WAPT Documentation

Version 2.2

Tranquil-IT Systems

janv. 09, 2024

Présentation de WAPT

Bienvenue sur la documentation officielle de WAPT par Tranquil IT dernière version en date du 2024-01-09.

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.

CHAPITRE 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.

FIG. 1 – 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.

CHAPITRE 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 :

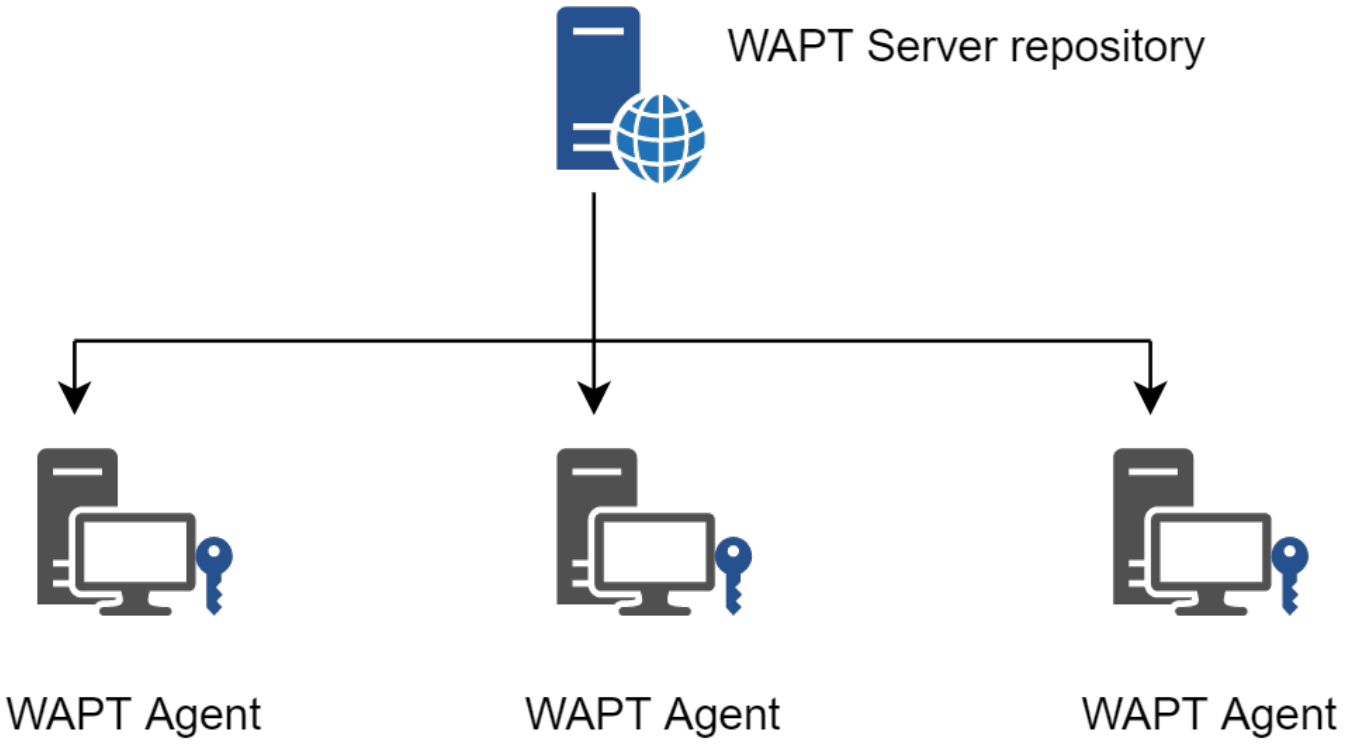


FIG. 1 – Réplication et dépots multiples

— 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*.

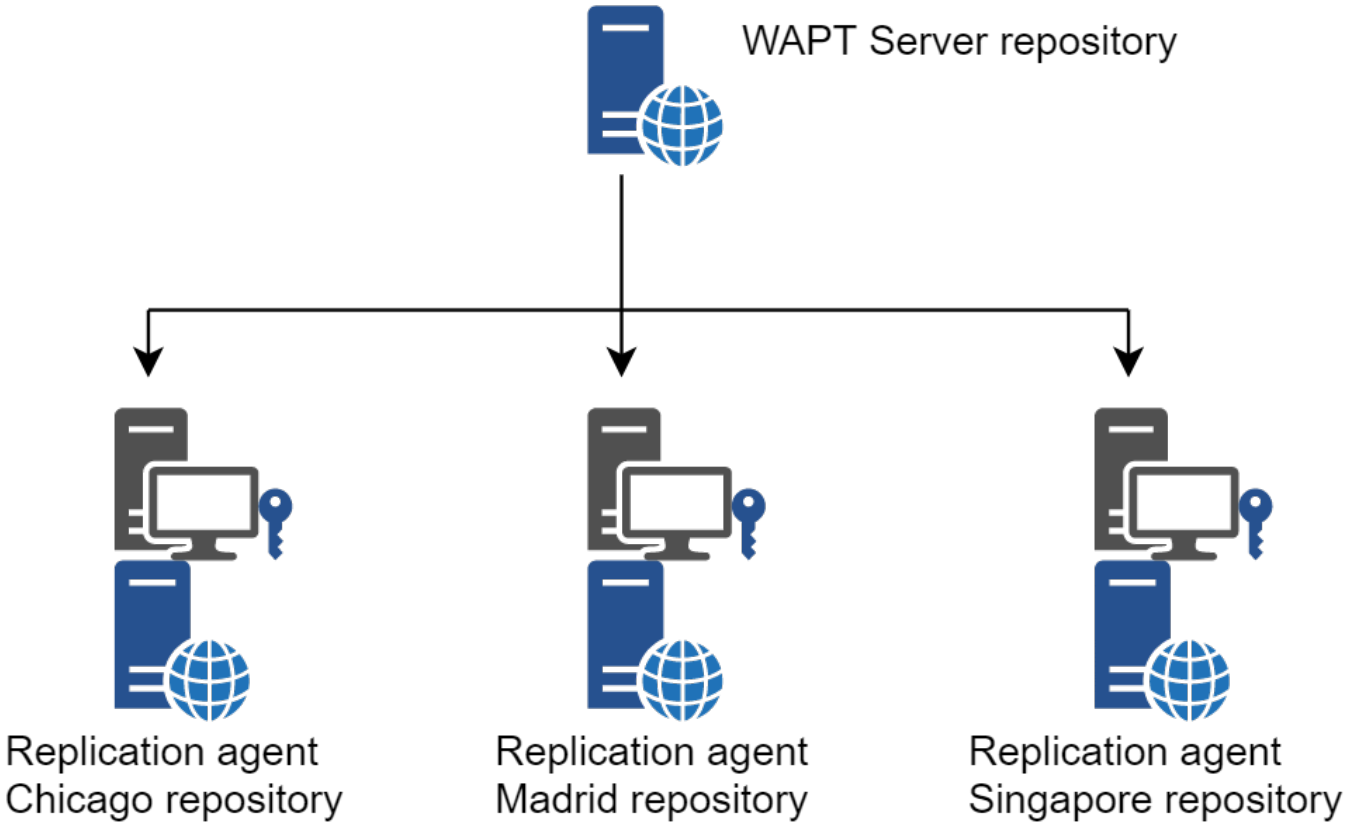


FIG. 2 – Réplication des dépôts WAPT

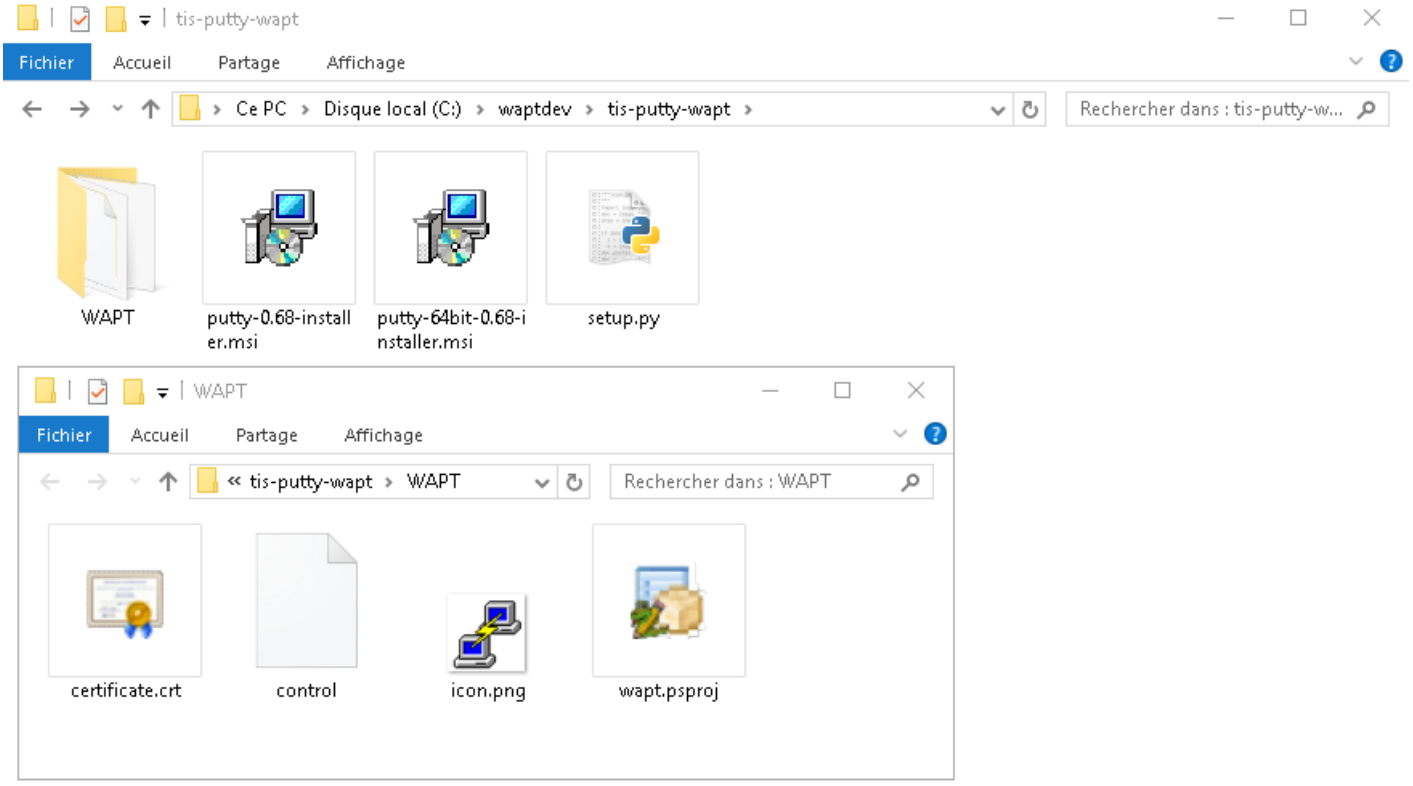


FIG. 3 – Structure du packaging WAPT affichée dans l'explorateur Windows

2.3.1 Types de paquets WAPT

Il existe 7 types de paquets WAPT :

FIG. 4 – 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 fonctionne dans l'*Organisation* :

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 indicateur devient **OK** et vert dans la console WAPT, ce qui signifie que l'hôte a le profil d'hôte que le *Administrator* ou *Package Deployer* a défini pour lui.

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.

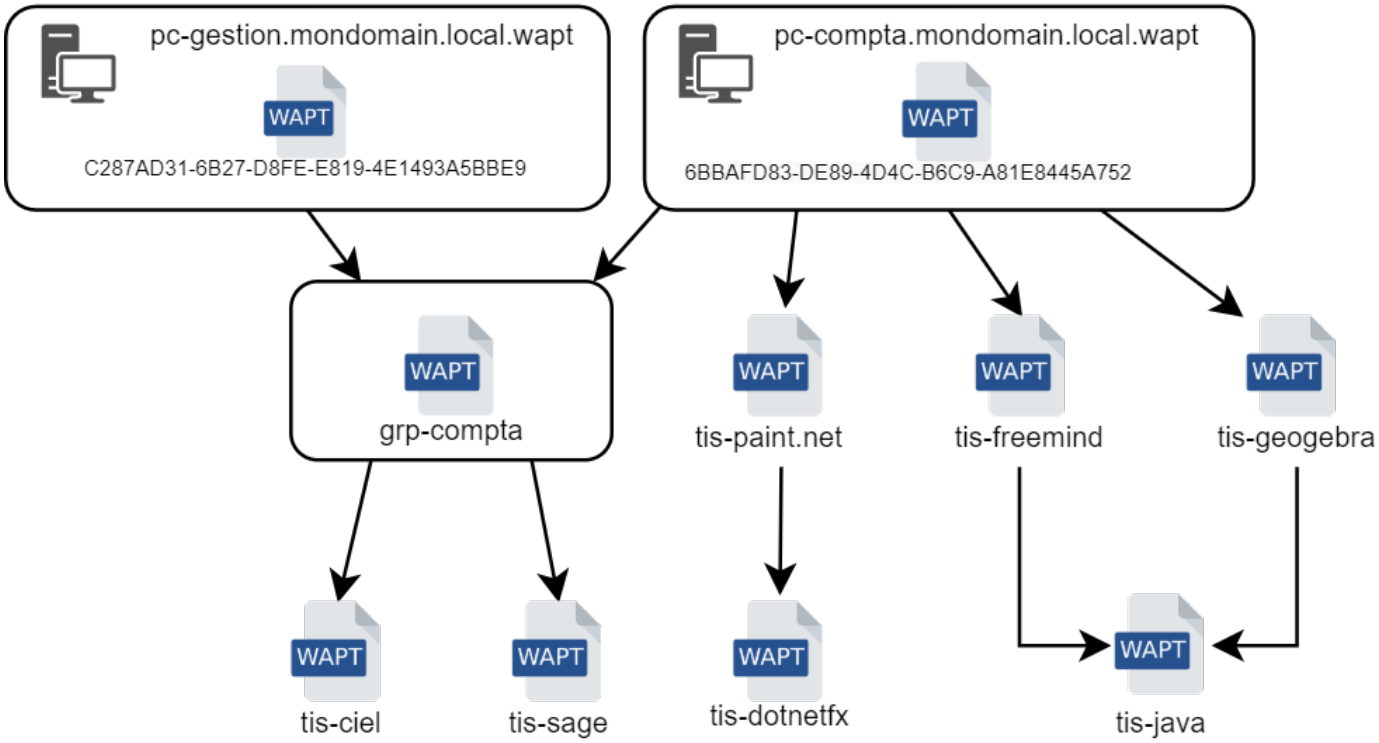


Fig. 5 – Schéma conceptuel du mécanisme de dépendance

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 hsah 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.

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.

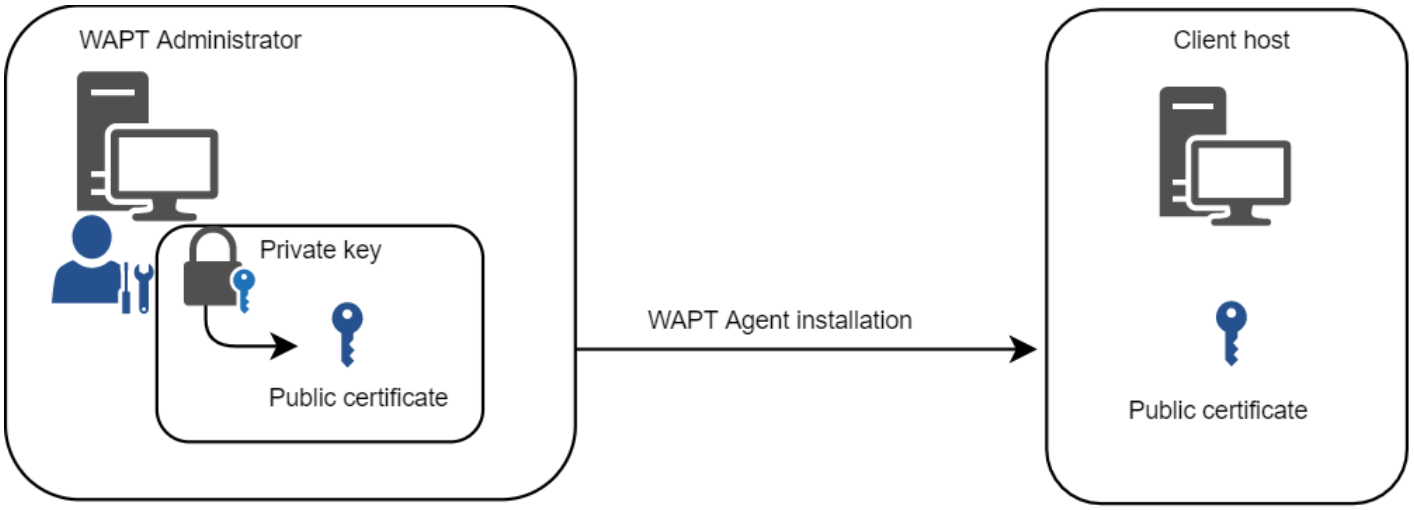


Fig. 6 – Clé privée / certificat public

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 A 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).

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 ».

