page.
- [FIX] setproctitle only for unix
- [FIX] locate packages in host packages grid using package_uuid instead of id, so that refreshing grid works properly with a multiselection of hosts.
- [UPG][SEC] upgrade python version from 3.8.11 to 3.8.12
- [FIX] remove python3 dependencie. Now python3 is included in wapt

62.3.2 WAPT-2.1.2.10605 (2021-11-30)

hash: e2a0e2a0

- [FIX] Fixed the WAPT Console: backport edit multiple hosts add/remove depends/conflicts (issue « no password available yet » when kerberos enabled) backport IpExecute from 2.2
- [FIX] unable to edit stripped down package with integrated package editor. (setup.py file hash issue) update package size
- [FIX] bad path for nginx dhparam for Windows server
- [FIX] upgrade mormot2
- [FIX] waptself local admin NOPASSWORD setting did not work anymore log authentication user when task is triggered from local wapt webservice don ot raise exception in check_auth_groups but return (None, None) instead to avoid Error 500 in browser backport fix for integer attributes in packages index backport fix for loading ssl libraries
- [FIX] Update wake on lan with broadcasts
- [FIX] Error « Add: Unexpected [%] object property in an array » for old package with empty package uuid
- [FIX] Acl handle boolean as global ACL
- [FIX][SEC] issue with acls : action is enabled when acl is set to json false

62.3.3 WAPT-2.1.2.10588-rc1 (2021-11-22)

hash: e70d9039

- [FIX] fix installed_softwares for older debian and improve inventory performance
- [FIX] fix glpi inventory failure (exception on int conversion)
- [SEC] [FIX] invalid condition on package hash check
- [SEC] [FIX] cleanup nginx config templates
- [NEW] add uwsgi support for Debian server
- [FIX] add user information in audit
- [FIX] Improve lazarus ini parser to support other values than “1”/”0” as boolean values (True, true, 1, 01, etc. same behavior as python iniparse)
- [IMP] support for message previsualisation and templates in waptmessage editor and better multiline support
- [UPD] waptsetup : do not use kerberos by default
- [NEW] show certificate when double click in acl tab
- [IMP] Do not propose to start the WAPT Console after install (due to different user context)

62.3.4 WAPT-2.1.1.10568 (2021-11-08)

hash: 978c00ae

This is a bugfix version with some small improvements. The main fix is for websocket issue.

- [IMP] Prevent multiple websockets connections from same host uuid on the WAPT Server (bugged wapt clients can maintain multiple websockets, which leads to a lack of avalable connections on the WAPT Server)
- [FIX] Fixed restart of the WAPT service with exit code 10 (managed by the nssm service manager)
- [FIX] Fixed case on the WAPT service where different threads access simultaneously to a shared Wapt instance
- [IMP] Introduced some randomness when the WAPT service reconnects its websocket.
- [IMP] Checking more cases to determine if token for websocket has to be updated.
- [IMP] Introducted a wait in the socket client until it is actually disconnected before trying to reconnect to avoid multiple websocket threads from same client.
- [IMP] Do not re-create a new SocketIOClient at each reconnection, but reuse existing one to minimize risk of multiple connections.
- [FIX] Do not consider "%" char as unsafe in filenames
- [IMP] Improved logging of the WAPT service (logger wapttasks report main actions triggered by the service in `waptlogwaptservice.log`). Removed "flask.app" logger config.
- [IMP] Remove the WAPT packages's persistent directory on the WAPT client when a WAPT package is forgotten
- [IMP] Added `ignore_empty_names` argument to *SetupHelpers*.installed_softwares
- [IMP] Improved display of `package_uuid` with command wapt-get list
- [IMP] Added *redhat_based* tag for WAPT package operating system tags
- [FIX] Fixed `decrypt_fernet` / `fernet_encrypt` functions
- [IMP] Improved the reporting of key as name in softwares inventory for softwares without a descriptive name
- [FIX] The `server_uuid` column in hosts database updates properly.
- [FIX] Fixed the removal of packages when `only_if_not_process_running` = `True`.