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

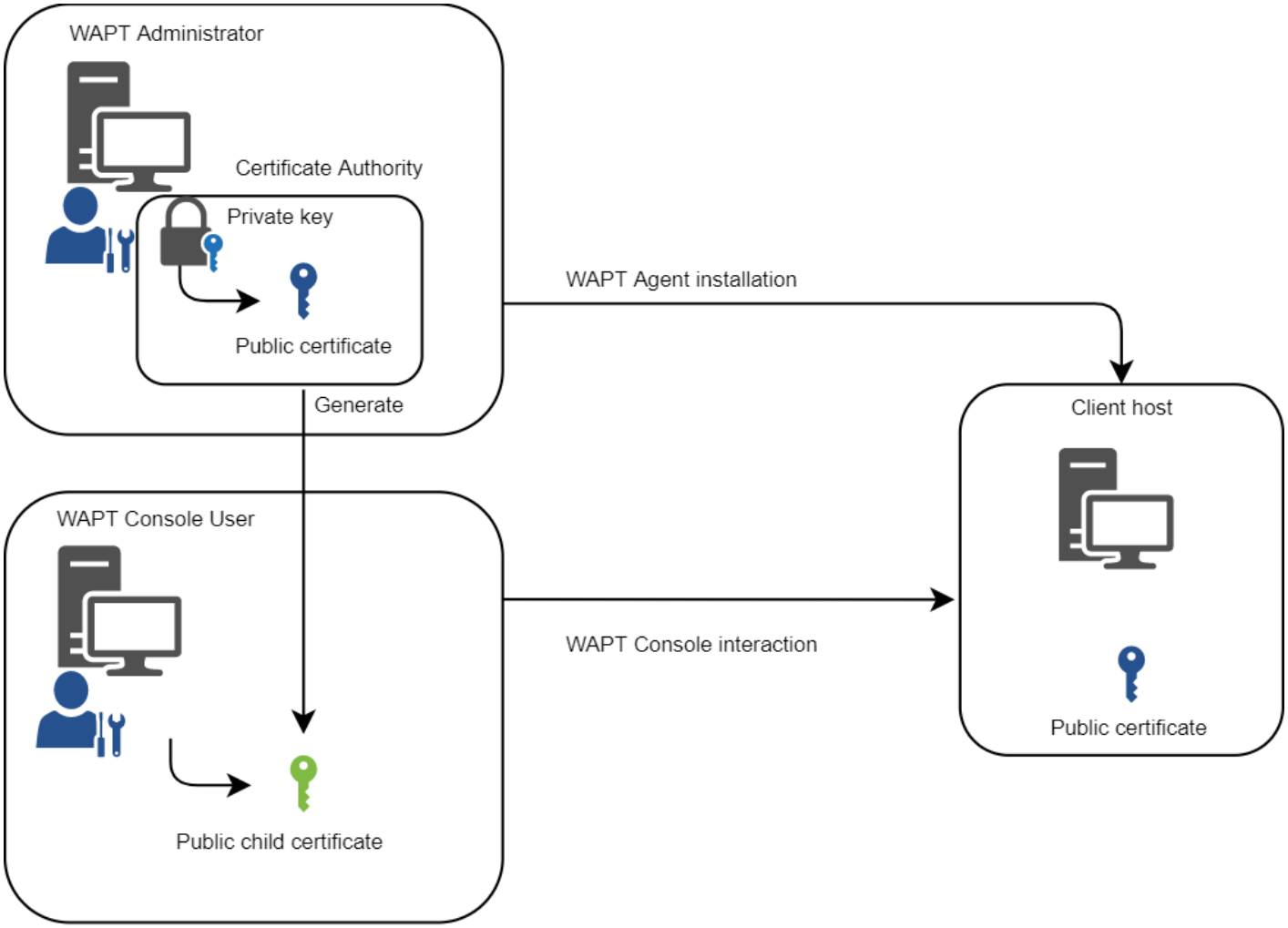


FIG. 7 – Différenciation des rôles des utilisateurs WAPT

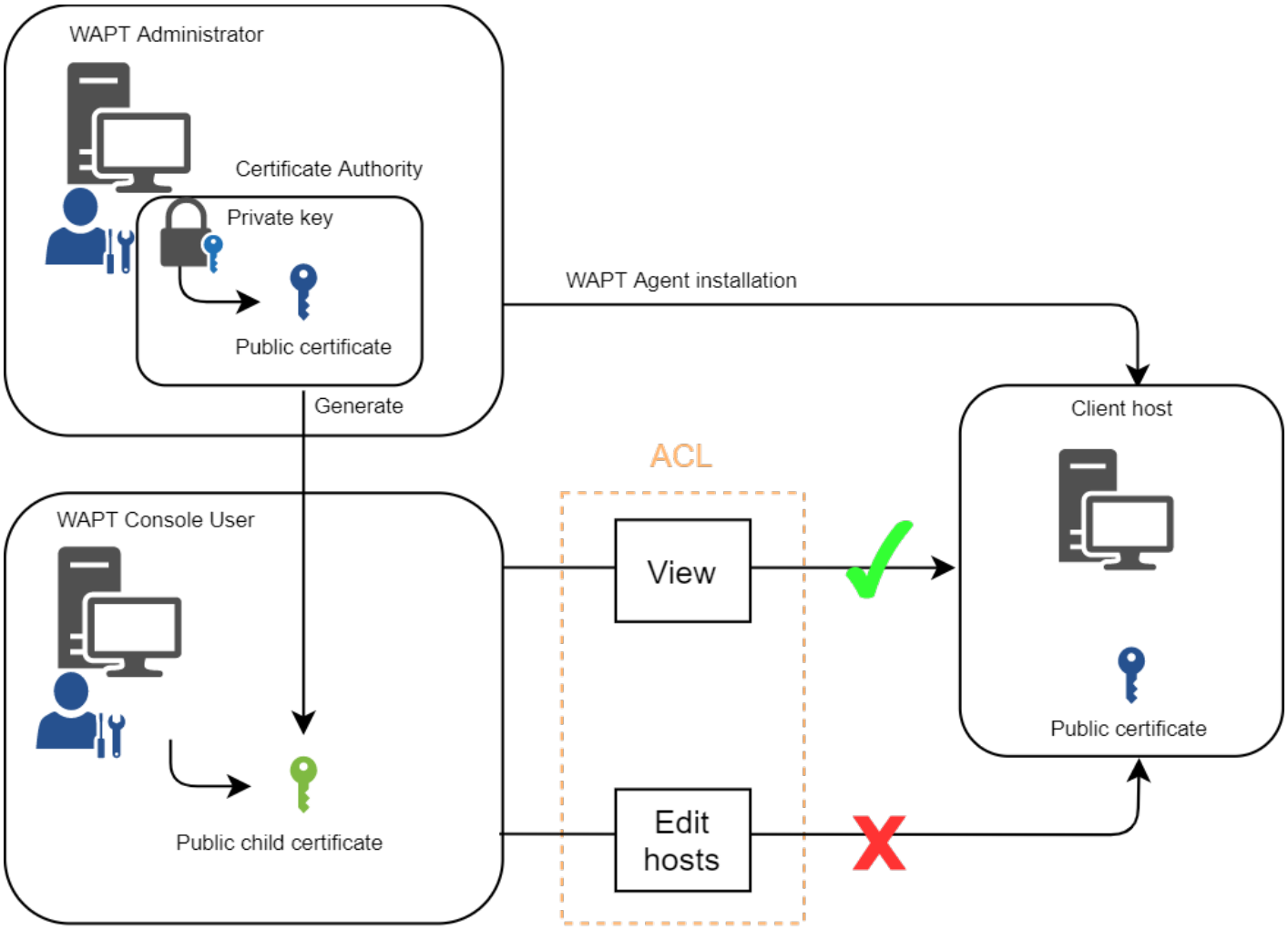


FIG. 8 – Différenciation des rôles ACL

CHAPITRE 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.

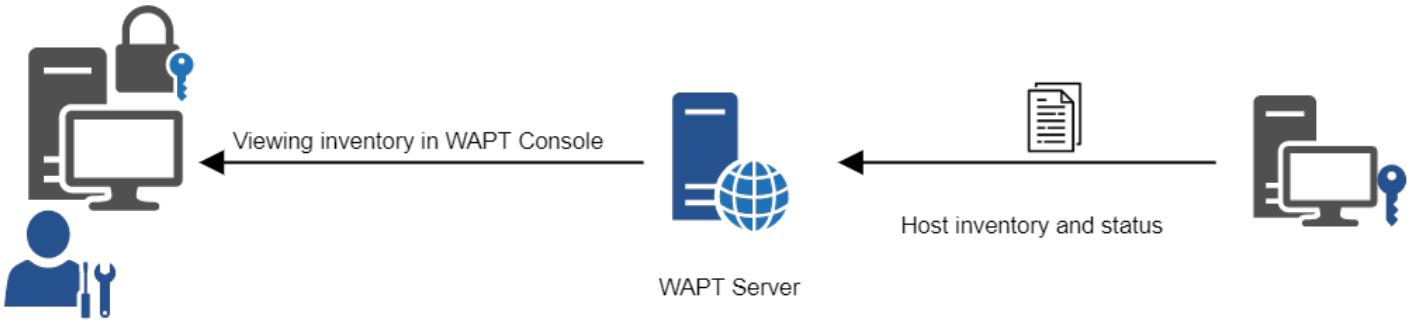


FIG. 1 – Fonctionnement de la remontée d'inventaire

— 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.

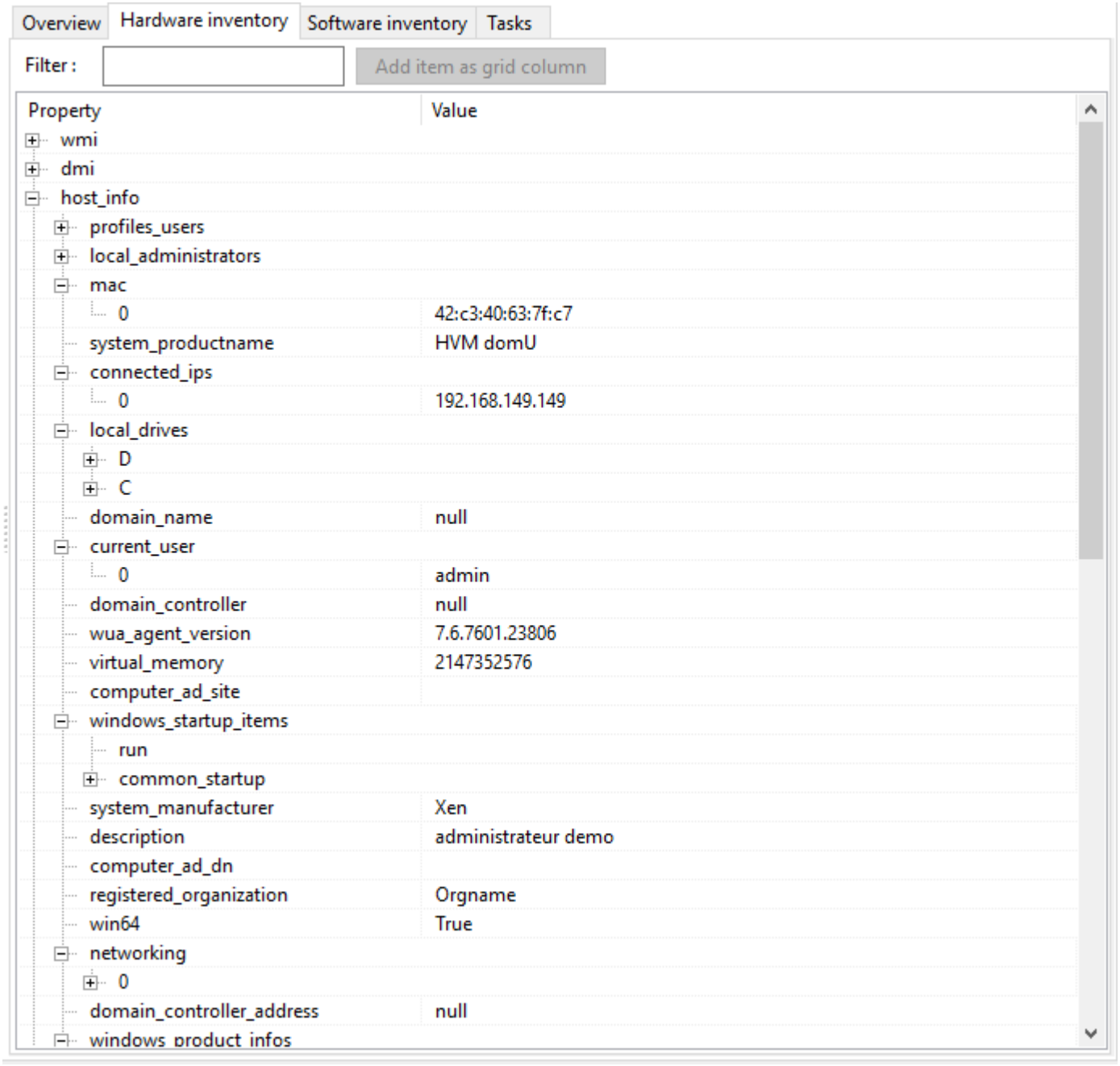


FIG. 2 – L'inventaire dans la console WAPT

3.2 La remontée des informations d'inventaire

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

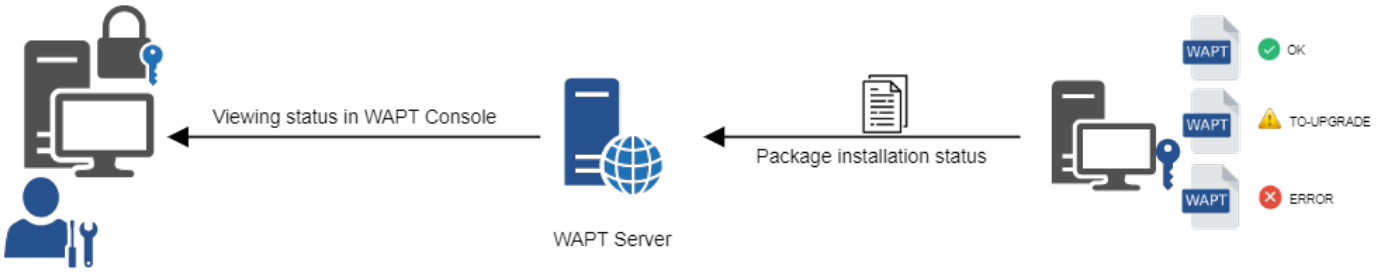


Fig. 3 – La remontée du statut des paquets vers le serveur WAPT

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 :

Tableau 1 – 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.

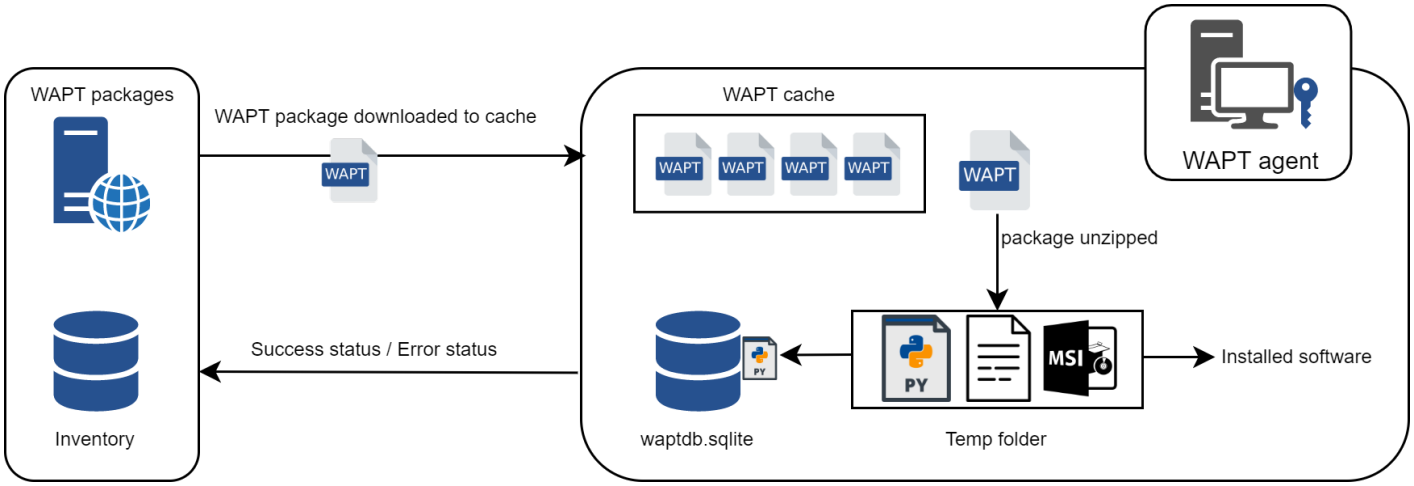


FIG. 4 – Diagramme de flux montrant le processus d'installation d'un packaging WAPT

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`**.

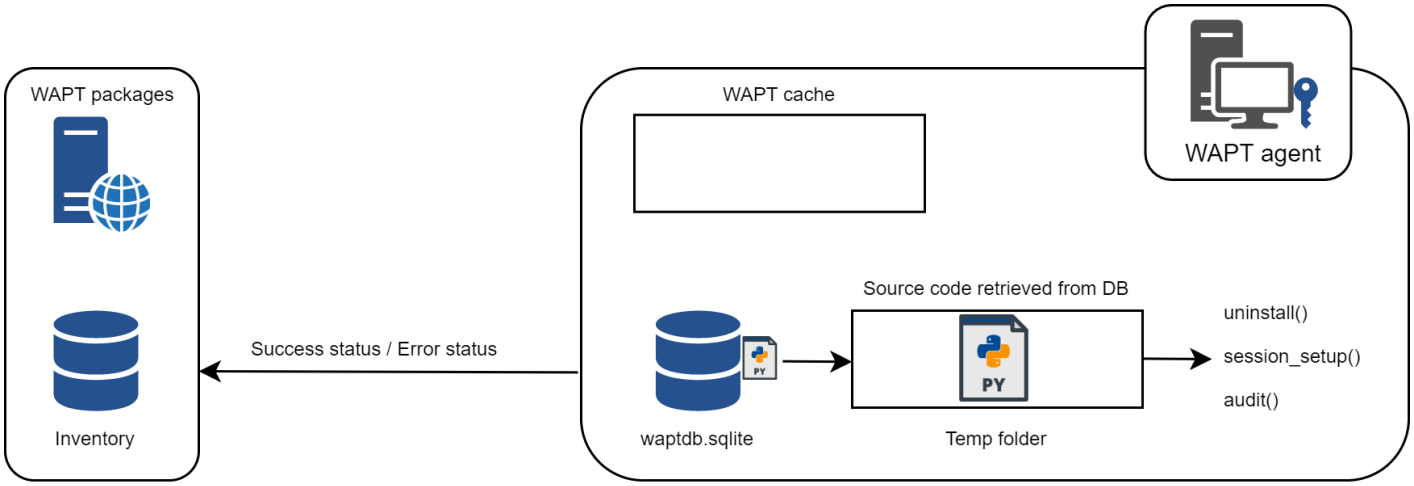


FIG. 5 – Comportement de l'agent WAPT avec la désinstallation, le session_setup et l'audit

3.5 Diagramme complet du fonctionnement de WAPT

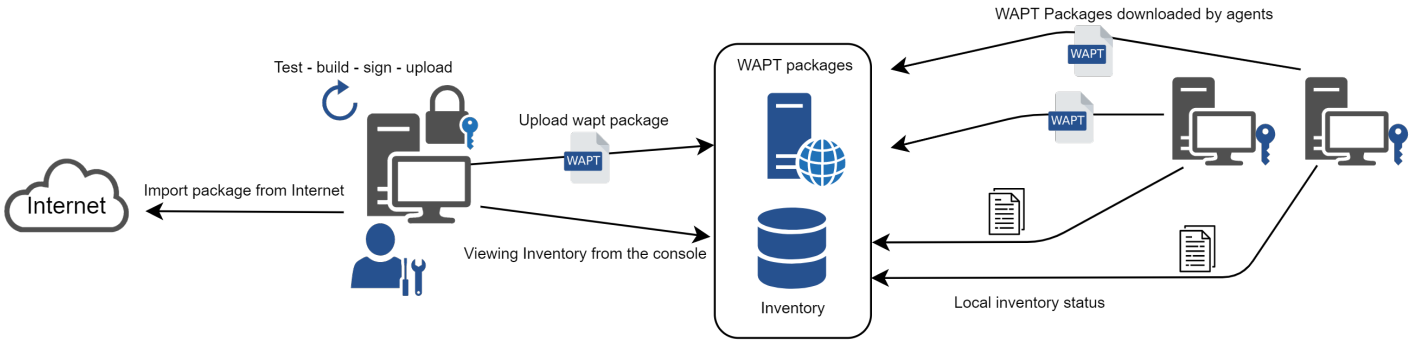


FIG. 6 – Diagramme de flux montrant le mode de fonctionnement général avec WAPT

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.
- 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.

CHAPITRE 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.

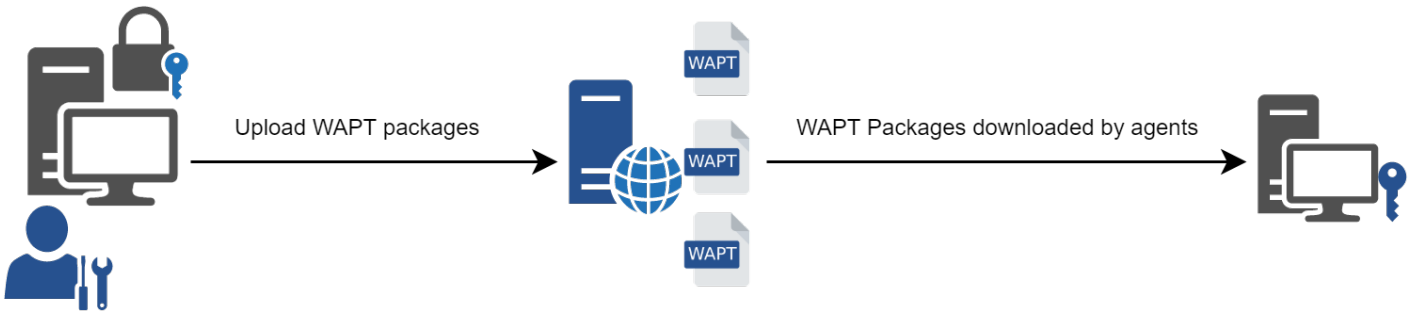


FIG. 1 – Schéma conceptuel du mécanisme de dépendance

- 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.

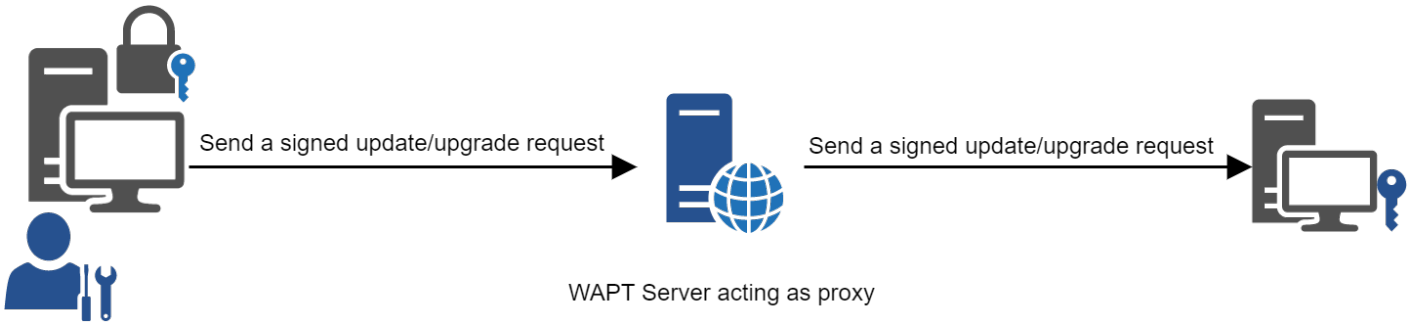


Fig. 2 – Schéma conceptuel du mécanisme de dépendance

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*.

CHAPITRE 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**.

CHAPITRE 6

Historique des éditions et des versions de WAPT

Tableau 1 – 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	Maintien de la sécurité uniquement	Maintien de la sécurité uniquement	Maintien de la sécurité uniquement	End Of Life		

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 2024-01-09

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.

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.

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.

Tableau 2 – Comparaison des caractéristiques entre les versions WAPT en date du 2024-01-09

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	✓	✓
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 (Public Key Infrastructure) pour le périmètre autorisé	✓	✗
Intégration avec Mesh Central pour un simple *partage d'écran* pour le support utilisateur	✓	✗
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	✓	✓

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.

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.

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

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.

CHAPITRE 7

Prérequis d'installation

7.1 Prérequis d'installation

7.1.1 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`.

7.1.2 Préconisations matérielles

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

Tableau 1 – 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).

7.1.3 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**.

Ouverture de ports

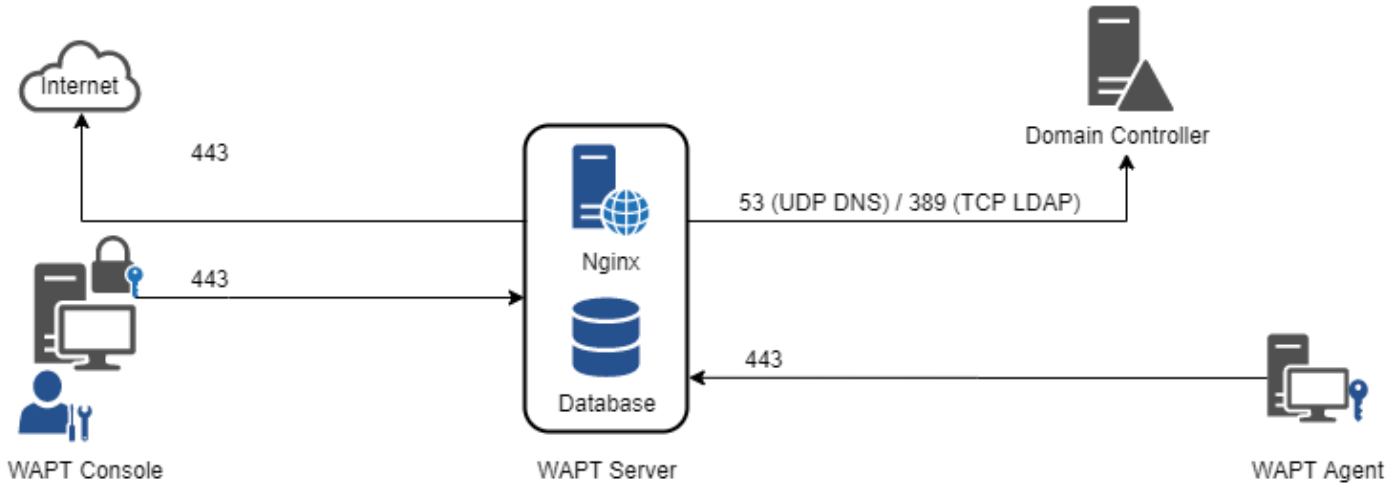


FIG. 1 – Diagramme des flux de données de WAPT

Comme vous pouvez le constater, seuls les ports **80** et **443** doivent être ouverts pour les connexions entrantes car les frameworks WAPT fonctionnent avec des websockets initiés par les agents WAPT.

Entrant

TABLEAU 2 – 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

Tableau 3 – 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.

7.2 Conseils avant l'installation

7.2.1 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 :

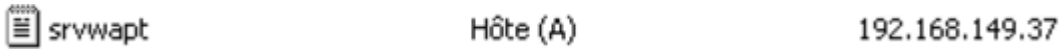
— *srvwapt.mydomain.lan*.
— *wapt.mydomain.lan*.

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.

7.2.2 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*.

CHAPITRE 8

Installer le serveur WAPT sur Debian

8.1 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*.

8.1.1 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* ;

8.1.2 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*.

8.1.3 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
```

— Vérifiez si la machine est correctement synchronisée avec le serveur NTP. Si elle n'est pas synchronisée, veuillez vous référer à la documentation du système d'exploitation pour configurer **`timedatectl`**

```
timedatectl status
```

— Mettez à jour et à niveau votre système d'exploitation et assurez-vous 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.

8.2 Installer les paquets du serveur WAPT

— Récupération de la clé `.gpg` et ajout du dépôt Tranquil'iT.

```
apt install apt-transport-https lsb-release gnupg -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-2.2/ $(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
```

8.3 Post-configuration

CHAPITRE 9

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 >
```

— 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 >
```

— 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*

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

CHAPITRE 10

Installation du serveur WAPT sur une distribution basée sur RedHat

10.1 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.

10.1.1 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* ;

10.1.2 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`.

10.1.3 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
```

— Vérifiez si la machine est correctement synchronisée avec le serveur NTP. Si elle n'est pas synchronisée, veuillez vous 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 est différente pour la mise à jour du serveur WAPT. Pour une mise à jour, rendez-vous sur *la documentation pour mettre à jour le serveur WAPT*.

10.2 Installer des paquets complémentaires

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-2.2/
enabled=1
gpgcheck=1
EOF
```

— Récupération de la clé `.gpg`.

```
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/postgresql14-setup initdb
sudo systemctl enable postgresql-14 waptserver nginx
sudo systemctl start postgresql-14 nginx
```

Redhat 8 et dérivés

```
cat > /etc/yum.repos.d/wapt.repo <<EOF
[wapt]
name=WAPT Server Repo
baseurl=https://wapt.tranquil.it/redhat8/wapt-2.2/
enabled=1
gpgcheck=1
EOF
```

— Récupération de la clé `.gpg`.

```
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:1.18 -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
```

10.3 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)

******************
```

(suite sur la page suivante)

(suite de la page précédente)

```
                < 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 >
```