Known issues:

- When the websocket is reconnecting, if the IP adrress has changed, the main IP adrress is not updated in IP adrress column in the WAPT Console.

62.3.5 WAPT-2.1.0.10550 (2021-10-08)

hash: 953c9552

This is a bugfix version with some small improvements.

- [FIX] Fixed mass add / remove on multiple host at once.
- [FIX] Fixed issue when editing a package without a « description_en » attribute in control file.
- [FIX] Fixed drag drop when editing *selfservice* package.
- [IMP] Improved feedback when uploading WAPT packages.
- [IMP] Improved handling of the list of wakeonlan relay.

- [IMP] Improved remote repository is now by default a wakeonlan relay.
- [FIX] Fixed access violation error when viewing certificate list.
- [FIX] Fixed do not enable verbose logging by default on the WAPT Console, the WAPT Exit utility and waptselfservice (might fill up %APPDATA% ...).
- [FIX] Fixed use `templates/wapt-logo.png` in the WAPT Exit utility if it exists.
- [IMP] Improved login error message.

62.3.6 WAPT-2.1.0.10517 (2021-09-30)

hash: fa2af298

This is the first release of the 2.1 branch. It is mainly a incremental improvement with many small but worthy fixes on the 2.0 branch.

The WAPT service

- [IMP] During upgrade, **`wapt-get session_setup`** is not run if no userspace configuration is defined for the installed WAPT packages.

The WAPT Deployment utility

- [IMP] Improved automatic proxy detection and configuration possible with the new `--http_proxy` = `True` / `False` parameter or explicit url command line parameter.
- [IMP] Disabled https verification when downloading **`waptagent.exe`** if a fingerprint is provided (allows installation with on out-of-date computer with expired certificate store).
- [IMP] Do nothing if no –waptsetupurl argument is provided (it reduces the probability of false positive on antivirus check).
- [IMP] Double check WAPT installed version after install and report error message if it does not match (allow detection of installation that have been blocked by a misconfigured antivirus for example).

The WAPT Console

- [NEW] tech preview: new tab to provide basic package editing functionnality directly in the WAPT Console without having to open **`Pyscripter`** or **`VSCode`**.
- [NEW] New tech preview: new tab to browse the developement directory directly from the WAPT console.
- [NEW] Single Sign On with Kerberos authentication (if `service_auth_type` = `waptserver-ldap` and `use_kerberos` = `True`).
- [NEW] New button to display WAPT packages that have a specific WAPT package as a dependency in the private repository tab.
- [NEW] New message box to decrypt message sent by the WAPT Agents (using `encrypted_data_str` / `print_encrypted_data` in waptcrypto). This allows an admin to upload sensitive information from desktop that will be asymetrically signed by the Administrator's public key.
- [NEW] New set of icons and many small visual improvments.
- [NEW] New software inventory tab to display installed software (not packages) and see which hosts have that specific software.
- [NEW] New button to delete Windows Update KB files that are not used anymore by any computers. This allows to keep the Windows Update storage volume under control.
- [NEW] New tab to have a user-friendly display of the certificates that are deployed on a specific host.
- [NEW] New tab to display the certificates that are available on a WAPT repository.

- [NEW] New warning icons on the hosts tab when the computer needs a restart (after a windows update for example).
- [NEW] New filter by OS option.
- [NEW] New icons in the OU tree view if a OU package exists for that Organizational Unit.
- [NEW] New information message about the choice of maturity when creating new WAPT Agent and by default uploading in DEV maturity (to avoid being directly deployed to all client computers, this allow to test the new WAP Agent on a subset of computer before full scale deployment).
- [IMP] Made GLPI export configuration more intuitive.
- [IMP] Improved the WAPT Console plugin versatility. All inventory attribute can now be used in command lines (it use the « mustache » template syntax, eg. {{ main_ip }} {{ computer_fqdn }} {{ host_capabilities.os_version }} « {{#host_capabilities.tags}}{{.}},{{/host_capabilities.tags}} » etc.
- [IMP] Allow non standard port in the WAPT Console configuration.