— 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 >
```

— 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*.

CHAPITRE 11

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.

— Téléchargez et exécutez `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**.

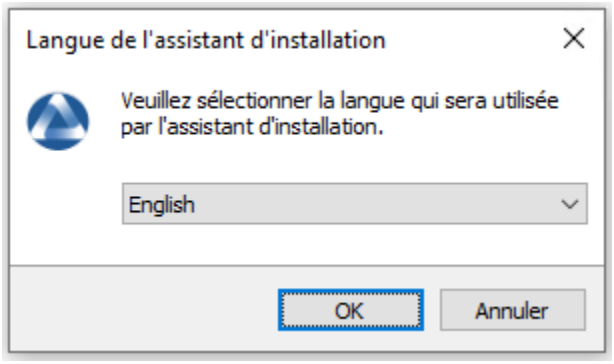
Avertissement : Le script de post-configuration réécrit la configuration de nginx. Si vous utilisez une configuration spéciale, sauvegardez votre fichier `nginx.conf` avec la commande :

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

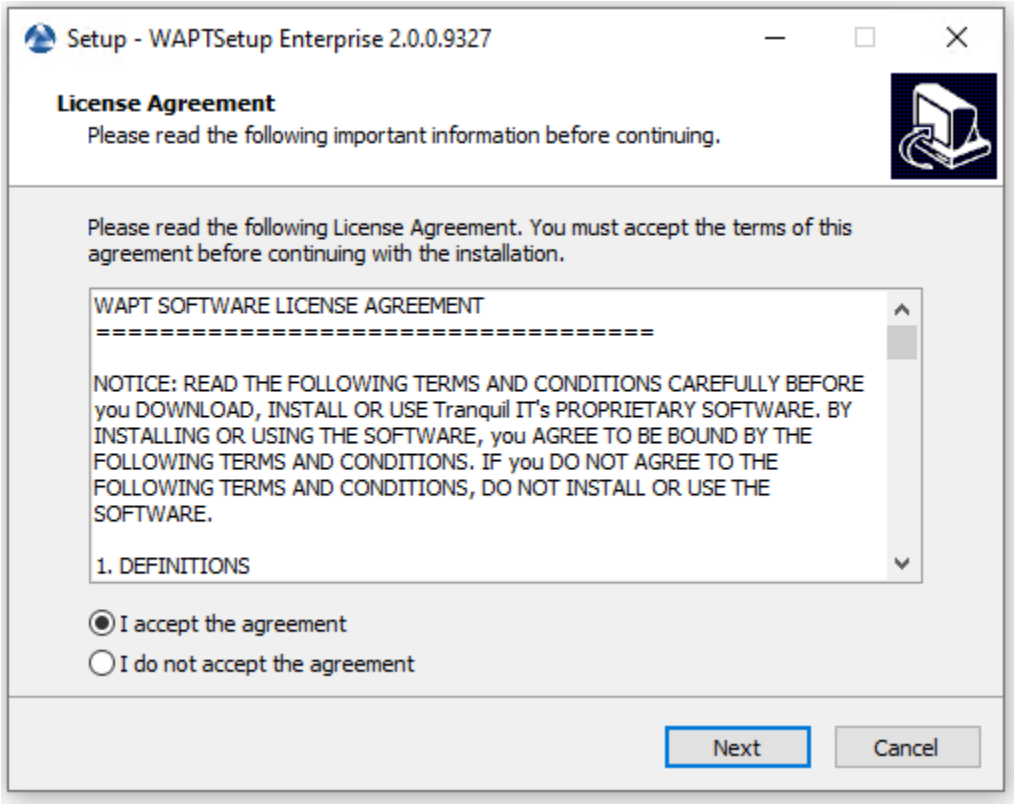
Il sera nécessaire d'écraser la configuration après le post-configuraiton avec la commande :

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

— Choisir la langue pour 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.

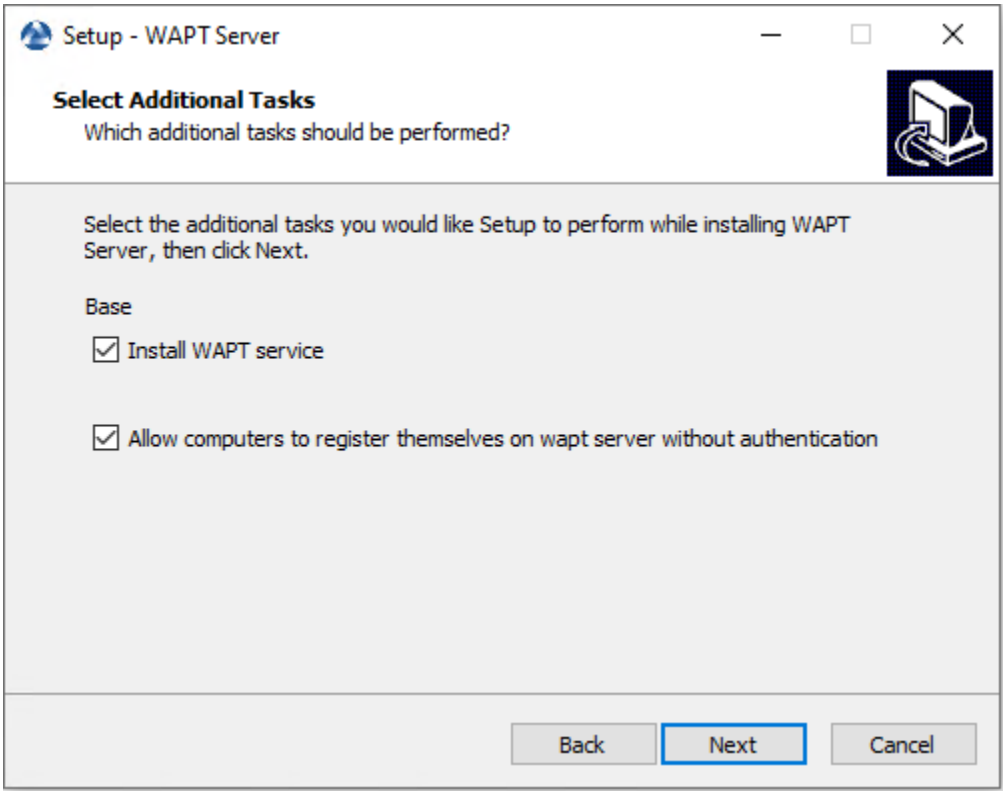
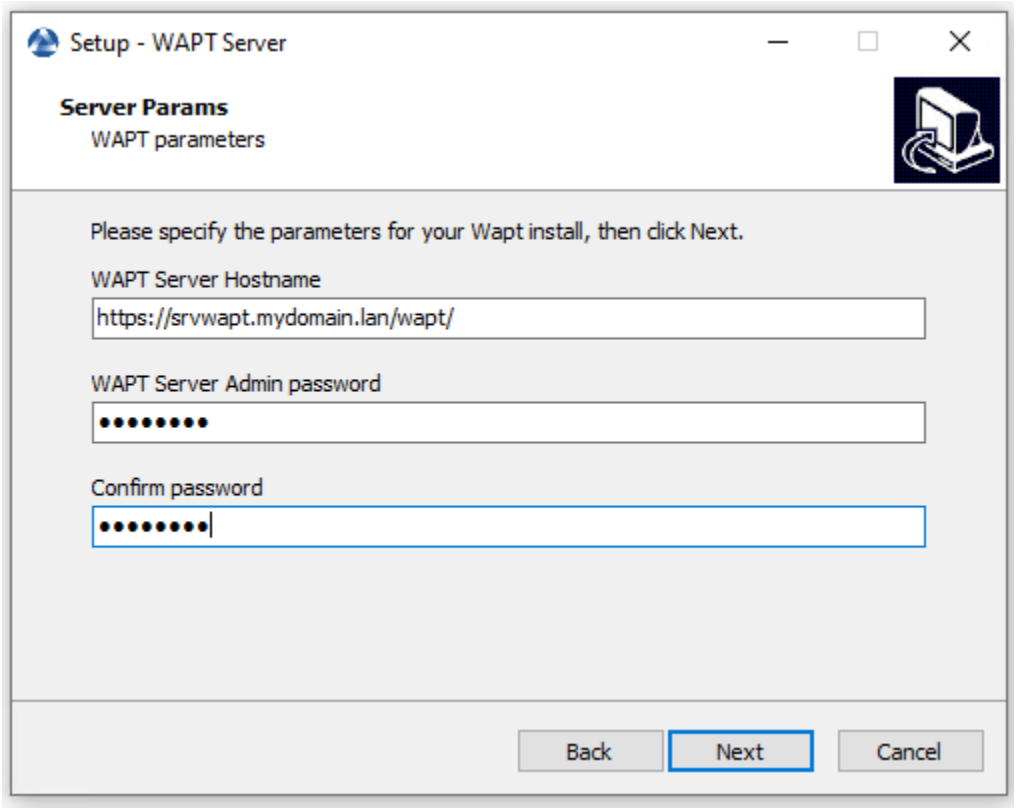
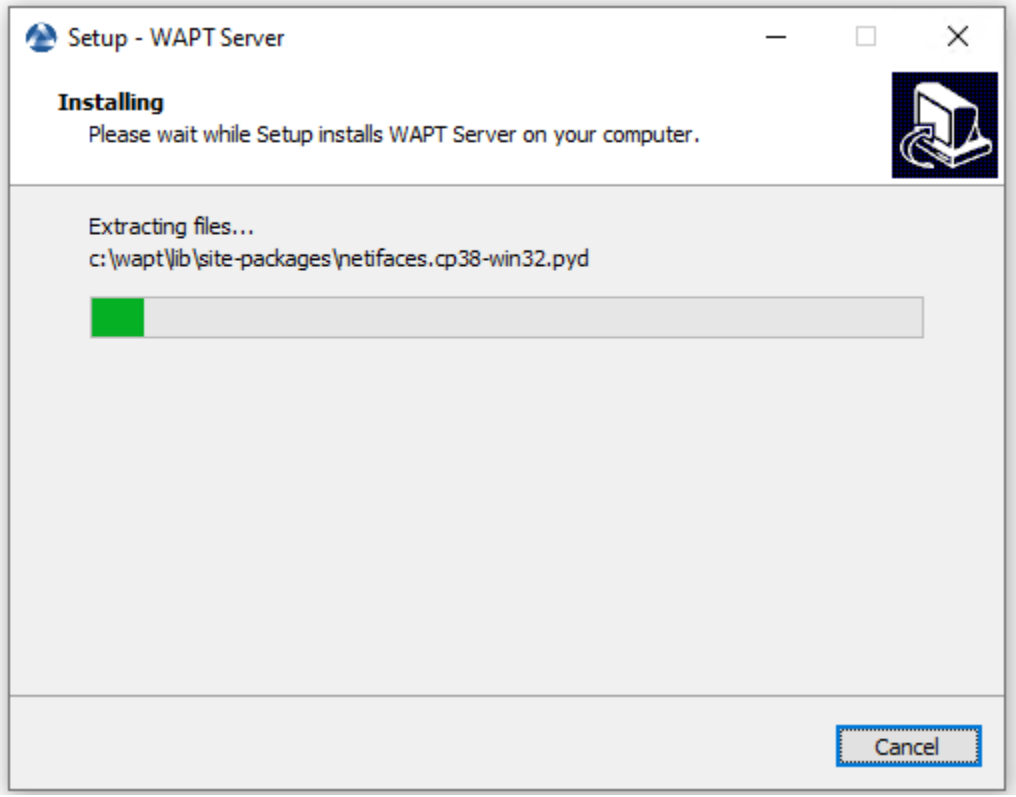


FIG. 1 – Choix des options du programme d'installation pour le déploiement du serveur WAPT



— 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.

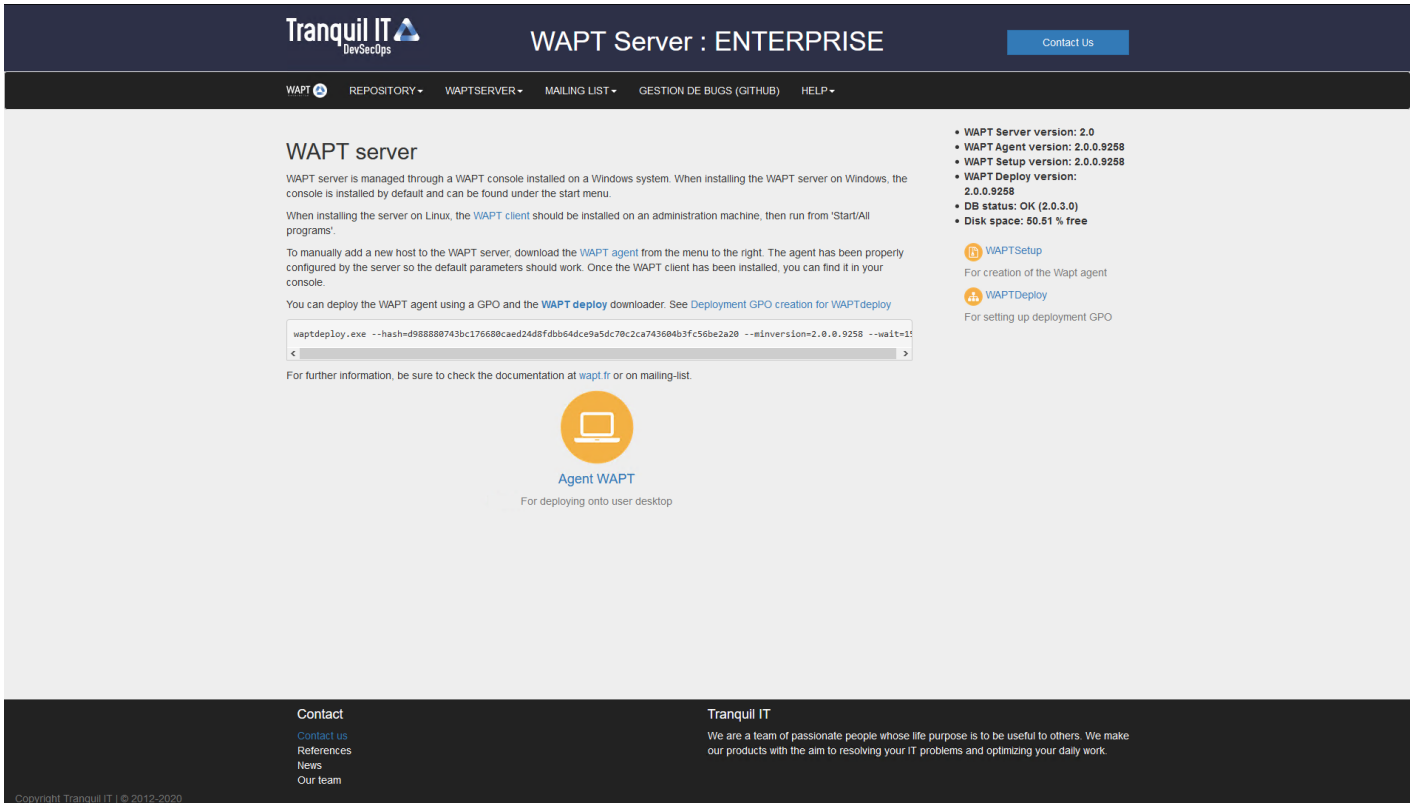


FIG. 2 – L'interface du serveur WAPT dans un navigateur web

CHAPITRE 12

Console de gestion WAPT

Attention : 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*.

Note :

- 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.
- A la date du 2024-01-09 la console WAPT ne s'installe que sous Windows.

Indication : Il est **hautement recommandé** d'utiliser la console sur une **machine de gestion dédiée**.

Serveur WAPT hébergé sur Windows

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 installer la console WAPT, téléchargez `waptsetup.exe` sur le serveur de Tranquil IT vers le serveur WAPT.

- Renommez le fichier *waptsetup-tis.exe*.
- Copiez-le dans `C:\wapt\waptserver\repository\wapt`.

Vous pouvez maintenant poursuivre *le téléchargement et le lancement de l'installation de la console WAPT sur l'ordinateur de l'Administrateur*

Serveur WAPT hébergé sur Linux

Passez à l'étape suivante, la Console WAPT est déjà sur votre serveur.

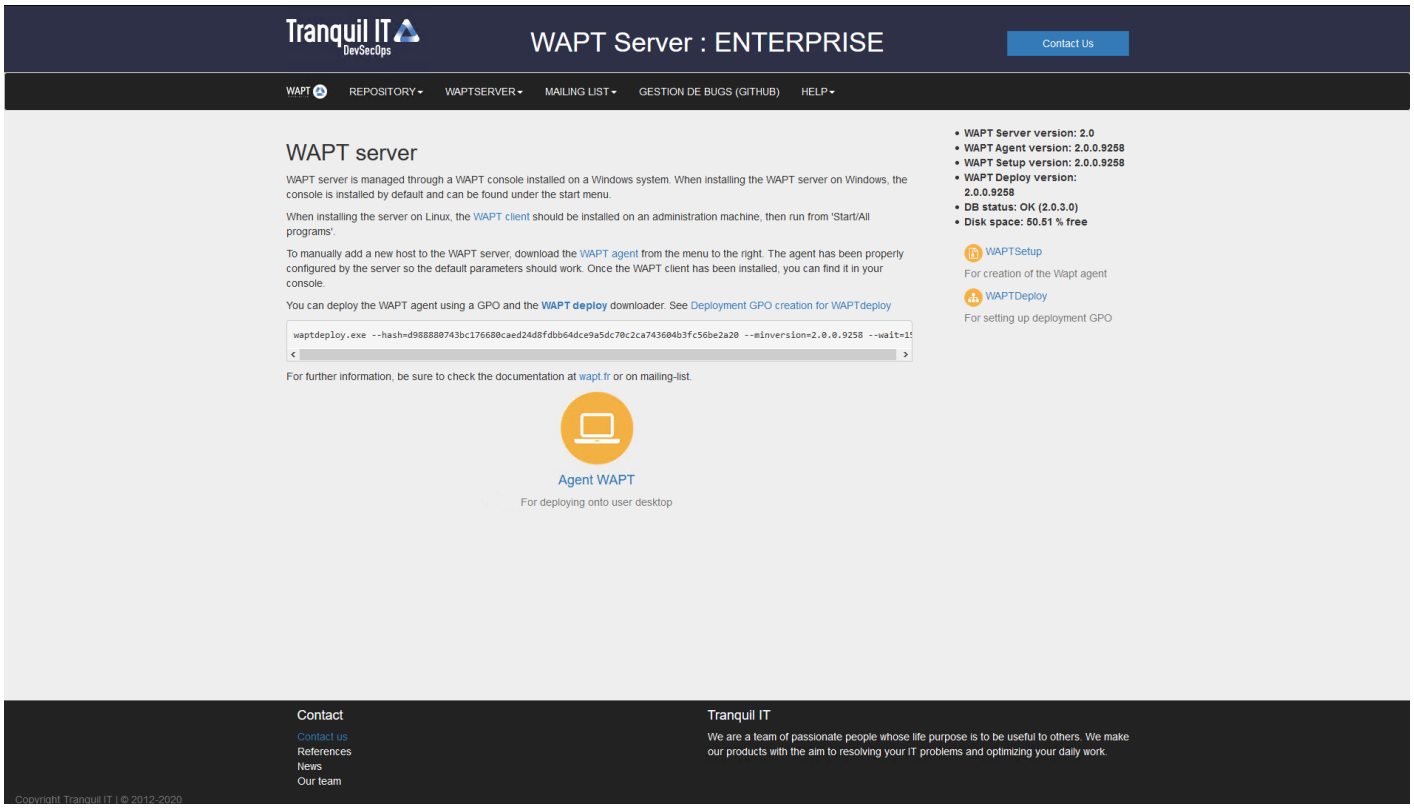


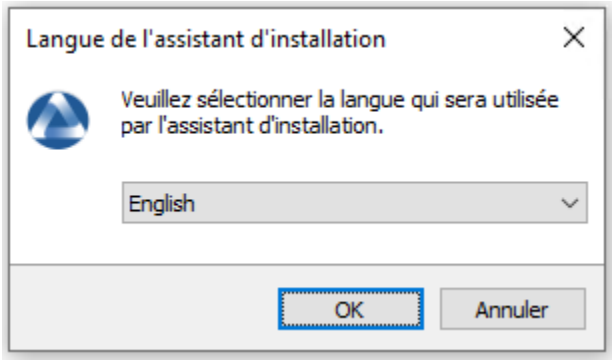
FIG. 1 – L'interface du serveur WAPT dans un navigateur web

— Si les enregistrements DNS sont correctement configurés, vous devriez pouvoir accéder à l'interface web WAPT en vous rendant à l'adresse suivante : https://srvwapt.mydomain.lan.
— Cliquez sur le lien *WAPTSetup* sur le côté droit de la page web du serveur WAPT.

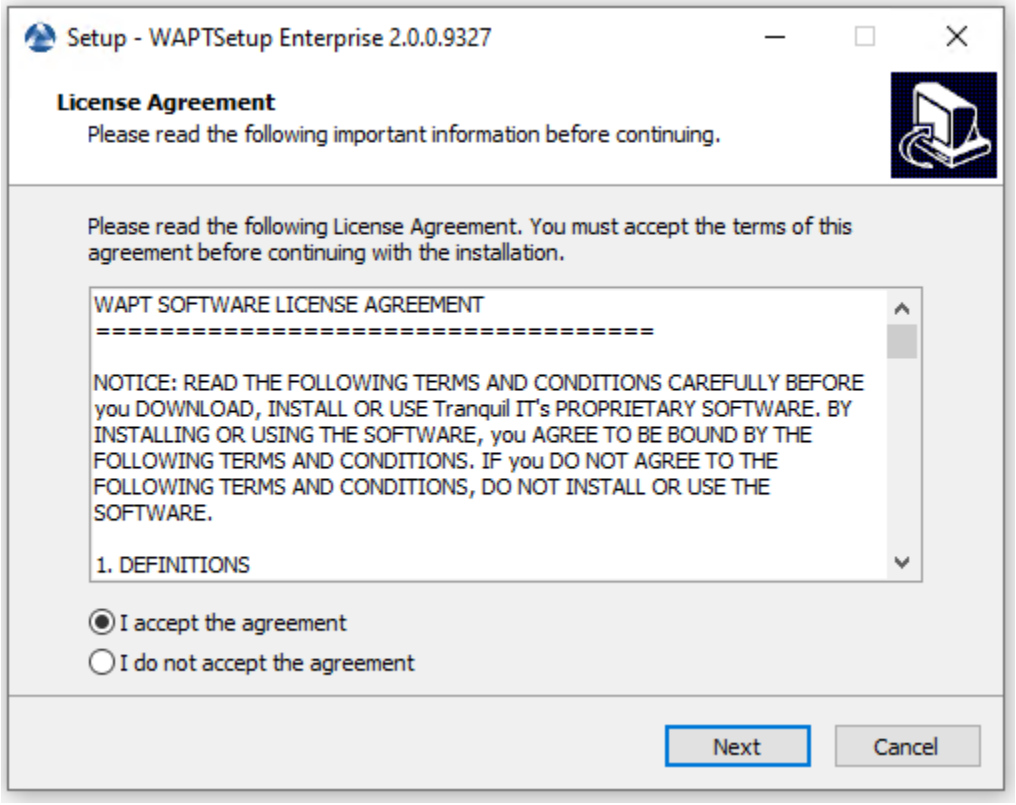
12.1 Installation sur l'ordinateur de l'administrateur

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*.
— Choix de 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).

Tableau 1 – 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

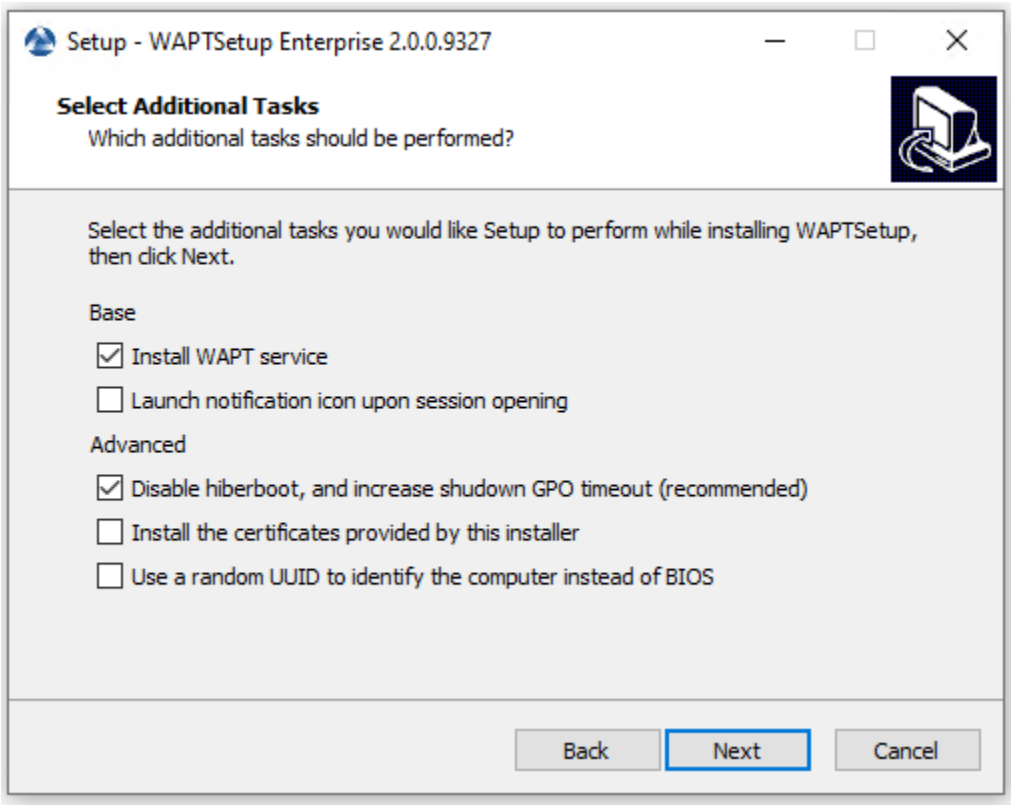


FIG. 2 – Choisir les options de l'installateur

— 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.

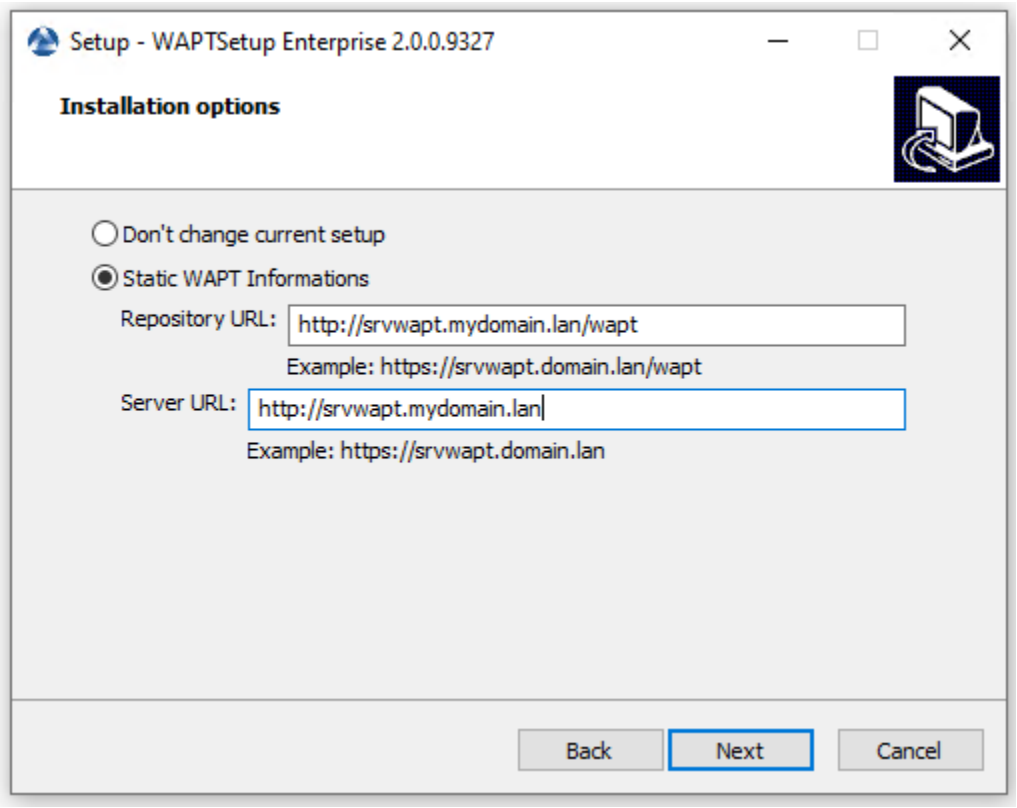


FIG. 3 – Choix du dépot et du serveur WAPT

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

— 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).

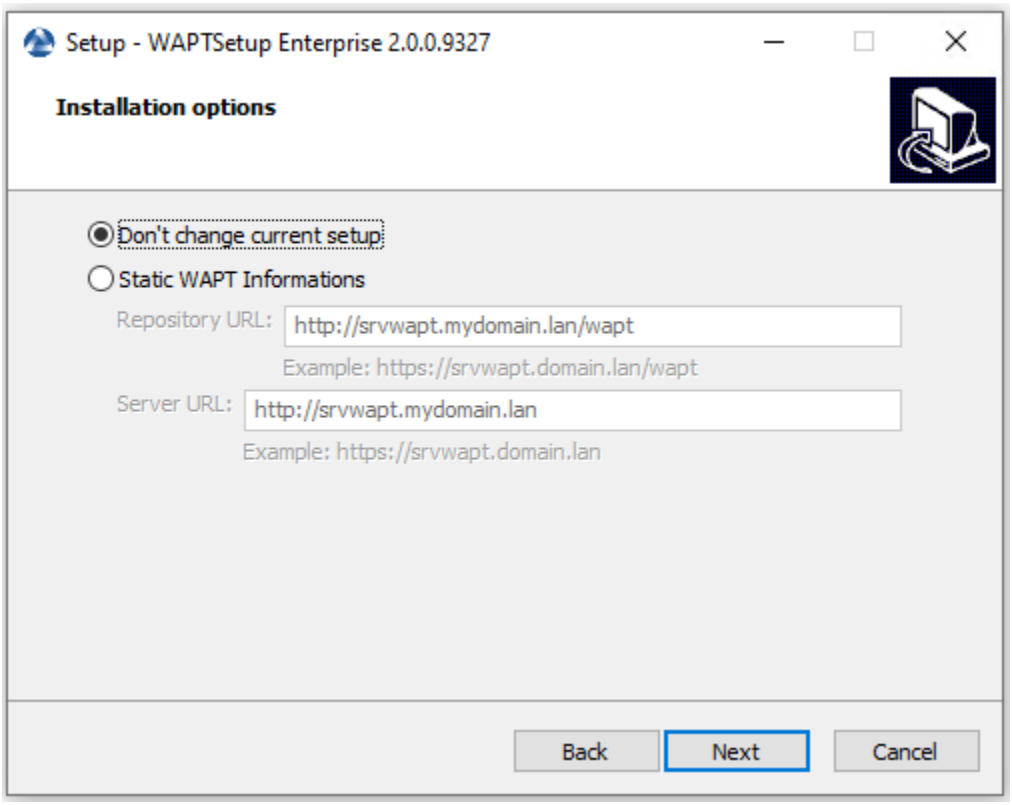


FIG. 4 – Le dépôt et le serveur WAPT sont déjà configurés

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

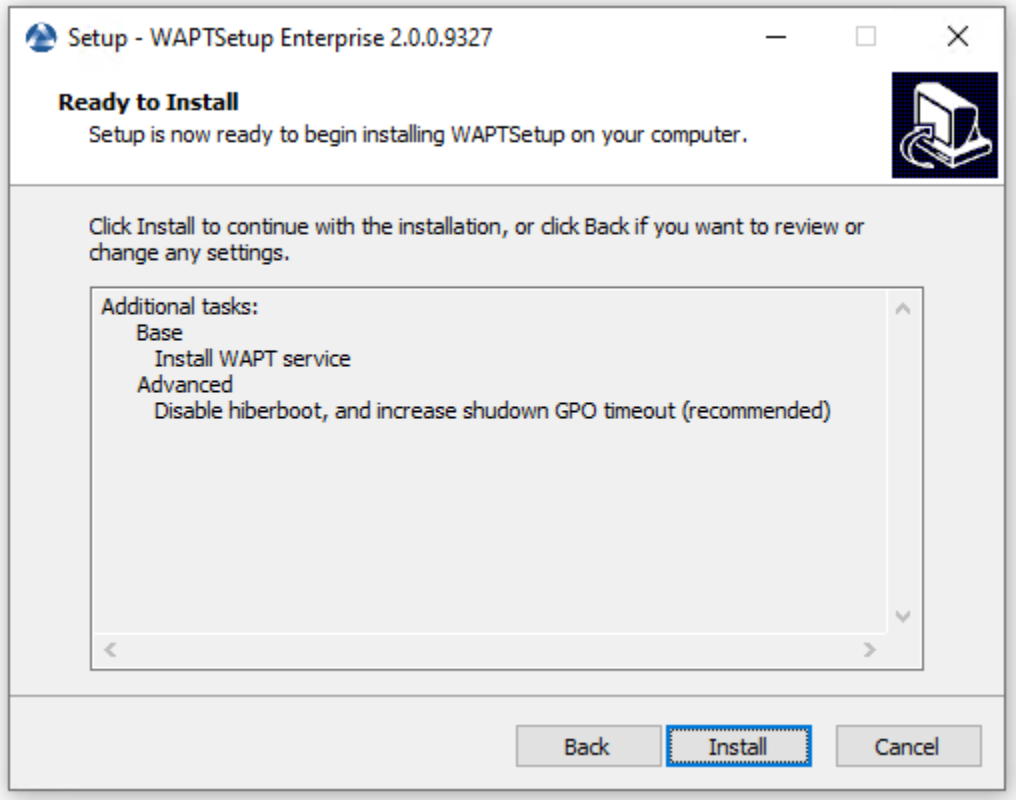
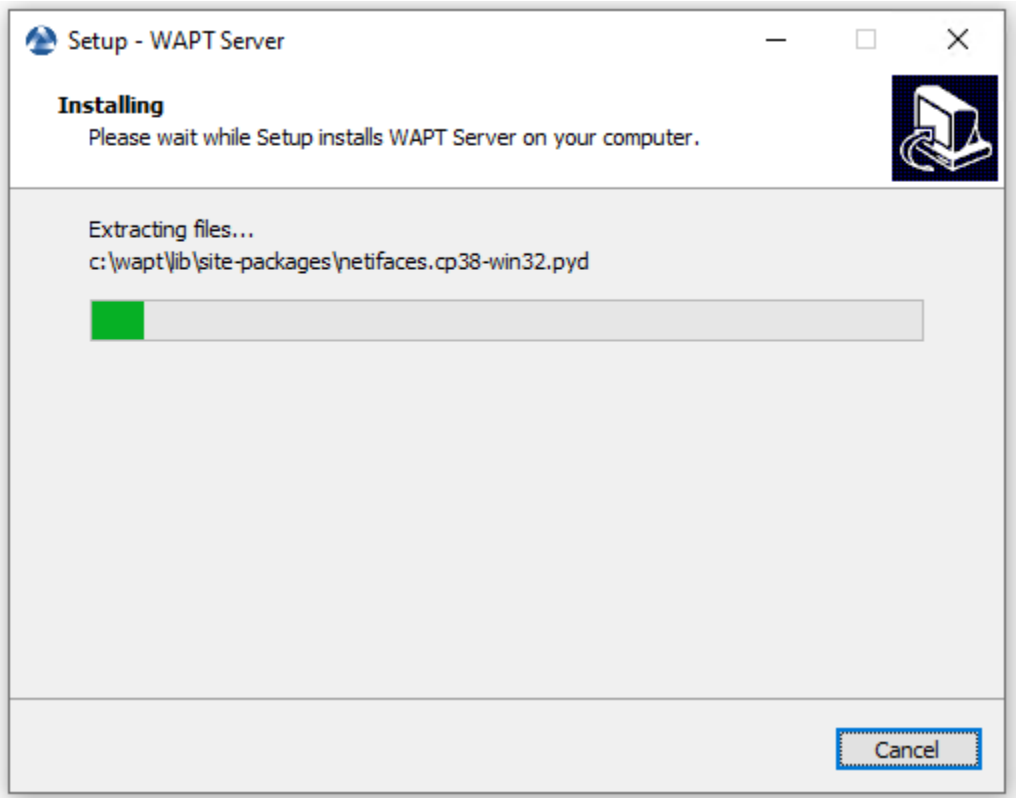
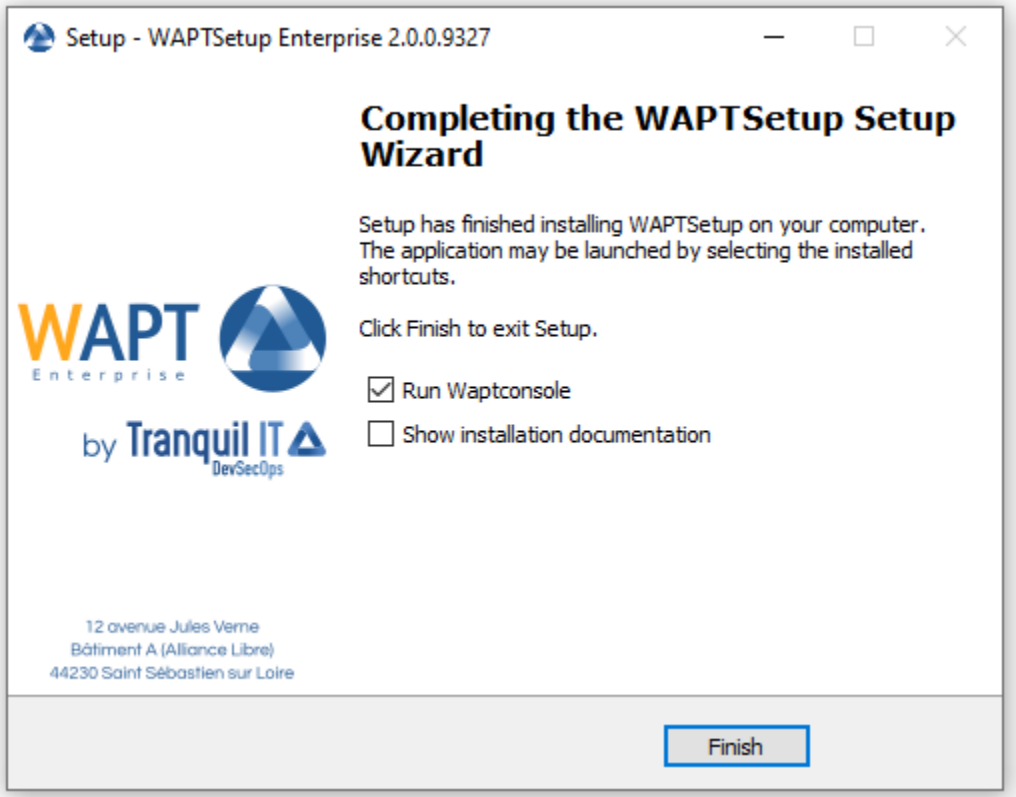


FIG. 5 – Obtenir un résumé de l'installation de la console WAPT.





— Décochez *Afficher la documentation d'installation*.

12.2 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*.

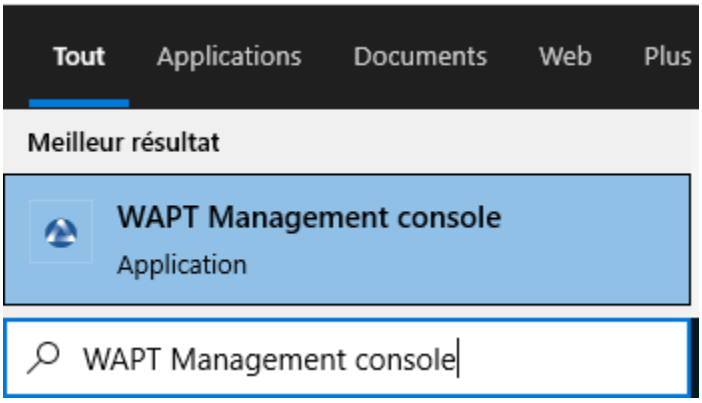


FIG. 6 – Lancement de la console WAPT à partir du menu de démarrage de Windows

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

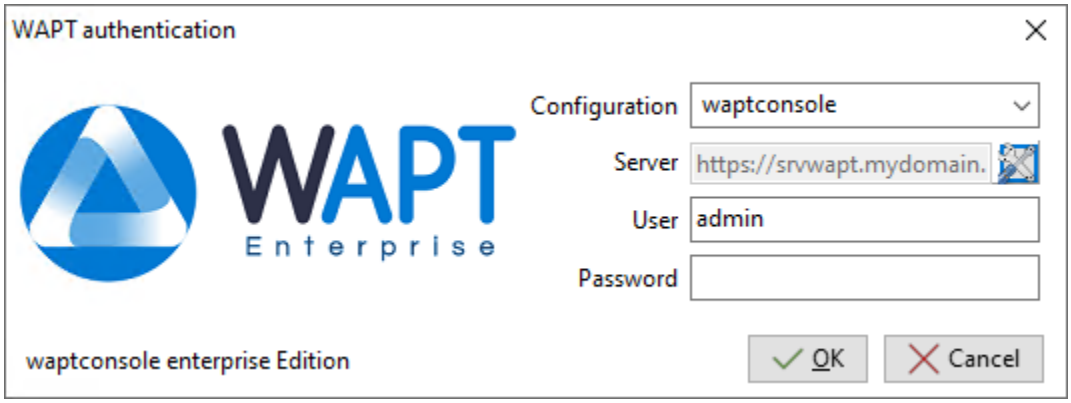


FIG. 7 – La fenêtre d'authentification de la console WAPT

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*.

12.2.1 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.