waptself

- [NEW] allow custom logo in waptselfservice
- [NEW] Single Sign On using Kerberos (`needs service_auth_type` = `waptserver-ldap` and `use_kerberos` = `True`)
- [IMP] allow customisation of package details view using template engine

WAPT Exit utility

- [IMP] allow custom logo (on Windows, Linux and macOS)

wapt-get

- [NEW] better handling of licence information. Now the licence is uploaded on the WAPT Server and it is not necessary to install it on every admin WAPT Console computer
- [IMP] propagate ExitCode from Python calls for better error handling
- [IMP] better handling of websocket reconnection (check of socket status every 120s)
- [IMP] periodic check of the UUID and the current certificate of the WAPT Agent for consistency between the WAPT Agent and the client computer
- [NEW] waptsetup et waptserversetup new parameters: `set_verify_cert` and `set_kerberos`

62.4 WAPT-2.0 Serie

62.4.1 WAPT-2.0.0.9470 (2021-10-07)

hash: 5065cb57

This is a security release with a few related bugfixes. All Wapt 2.0 version below 2.0.0.9467 are affected.

- [SEC] fix for vuln in urllib3 CVE-2021-33503 (CVSS Score: 7.5 High, CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:N/I:N/A:H).
- [SEC] Sanitize filename used when downloading files on local client. (CVSS Score : 7.5 High, CVSS;3.1/AV:L/AC:H/PR:H/UI:N/S:C/C:H/I:H/A:H/E:U/RL:O/RC:C). Enforced on wget and local filenames for downloaded packages (chars "\" ".." @ | ( ) : / , [ ] < > * ? ; ` n are removed or replaced).
- [SEC] Do not use PackageEntry filename attribute to build target package filename as it is not signed.

- [UPD] **`wapt-get remove`**: reraise exception if there is exception in uninstall script return traceback in "errors" key return code 3 if there are errors when removing packages in **`wapt-get remove`**.
- [FIX] handles wildcards in certificates in the WAPT Console config and create waptsetup update UI in external repositories config when setting CA bundle.
- [FIX] use PackageEntry.localpath only for local status of a package.
- [UPD] split PackageEntry non_control_attributes into *repo_attributes* and *local_attributes*. *local_attributes* are not put into Packages index as they are not relevant for remote access.
- [UPD] update python modules requirements following urllib3 upgrade idna==3.2 (from 2.10) certifi==2021.5.30 (from 2020.12.5) requests==2.26.0 (from 2.25) urllib3==1.26.6 (from 1.26.5)

62.4.2 WAPT-2.0.0.9450 (2021-08-10)

hash: 7bc6920c

This is a security fix version affected by CVE-2021-38608.

Please visit the *security bulletin* to learn more.

62.4.3 WAPT-2.0.0.9449 (2021-06-22)

hash: 70283a14

This is a bugfix version with some small improvements.

WAPT Agent

- [FIX] Fixed Windows Update fix in the progress bar.
- [IMP] Allow the WAPT Agent to upgrade even when on batteries.

The WAPT Server

- [IMP] Many fixes in GLPI sync.
- [FIX] Better handling of service_delete exception cases.
- [FIX] Fixed database migration handling with `create_defaults_users` procedure.
- [FIX] Fixed on windows skip the WAPT Agent build if there is no available certificate for signing.

The WAPT Core

- [IMP] Improved the compatibility of `Packages` file for easing upgrade from WAPT 1.8.2.
- [IMP] Improved the WAPT Deployment utility: behavior to avoid wrong red flag from AV softwares.

Caveat

For macOS support one should use the WAPT Agent 2.1 version available in nightly channel.

62.4.4 WAPT-2.0.0.9428 (2021-05-06)

hash: 4b33cf96

This is a bugfix version with many small improvements.

WAPT Console:

- [IMP] Improve *CreateWaptSetup* form layout.
- [IMP] Restore focused column visibility when refreshing grid data.
- [FIX] Fix wrong path for wapt-get.py in vscode project.
- [UPD] Update No fallback in rules to true by default.
- [FIX] `enable-check-certificate` with wildcard.
- [FIX] take into account the `use_http_proxy_for_repo` ini setting (if not present, assume `False`).
- [FIX] Fix `setup_package_template_msu.py.tmpl` for package Wizard.
- [IMP] Add new template for creating package with certificate.
- [IMP] Add option to check downloaded package with VirusTotal in package import GUI.
- [IMP] Add update-package source action directly in Private repository in the WAPT Console.

WAPT Agent:

- [IMP] Use task queue for the forced installs instead of running them inline.
- [FIX] Database not opened when we check Hosts who are secondary repositories.
- [IMP] Restart partial download of Windows Update files.
- [IMP] Improved icons handling in **`WaptSelfService`**.
- [IMP] On macOS use host certificate store by default for https certificate validation.
- [IMP] `reload_config_if_updated` now reload config if `public_certs_dir` has changed.
- [FIX] WUA: better handling of return code « does not apply to this computer ».

WAPT Server:

- [FIX] Fixed bad migration of PGSQL databse server side.
- [FIX] Improved database upgrade in corner cases.

SetupHelpers

- [FIX] Fixed `register_windows_uninstall` calculation and using correct x86_64 environment with **`register_uninstall`** and **`unregister_uninstall`**.
- [IMP] Improved inline function description for documentation.

62.4.5 WAPT-2.0.0.9343 (2021-04-08)

hash: 117d62b8

This is mainly a bugfix release after the initial 2.0.0 release.

WAPT Console:

- [IMP] Show an explicit message if the user can not build a customized WAPT Agent.
- [IMP] Enabled remote repo sync if there are repo configured (making `remove_repo_support` parameter obsolete).
- [IMP] Better filtering on `maturities`.
- [FIX] Fixed templates for vscode

WAPT Server:

- [IMP] Include certificates from WaptUsers table in result of /api/v3/known_signers_certificates.

WAPT ACL handling:

- [UPD] ACL: added an action to show the user certificate.
- [UPD] Creates default (empty) WaptUserAcls record on user login even for non ldap logins.
- [IMP] Better naming for ACL domains.

SetupHelpers

- [FIX] Fixed `register_uninstall`.
- [FIX] Do not change silently `maturity` and `locale` in `check_package_attributes`.
- [FIX] Fixed regression in wget resume.

Other technical stuff:

- [IMP] Added support for installation on OracleLinux.
- [FIX] Tightened files ACLs on Linux + fixes + SELinux fixes in postconf.
- [IMP] Introduced **`mORMot2`** framework in Lazarus code.
- [FIX] Fixed datetime conversion in the WAPT Console.

62.4.6 WAPT-2.0.0.9300 (2021-03-30)

hash: 018b8b57

This is the first release of the 2.0 series. After one year in development and more than 1600 commits it brings a bunch of new features and enhancement to the last major update of WAPT 1.8.2. On the technical side WAPT 2.0 now embed Python3 and now support 8 new platforms (some of them backported to 1.8.2).

The switch to Python3 may require minor adjustment to the existing package that may have been development in-house (refer to the corresponding doc page). The packages offered by Tranquil IT through the WAPT Store are already compatible with WAPT 2.0.

From a sysadmin point of view

- [NEW] ACLs.
- [IMP] WAPT Server side ACLs in addition to certificate validation.
- [IMP] User management interface with certificate listing.
- WAPT Console:
- [IMP] gui: change maturity directly from the WAPT Console.
- [IMP] gui: all WAPT package types are grouped in one tab.
- [IMP] helpers: build and upload locally development package from the WAPT Console.
- [IMP] helpers: import default reporting queries from internet.
- [IMP] helpers: restart the WAPT Agent and restart client computer from the WAPT Console.
- [IMP] Package wizard: support for RPM/DEB/PKG/DMG.
- [IMP] Remote repositories: status bar for progression of creation/ update of `sync.json` for repo sync.
- [IMP] Windows Updates: new search bar, view host with specific KB.
- [IMP] Faster import and resigning of package, change of maturity, etc.
- [IMP] **`waptmessage`**: better handling of user oriented notification.
- [IMP] Better logging of WAPT Console actions and WAPT Agent activity.
- Performance improvements for larger installations:
- [IMP] Better handling of insert / update of inventory.
- [IMP] Better handling of websocket updates.
- [IMP] GLPI integration: synchronize WAPT inventory to GLPI server.
- Better OS integration:
- [IMP] TLS certificate handling: **`certifi`** uses local OS certificate store instead of Python **`certifi`** integrated certificate store.
- [IMP] Increased the number of supported platform, improved packaging for Linux (deb and rpm) with support for a WAPT Agent running on arm64 and macOS BigSur 64bit.
- Package development:
- [IMP] Improved package wizard.
- [IMP] Many small fixes and improvements to *SetupHelpers* and better support for Linux and macOS.
- [IMP] Improve os targeting now you can specify targeted OS and specific version of OS : eg. Debian(>=9,<=10).