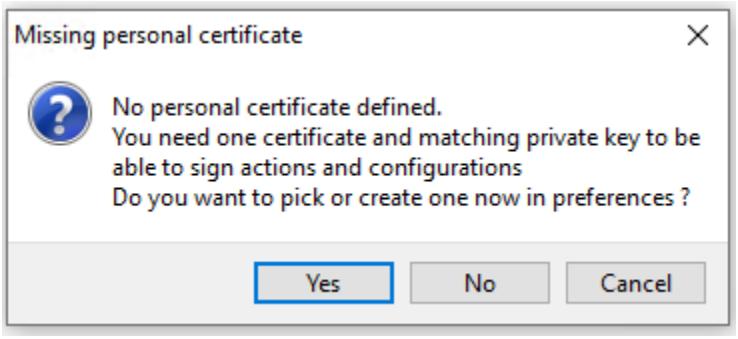


FIG. 8 – Certificat personnel WAPT non présent

— 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.

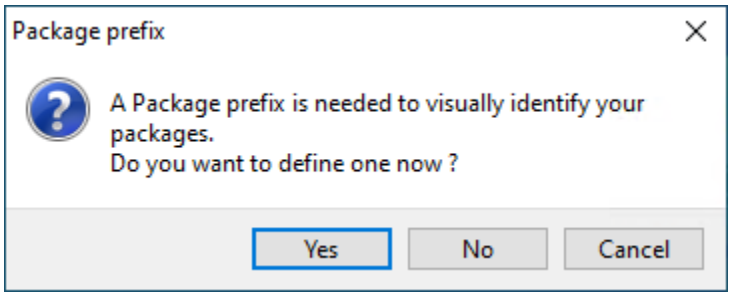


FIG. 10 – Boîte de dialogue informant qu'aucun préfixe n'a été défini dans la configuration WAPT

— Sélectionnez *Oui*

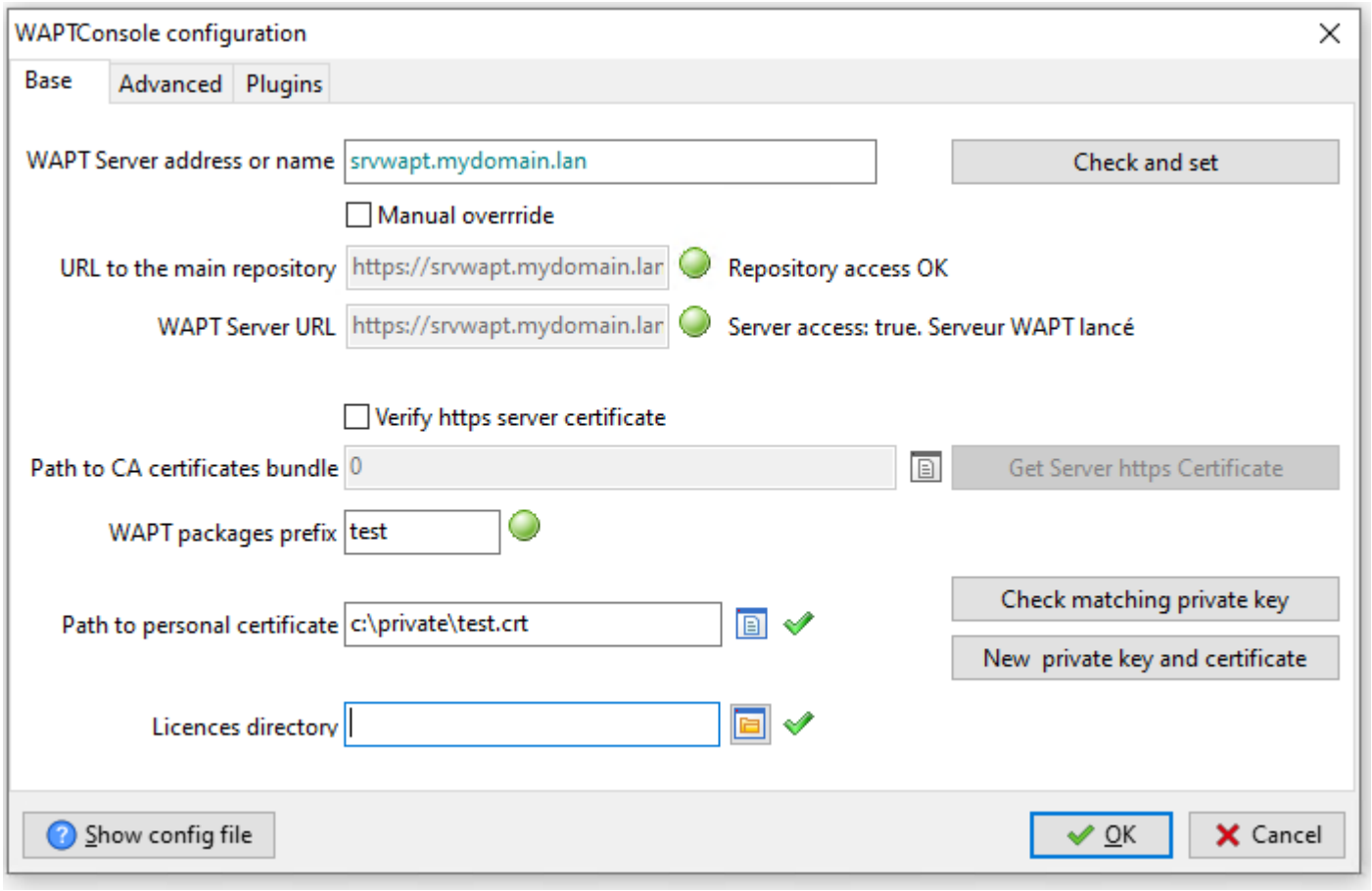


FIG. 9 – Fenêtre pour la configuration basique de la console WAPT

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

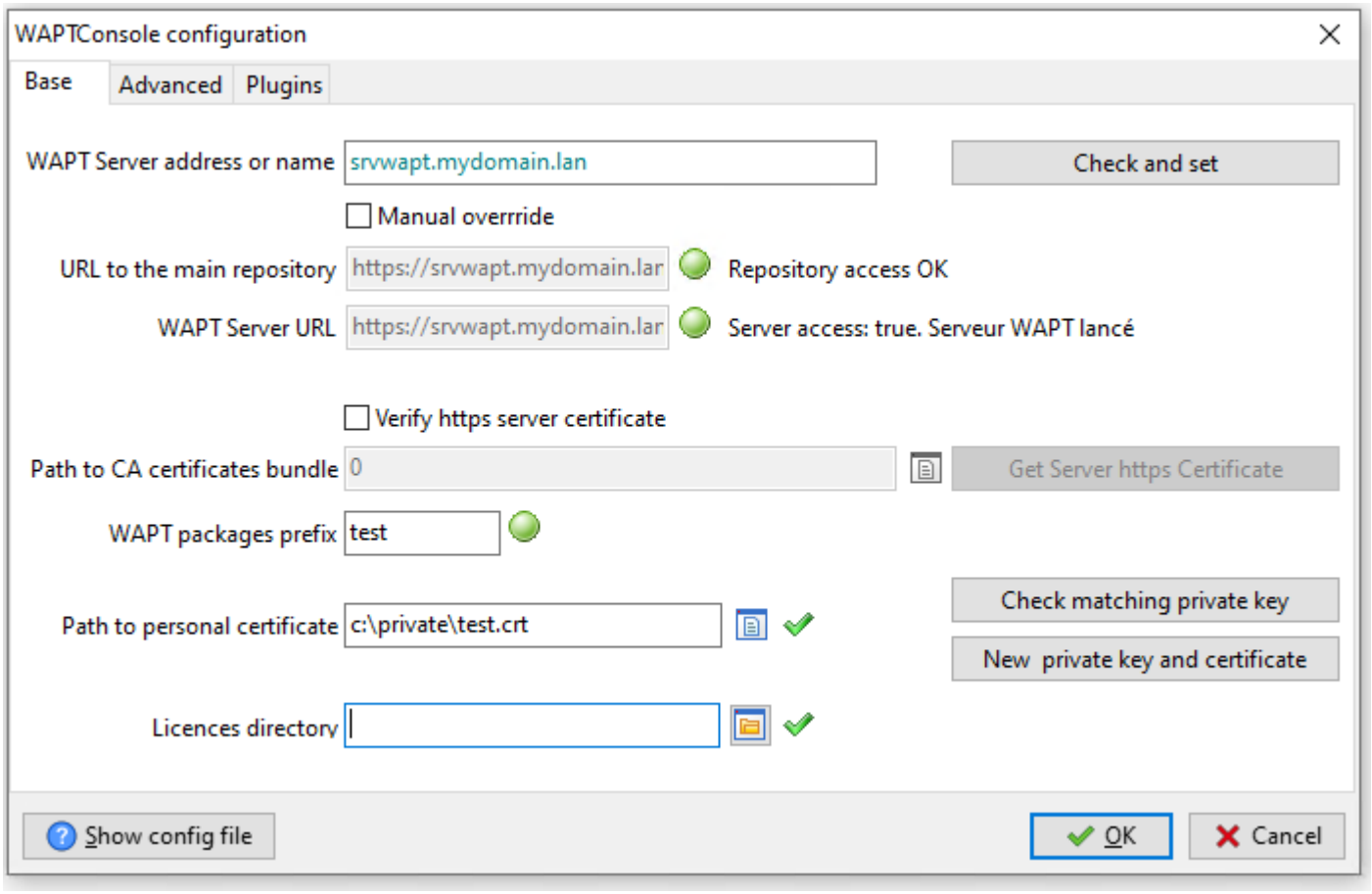


FIG. 11 – Fenêtre pour la configuration basique de la console WAPT

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*.

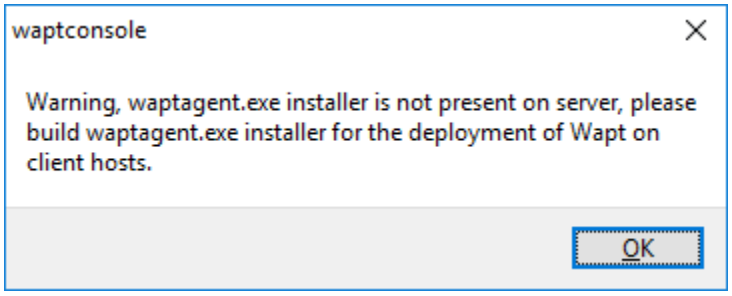


FIG. 12 – Boîte de dialogue informant que l'agent WAPT n'est pas présent sur le serveur WAPT

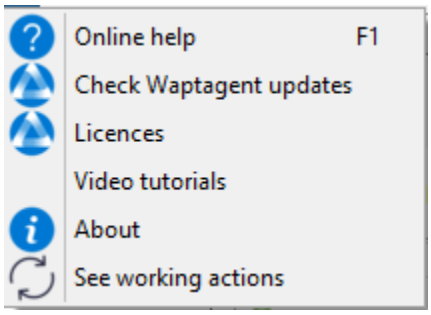
CHAPITRE 13

Activer la licence

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

Indication : 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* :
— Sélectionnez votre `licence.lic` et cliquez sur *Ouvrir*.

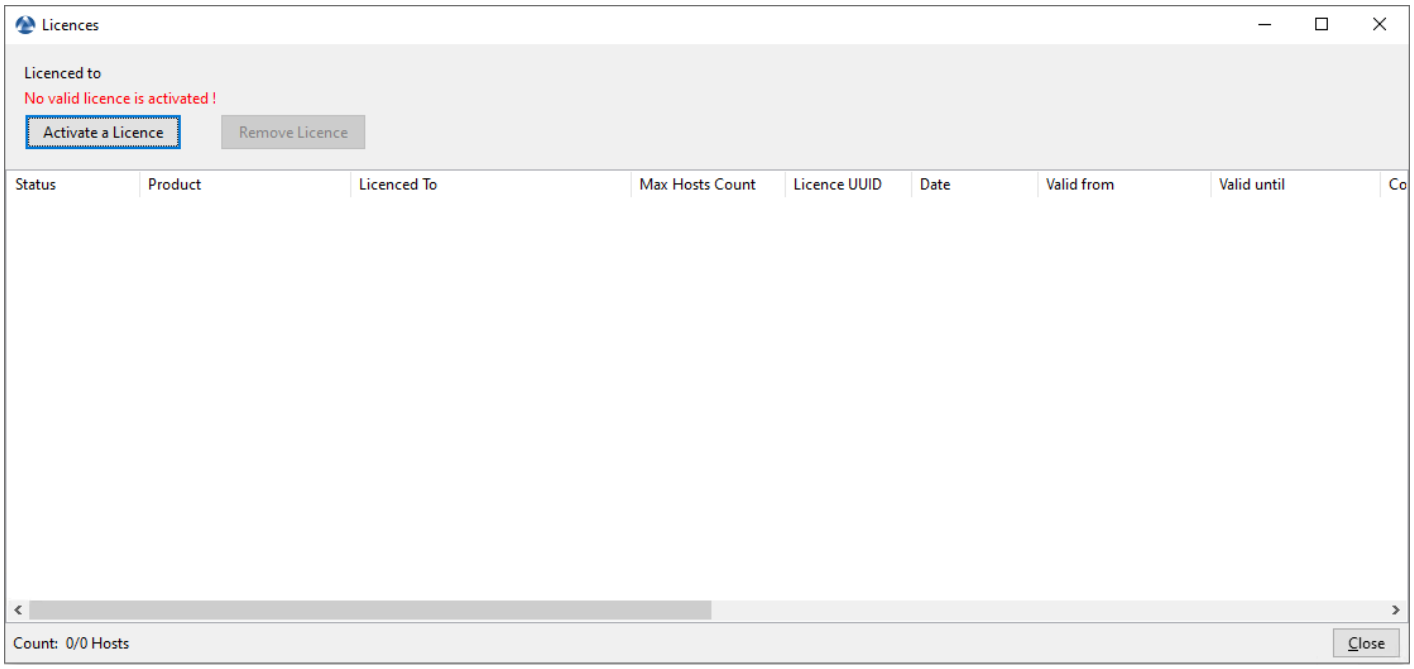


FIG. 1 – Fenêtre indiquant qu'il n'y a pas de licences WAPT souscrites dans la console WAPT

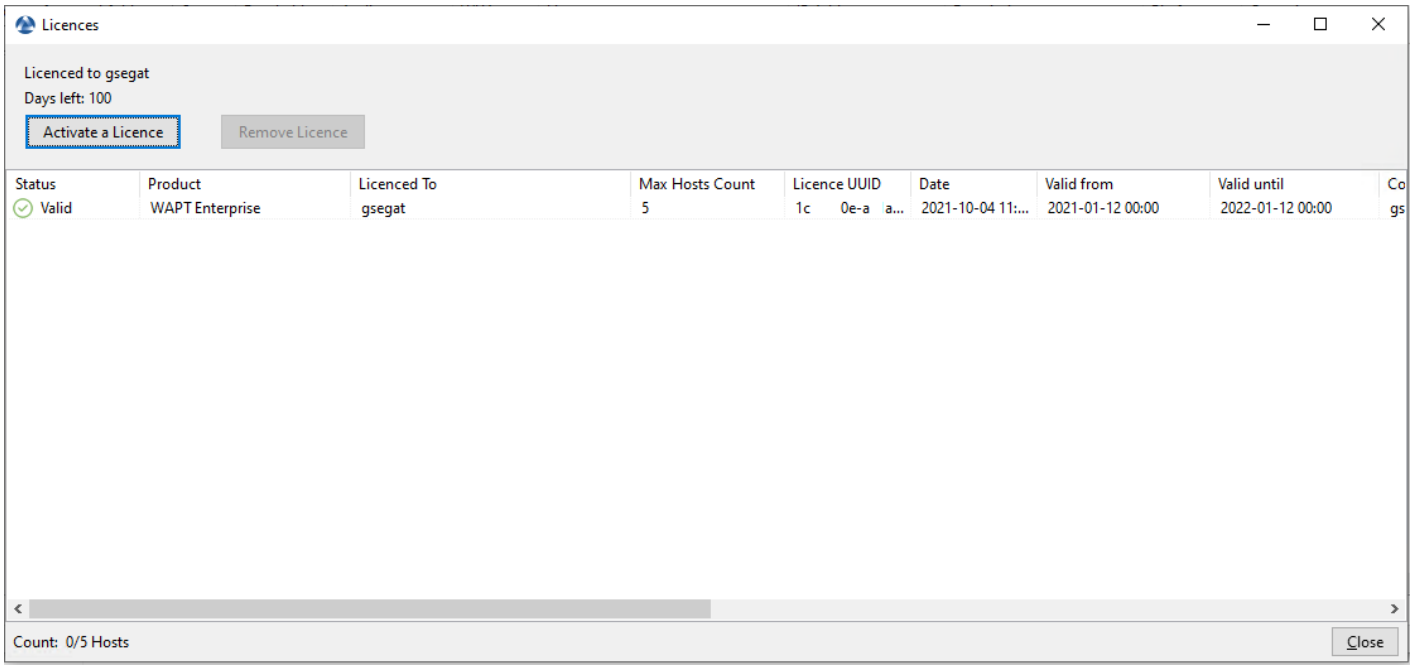
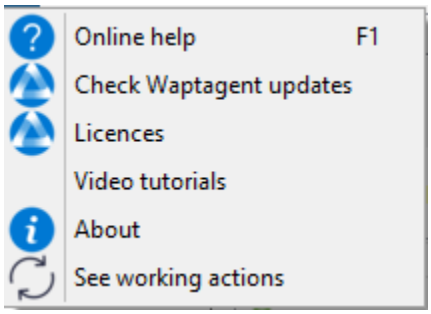


FIG. 2 – Fenêtre montrant une licence activée dans la console WAPT

13.1 Supprimer la licence

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



— Choisissez *Licences* :

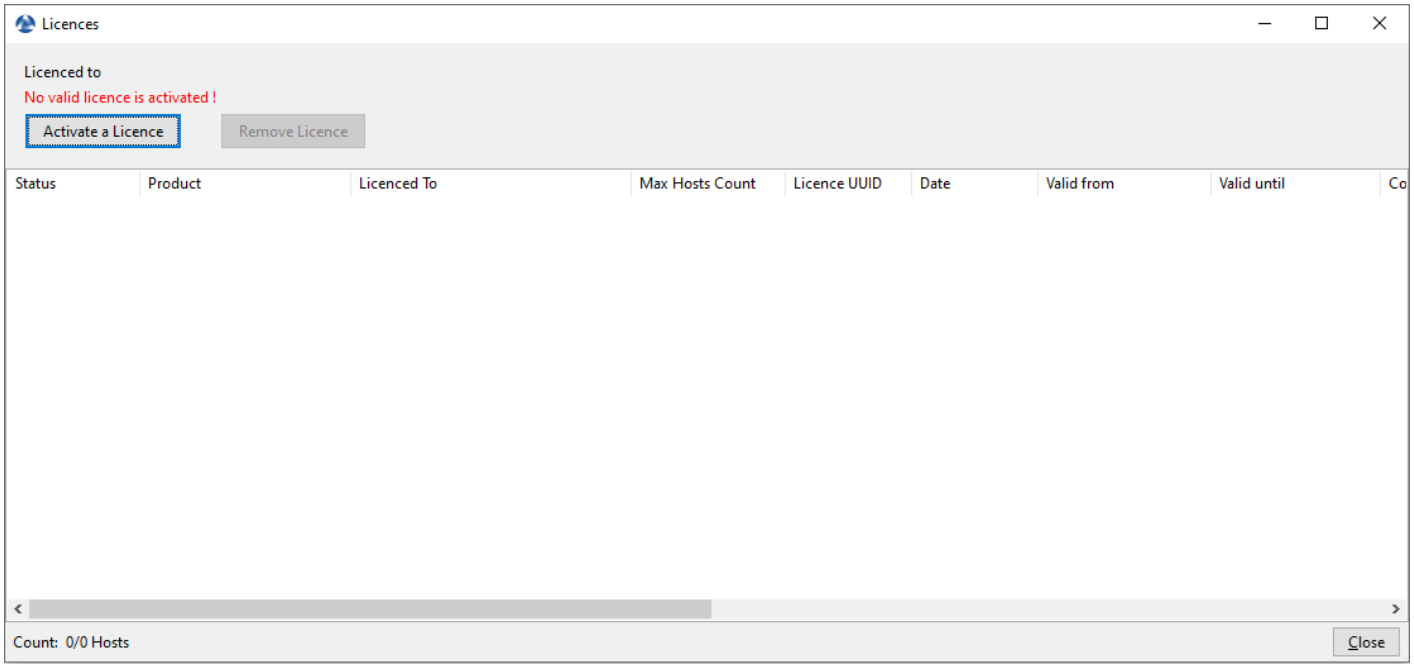


FIG. 3 – Fenêtre indiquant qu'il n'y a pas de licences WAPT souscrites dans la console WAPT

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

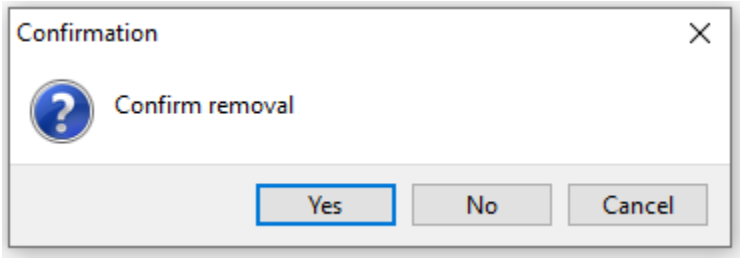


FIG. 4 – Fenêtre de confirmation pour retirer une licence de la Console WAPT

— Après confirmation, la licence est retirée :

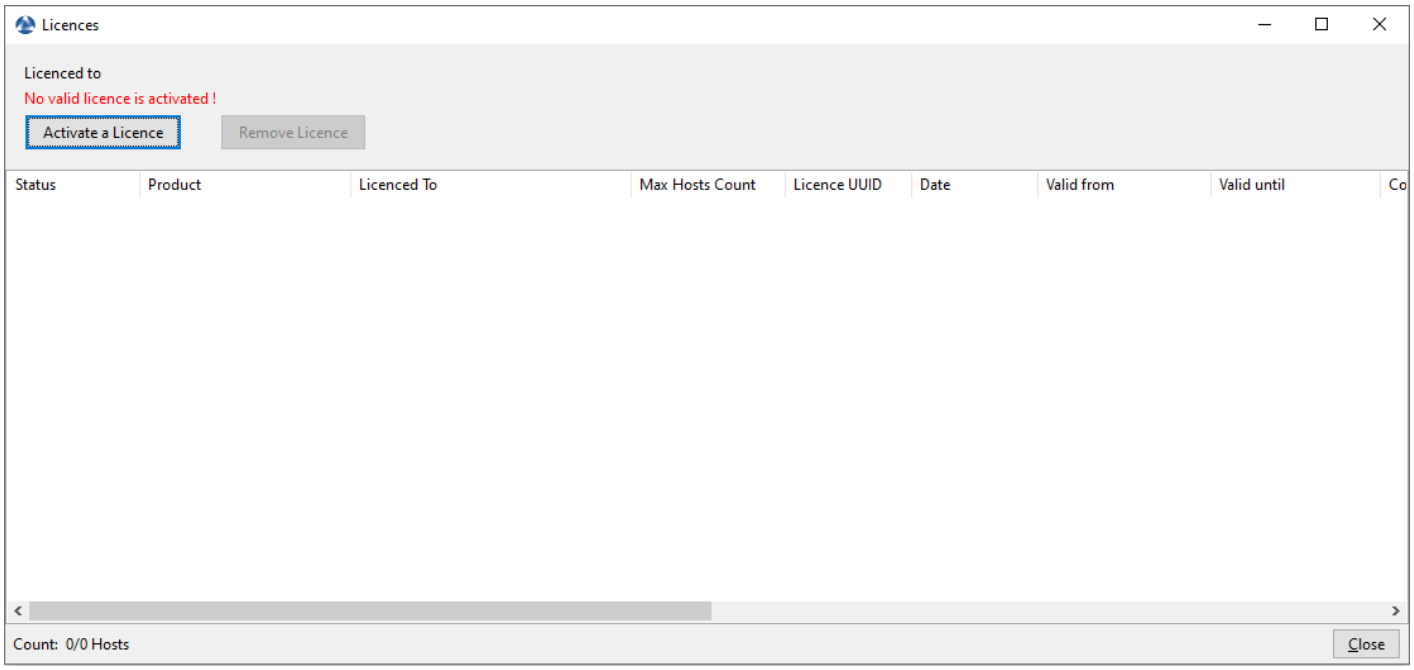


FIG. 5 – Fenêtre indiquant qu'il n'y a pas de licences WAPT souscrites dans la console WAPT

13.2 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
```

13.3 Erreur de licence

13.3.1 Expiration de la licence

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

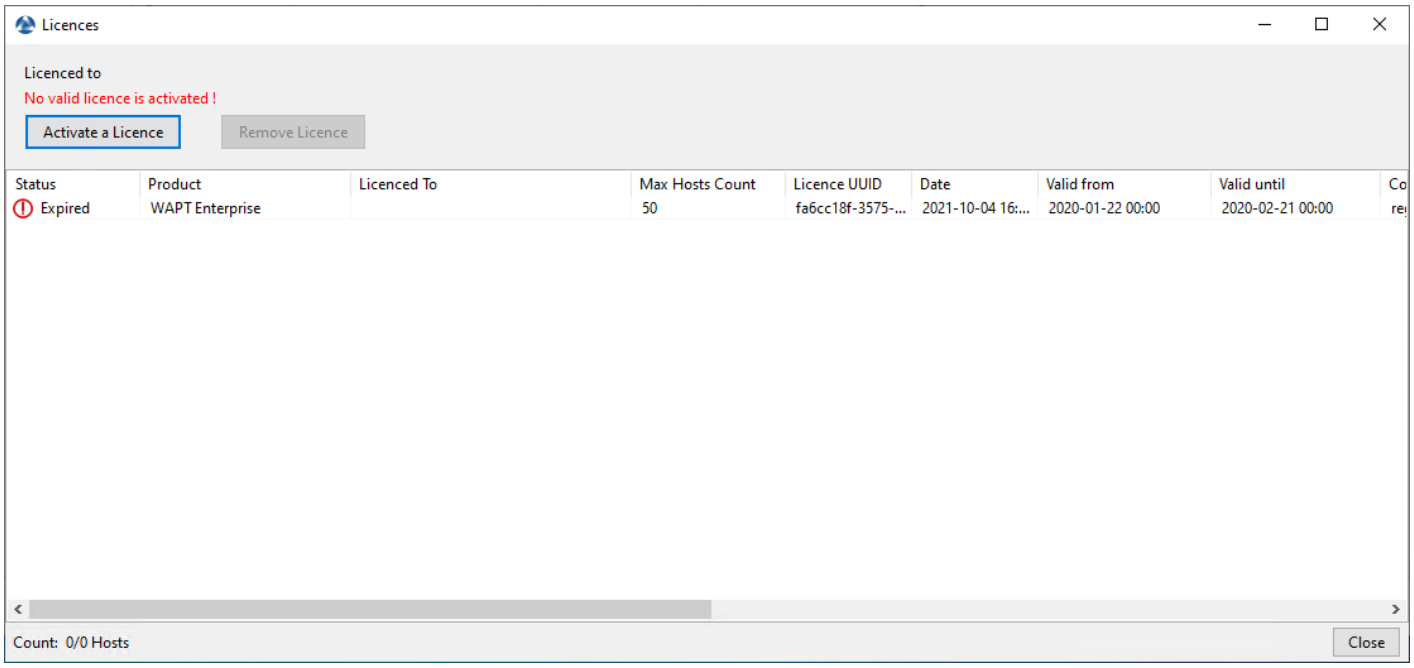


FIG. 6 – Fenêtre montrant une licence expirée dans la console WAPT

13.3.2 Emplacement de l'ancienne licence

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

13.3.3 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`.

```
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
```

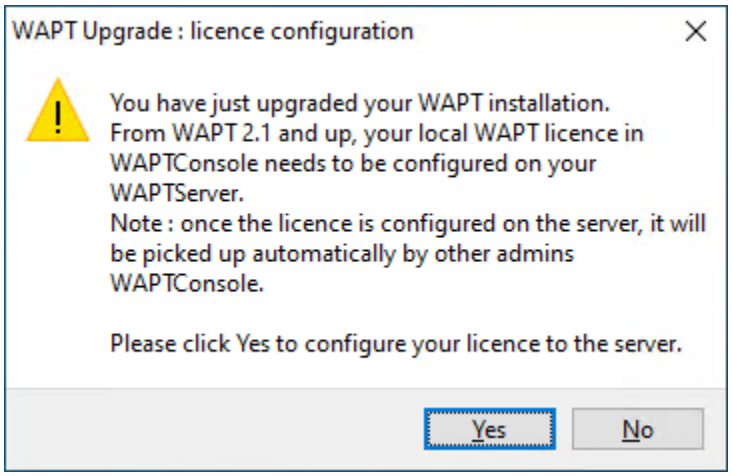


FIG. 7 – Message d'erreur de la licence WAPT lors de la mise à niveau de WAPT vers 2.1

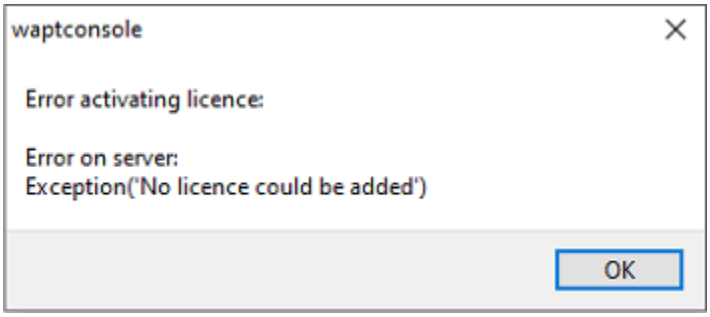


FIG. 8 – Boîte de dialogue informant qu'une erreur s'est produite lors de l'activation d'une licence WAPT