From a technical point of view

- Python: switch from Python2.7 to Python3:
- Linux: use of venv by default with distrib python 3 version.
- Windows: switch python3 install to embedded edition 3.8.7.
- Different installer for WinXP / WinVista / Win2k3r2 / win2k8 (nonr2) (recent CPython version does not support older Windows systems anymore).
- Better handling of passwords with special chars.
- Upgraded WAPT core libs and scripting environment.
- Upgraded to Python3 and Python libraries, changed kerberos and websocket libraries.
- Upgraded to Lazarus 3.0.10 and FPC 3.2.

Caveat

- Support for non supported Windows version (WinXP, WinVista, Win2k8 (non-R2) and Win2k3) is still baking in the oven and should be ready shortly after the 2.0 release date.
- Redhat8 and derivative distributions: for upgrade it is necessary to remove WAPT SELinux rules before using postconf again.

CHAPTER 63

Accord de licence d'utilisateur final WAPT

NOTICE: READ THE FOLLOWING TERMS AND CONDITIONS CAREFULLY BEFORE YOU DOWNLOAD, INSTALL OR USE Tranquil IT's PROPRIETARY SOFTWARE. BY INSTALLING OR USING THE SOFTWARE, YOU AGREE TO BE BOUND BY THE FOLLOWING TERMS AND CONDITIONS. IF YOU DO NOT AGREE TO THE FOLLOWING TERMS AND CONDITIONS, DO NOT INSTALL OR USE THE SOFTWARE.

63.1 Definitions

« You and your » means the party licensing the software hereunder.

« Software » means the computer programs provided under the terms of this license by Tranquil IT together with any documentation provided therewith.

« WAPT Server » means the system running the WAPT server software.

« Managed computer » means a computer running the WAPT service agent software.

63.2 Grant of rights

63.2.1 General

The license granted for software under this agreement authorizes you on a non-exclusive basis to use the software. The license is personal to you and may not be assigned by you to any third party.

63.2.2 License Provisions

Subject to the receipt by Tranquil IT of the applicable license fees, you have the right use the software as follows:

> You may use and install the WAPT client software for the duration of the license on as many « managed computers » as the license agrees. Nothing in this agreement shall permit you, or any third party to disclose or otherwise make available to any third party the licensed software, source code or any portion thereof. You agree to indemnify, hold harmless and defend Tranquil IT from and against any claims or lawsuits, including attorney's fees, that arise as a result from the use of the software; You do not permit further redistribution of the software by your end-user customers

63.3 No derivative works

The inclusion of source code with the License is explicitly not for your use to customize a solution or re-use in your own projects or products. The benefit of including the source code is for purposes of security auditing. You may modify the code only for emergency bug fixes that impact security or performance and only for use within your enterprise. You may not create or distribute derivative works based on the software or any part thereof. If you need enhancements to the software features, you should suggest them to Tranquil IT for version improvements.

63.4 Ownership

You acknowledge that all copies of the software in any form are the sole property of Tranquil IT. You have no right, title or interest to any such software or copies thereof except as provided in this Agreement.

63.5 Confidentiality

You hereby acknowledge and agreed that the software constitute and contain valuable proprietary products and trade secrets of Tranquil IT, embodying substantial creative efforts and confidential information, ideas, and expressions. You agree to treat, and take precautions to ensure that your employees and other third parties treat, the software as confidential in accordance with the confidentiality requirements herein.

63.6 Disclaimer of warranties

EXCEPT AS OTHERWISE SET FORTH IN THIS AGREEMENT THE SOFTWARE IS PROVIDED TO YOU « AS IS », AND Tranquil IT MAKES NO EXPRESS OR IMPLIED WARRANTIES WITH RESPECT TO ITS FUNCTIONALITY, CONDITION, PERFORMANCE, OPERABILITY OR USE. WITHOUT LIMITING THE FOREGOING, Tranquil IT DISCLAIMS ALL IMPLIED WARRANTIES INCLUDING, WITHOUT LIMITATION, ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR FREEDOM FROM INFRINGEMENT. SOME JURISDICTIONS DO NOT ALLOW THE EXCLUSION OF IMPLIED WARRANTIES, SO THE ABOVE EXCLUSIONS MAY NOT APPLY TO YOU. THE LIMITED WARRANTY HEREIN GIVES YOU SPECIFIC LEGAL RIGHTS, AND YOU MAY ALSO HAVE OTHER RIGHTS THAT VARY FROM ONE JURISDICTION TO ANOTHER.

63.7 Limitation of liability

You ACKNOWLEDGE AND AGREE THAT THE CONSIDERATION WHICH Tranquil IT IS CHARGING HEREUNDER DOES NOT INCLUDE ANY CONSIDERATION FOR ASSUMPTION BY Tranquil IT OF THE RISK OF YOU CONSEQUENTIAL OR INCIDENTAL DAMAGES WHICH MAY ARISE IN CONNECTION WITH YOUR USE OF THE SOFTWARE. ACCORDINGLY, YOU AGREE THAT Tranquil IT SHALL NOT BE RESPONSIBLE TO YOU OR ANY THIRD PARTY FOR ANY LOSS-OF-PROFIT, LOST SAVINGS, INDIRECT, INCIDENTAL, SPECIAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF A LICENSING OR USE OF THE SOFTWARE.

63.8 Indemnification

You agree to defend, indemnify and hold Tranquil IT and its employees, agents, representatives and assigns harmless from and against any claims, proceedings, damages, injuries, liabilities, costs, attorney's fees relating to or arising out of your use of the software or any breach of this Agreement.

63.9 Termination

Your license is effective for a defined period and is terminated when this period is over. You may terminate it at any time by destroying the software or returning all copies of the software to Tranquil IT. Your license will terminate immediately without notice if you breach any of the terms and conditions of this Agreement, including non or incomplete payment of the license fee. Upon termination of this Agreement for any reason: you will uninstall all copies of the software; you will immediately cease and desist all use of the software; and will destroy all copies of the software in your possession.

63.10 Updates and support

Tranquil IT has the right, but no obligation, to periodically update the software, at its complete discretion, without the consent or obligation to you or any licensee or user.

YOU HEREBY ACKNOWLEDGE THAT YOU HAVE READ THIS AGREEMENT, UNDERSTAND IT AND AGREE TO BE BOUND BY ITS TERMS AND CONDITIONS.

CHAPTER 64

Licences des composants externes utilisés dans WAPT

Le développement du logiciel WAPT a commencé en mars 2012 ; il est porté en très grande partie par l'équipe de Tranquil IT.

Avec WAPT >= 1.9, les développements réalisés dans le cadre de WAPT sont soumis à une *licence propriétaire*.

Table1: Licences des composants externes utilisés dans WAPT

Composant WAPT	Licence
Python	Python Software License
Librairies Python	Licenses OpenSource diverses
Lazarus	GNU Public Licence
Composants Lazarus	GNU Lesser General Public License
Librairies Lazarus	Licenses OpenSource diverses
OpenSSL	Openssl License
Redistr. Microsoft Visual C++	Microsoft Software License Terms
PostgreSQL	PostgreSQL License
NSSM	Public Domain
Nginx	2-clause BSD-like license
mORMot2	Licenses OpenSource diverses