— 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
```

2. Vérifier si `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 *licence activation*

CHAPITRE 14

Génération du certificat principal

Indication :
- Le nom de la clé privée est `wapt-private.pem`.
- Le nom du certificat public signé avec la clé privée est `wapt-private.crt`.

14.1 Clef privée *wapt-private.pem*

Attention : 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` 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.

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

14.2 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.

14.3 Génération d'un certificat à utiliser avec WAPT

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

Note : Avec WAPT Enterprise, vous pouvez créer une clé principale d'autorité de certification qui peut à la fois signer des paquets et signer de nouveaux certificats.

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

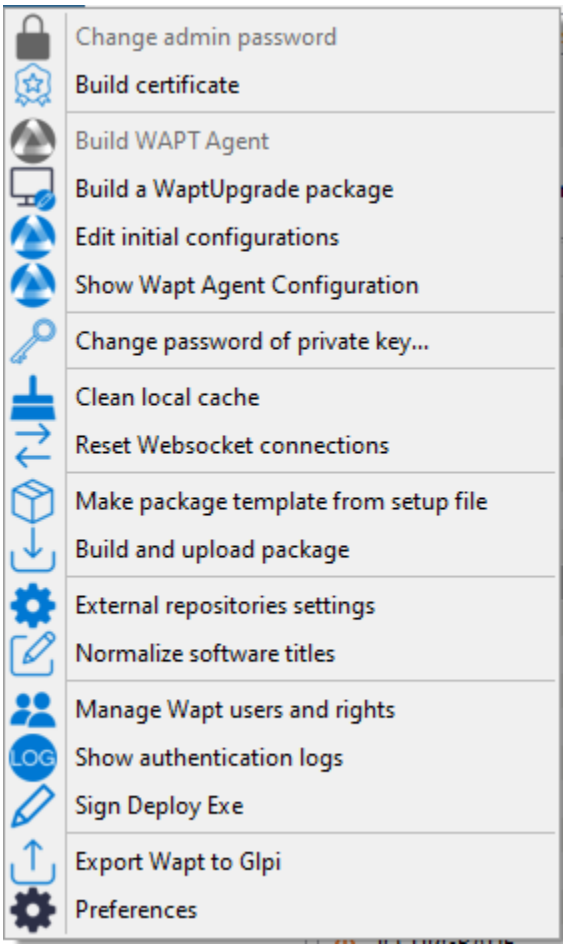


FIG. 1 – Création d'un certificat auto-signé

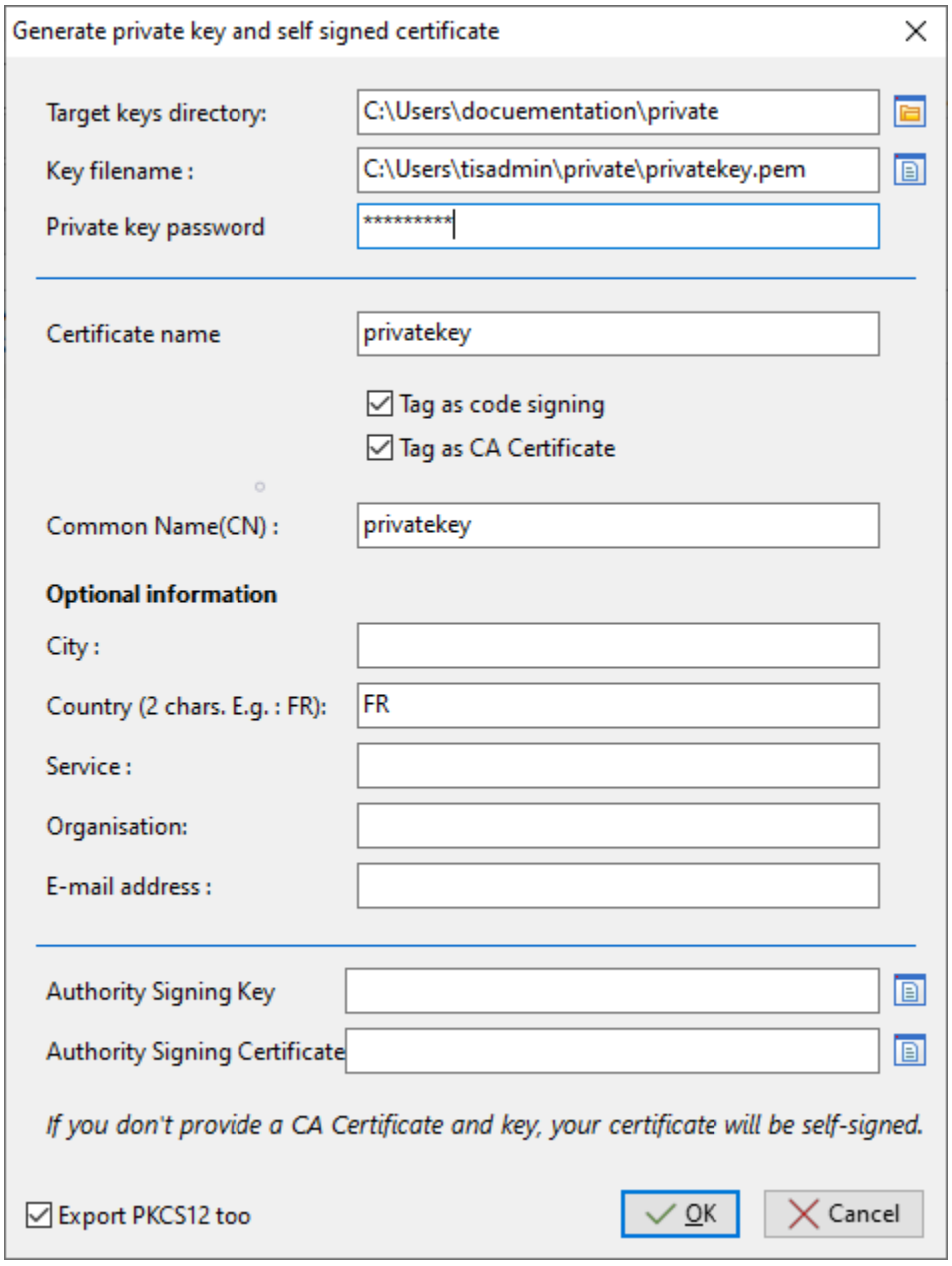


FIG. 2 – Création d'un certificat auto-signé pour la version WAPT Enterprise

Tableau 1 – 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.

Indication : 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 chemin d'accès à votre clé privée ne doit pas se trouver dans le chemin d'installation de WAPT (`C:\Program Files (x86)\wapt`).
- Si votre clé est stockée dans `C:\Program Files (x86)\wapt`, votre clé privée d'administrateur sera déployée sur vos clients, **ce qui est absolument à proscrire !**.
- 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 :

— Sélectionnez *Oui*

— Cliquez sur *Yes* pour copier le certificat nouvellement généré dans le dossier `C:\Program Files (x86)\wapt\ssl` sous

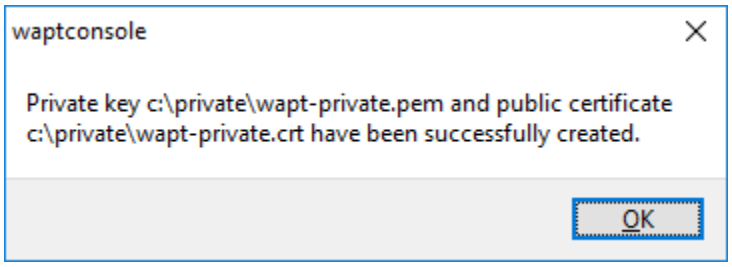


FIG. 3 – Le certificat a été généré avec succès

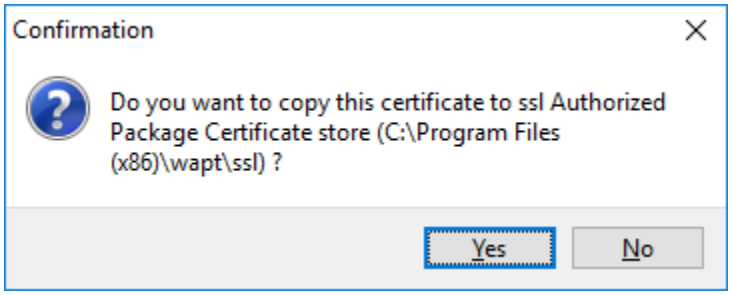


FIG. 4 – Boîte de dialogue demandant la confirmation de la copie du certificat dans le dossier ssl de la console WAPT

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*.

CHAPITRE 15

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 :

— Les fichiers qui seront utilisés lors de la construction de l'agent WAPT sont situés dans `C:\Program Files (x86)\wapt`.
— Les fichiers sources du programme d'installation (fichiers `.iss`) se trouvent dans `C:\Program Files (x86)\wapt\waptsetup`.

Indication : Avant de construire l'agent WAPT, veuillez vérifier le(s) certificat(s) public(s) dans `C:\Program Files (x86)\wapt\ssl`.

Si vous souhaitez déployer d'autres certificats publics sur les ordinateurs de votre *Organisation* qui sont équipés de WAPT, vous devrez les copier dans ce dossier.

Danger : NE COPIEZ PAS la clé privée d'un *Administrator* dans `C:\Program Files (x86)\wapt`.

Ce dossier est utilisé lors de la construction de l'agent WAPT et les clés privées seront ensuite déployées sur tous les ordinateurs.

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

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

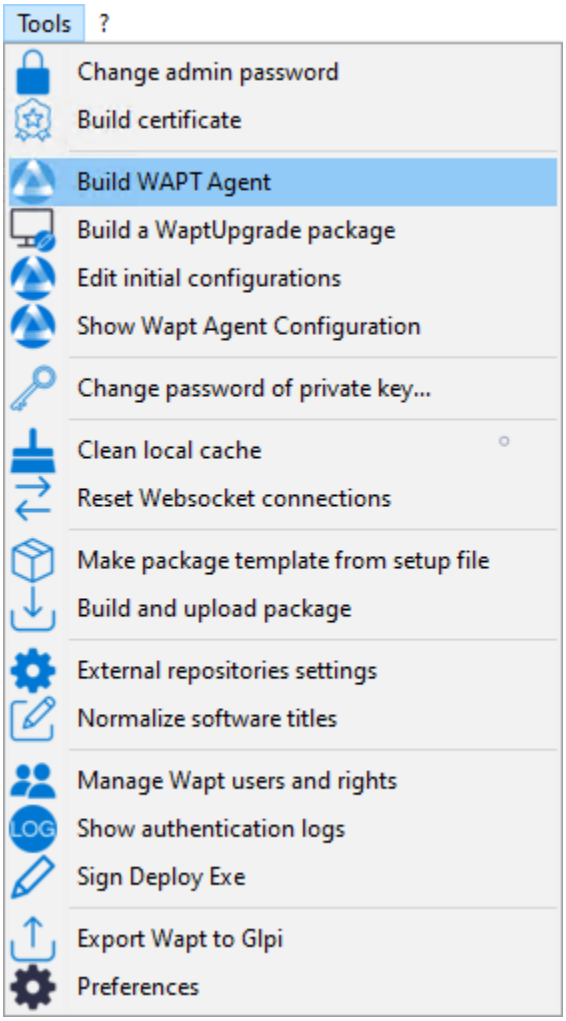


FIG. 1 – Générer l'agent WAPT depuis la console

15.1 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 de votre parc portent le même nom. Nous savons tous que cela ne devrait pas arriver !

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

15.2 Construire

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

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

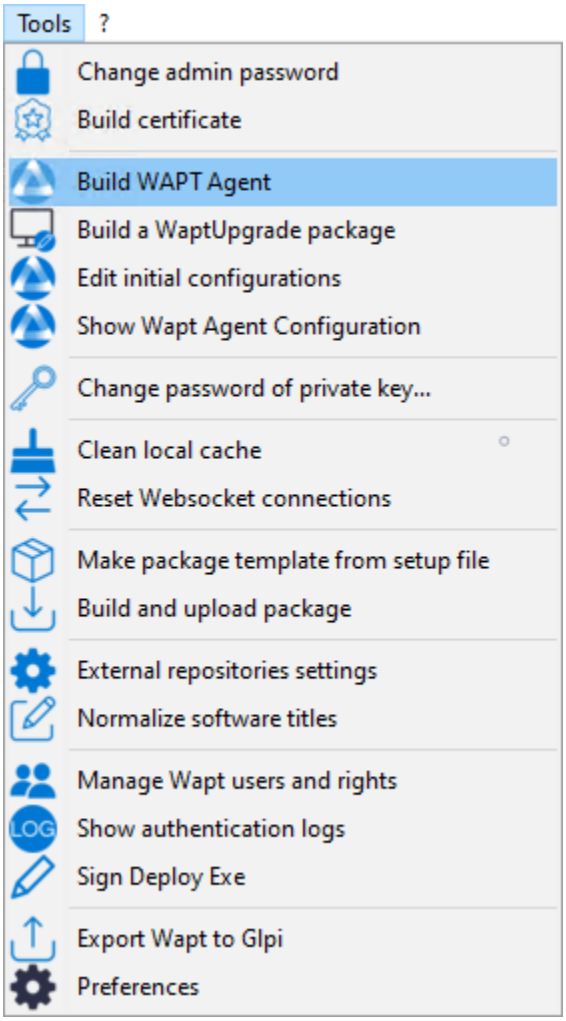


FIG. 2 – Générer l'agent WAPT depuis la console

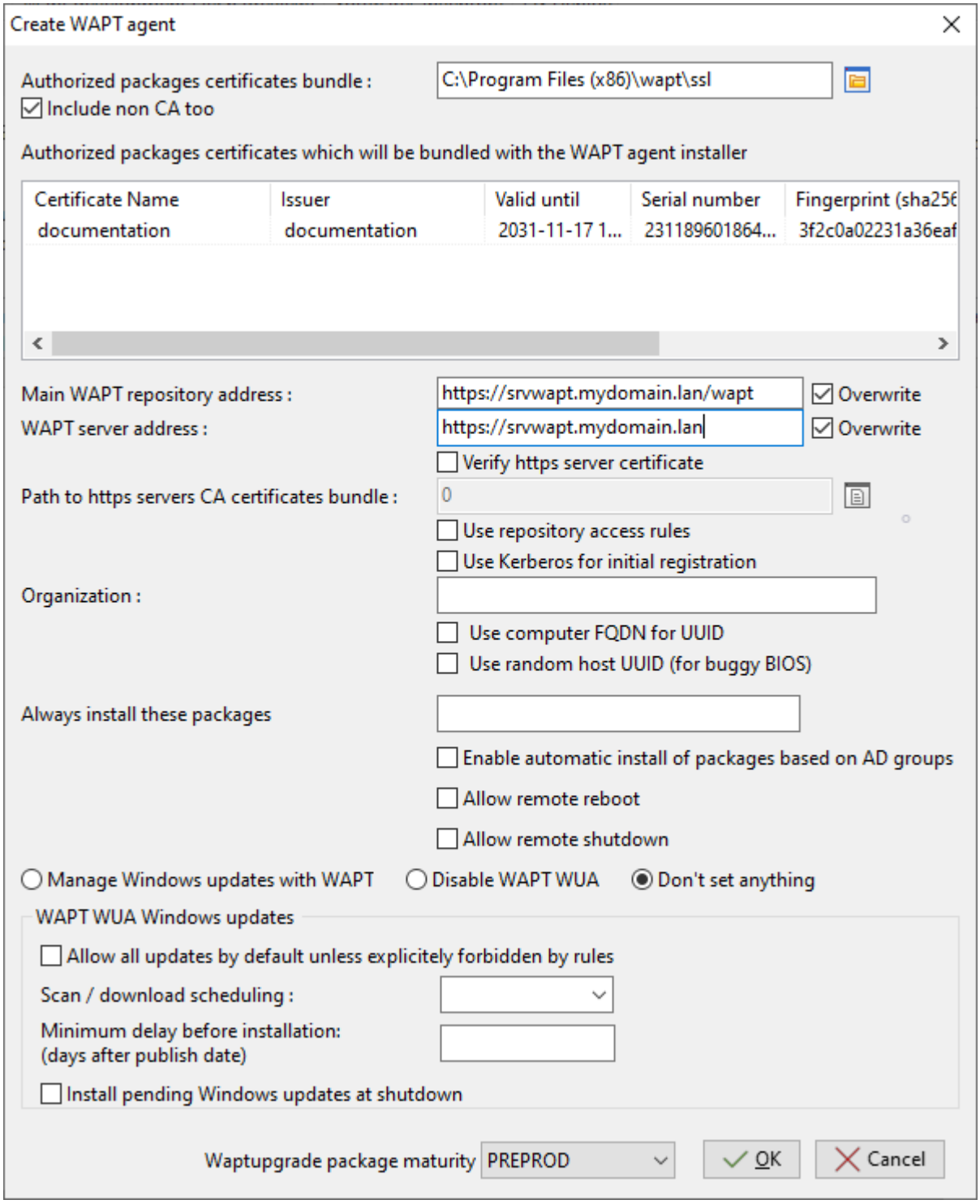


Fig. 3 – Remplir les informations sur votre organisation

Tableau 1 – 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	Allows to choose the maturity of the waptupgrade package.	❌	+

Indication : 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 sur *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 sur *Activer la vérification du certificat SSL / TLS*.

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

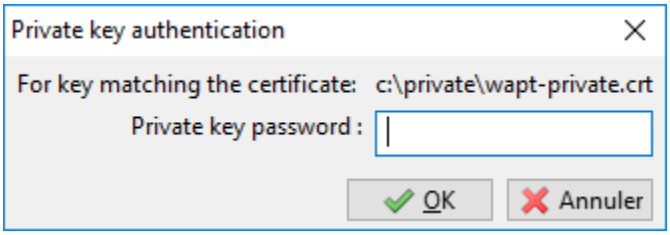


Fig. 4 – Fournissez le mot de passe pour déverrouiller la clé privée.

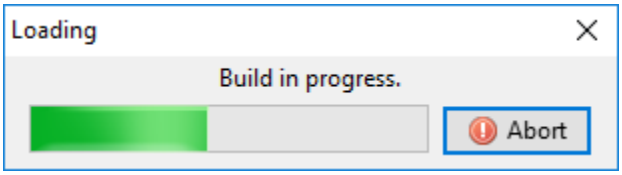


Fig. 5 – Progression de l'installation de l'agent WAPT

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/.

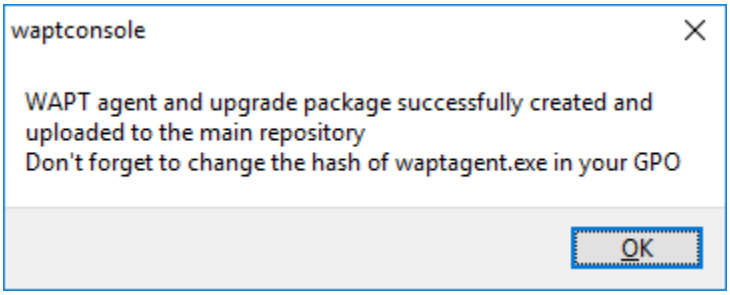


Fig. 6 – Confirmation du chargement de l'agent WAPT sur le référentiel WAPT

Note : 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.

CHAPITRE 16

Configuration initiale

Enterprise

Il est possible de configurer l'agent pour les options standard et avancées via une interface graphique.

— Dans la console WAPT, allez dans *Outils* → *Générer un agent WAPT*
— Remplissez les informations qui sont nécessaires pour l'installateur.

Tableau 1 – En-tête

Valeur	Description
Organisation	Sert à afficher *les options* comme dans `wapt-get.ini`.
Vérifier le certificat https serveur	Ajouter le certificat déployé avec la configuration.
Adresse du serveur WAPT	Charger une configuration précédemment créée
Vérifier le certificat https serveur	Rafraîchit la liste des configurations disponibles
Organisation	Créer une nouvelle configuration
Organisation	Supprimer une configuration

mondial

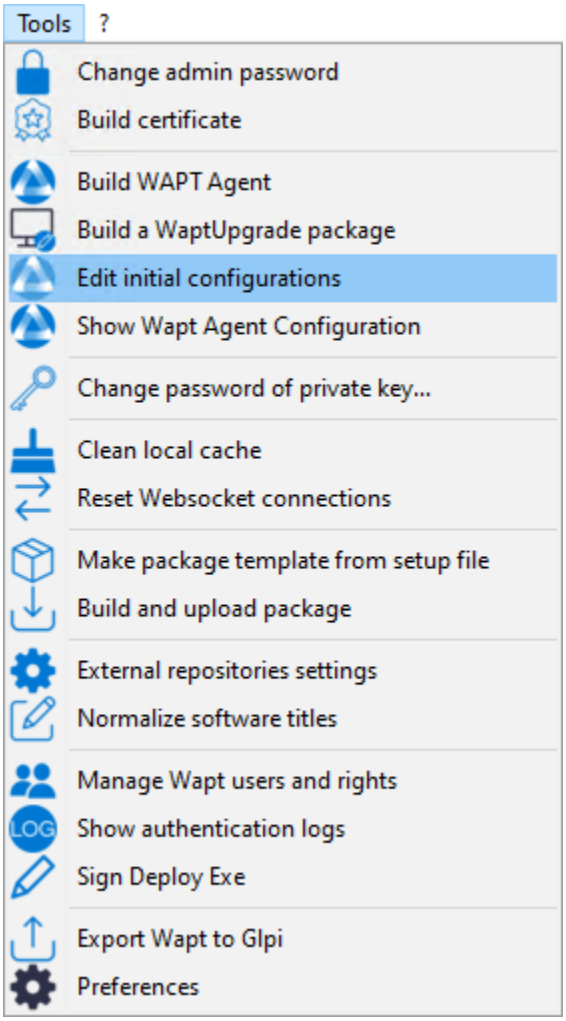


FIG. 1 – Création de la configuration initiale

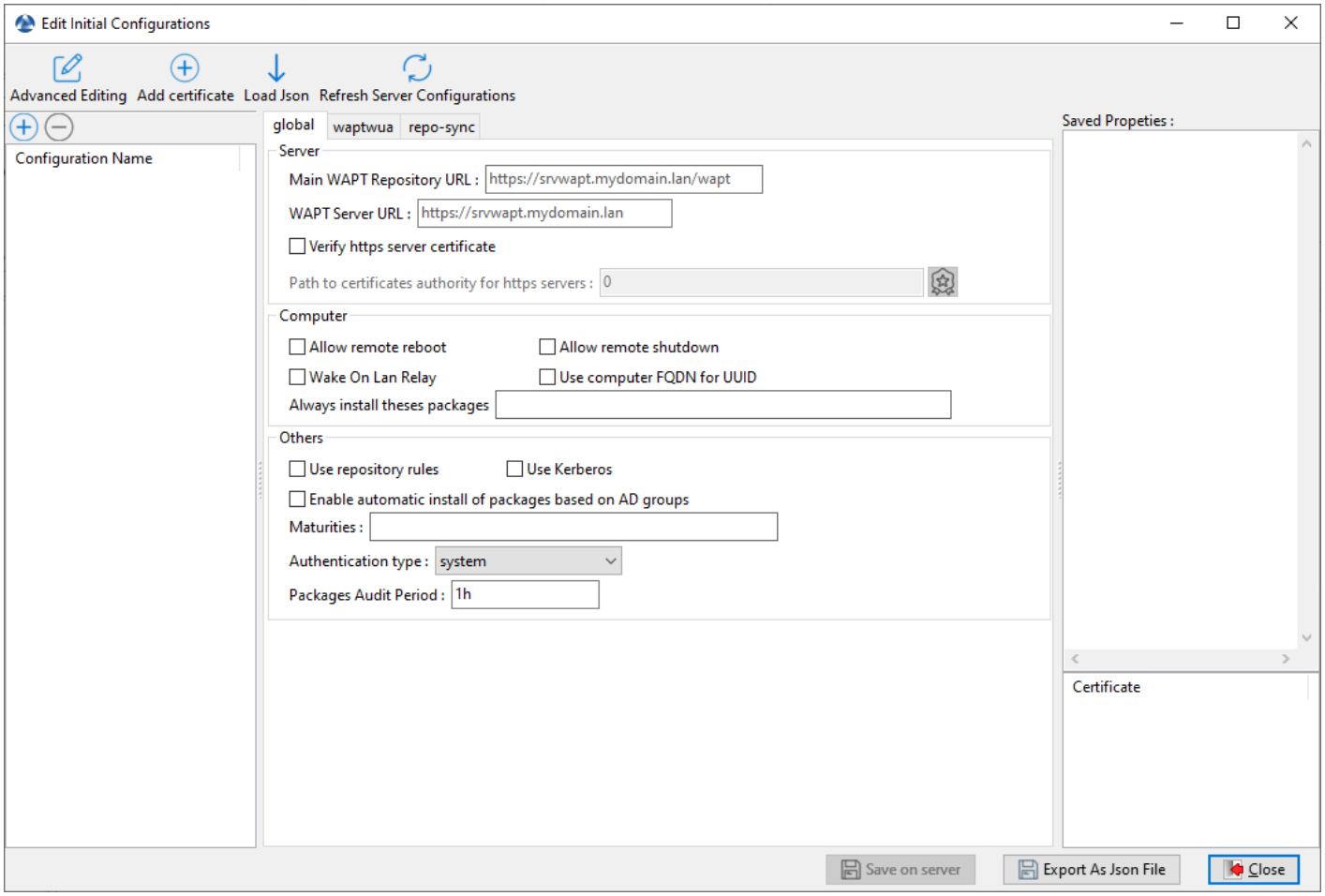


FIG. 2 – Modification de la configuration initiale

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

Enterprise

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

Enterprise

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)	❌

Tableau 2 – Colonne

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

Tableau 3 – 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

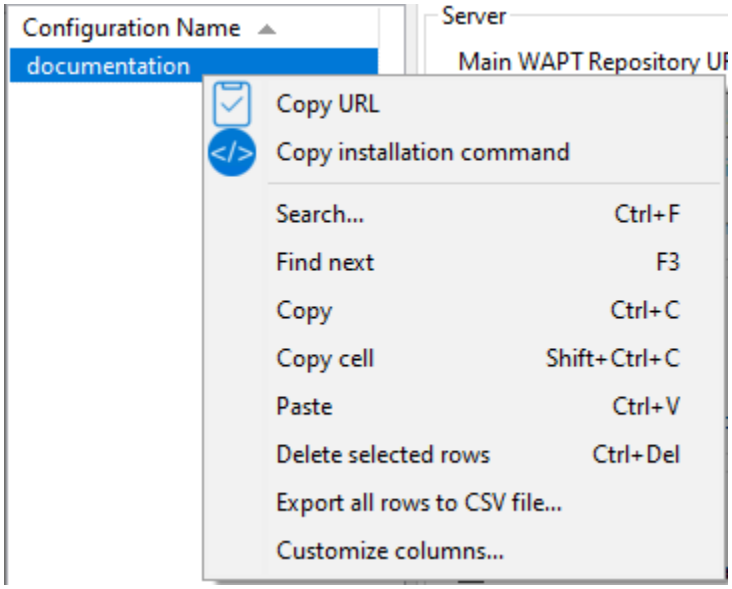


FIG. 3 – Commande de copie

TABLEAU 4 – 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.

CHAPITRE 17

Gestion de l'agent WAPT sous Windows

17.1 Déploiement de l'agent WAPT sur Windows

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 après l'avoir généré, 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 *.exe* avec un certificat `Code Signing` qui peut être vérifié par un faisceau d'AC stocké dans le magasin de certificats de l'hôte.

17.1.1 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.

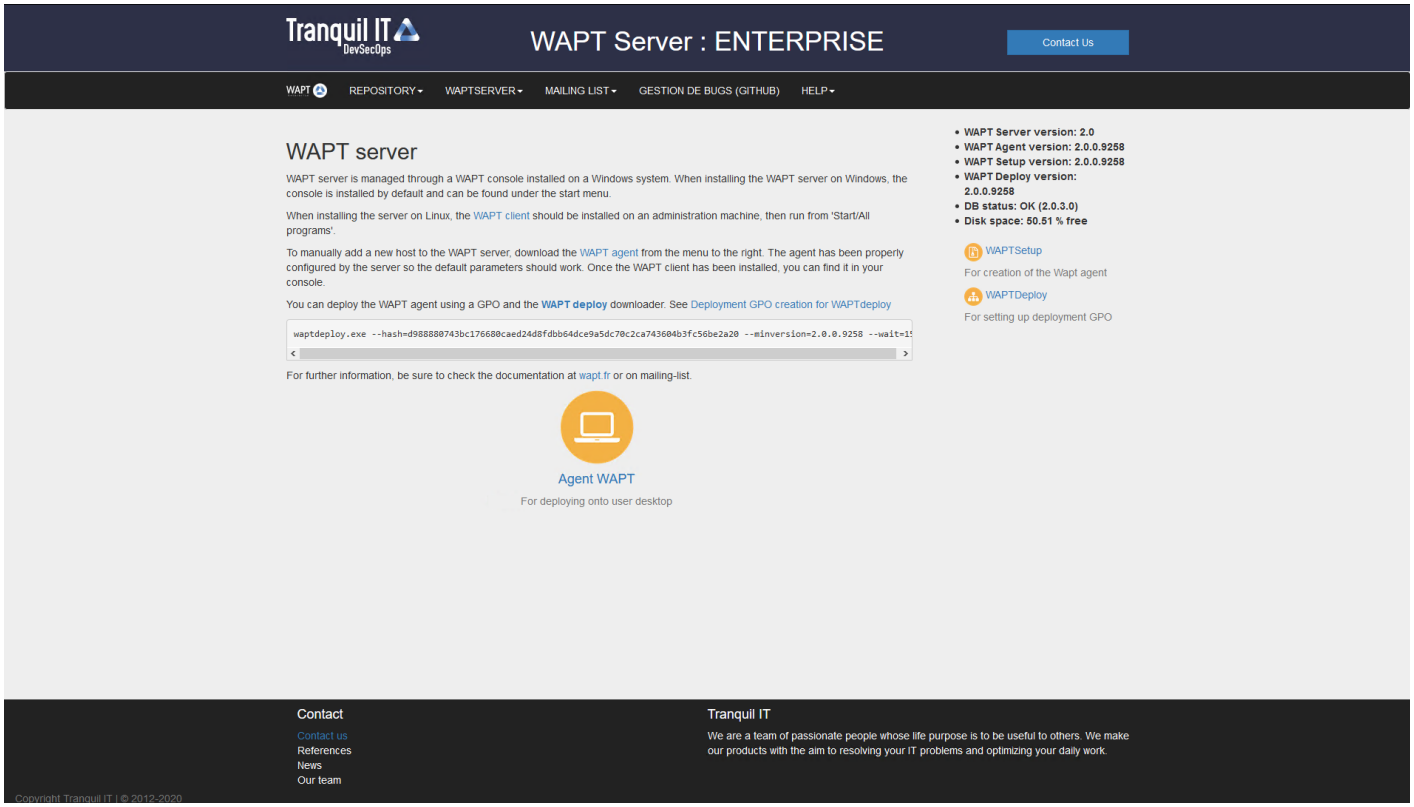
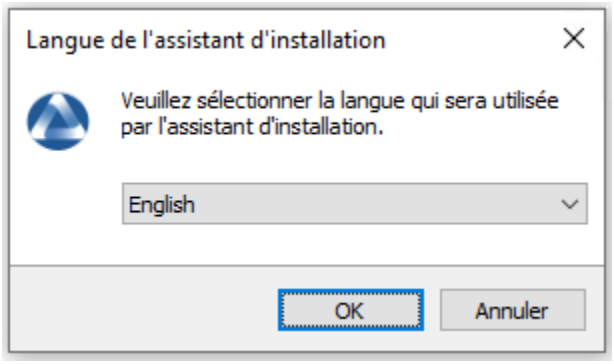
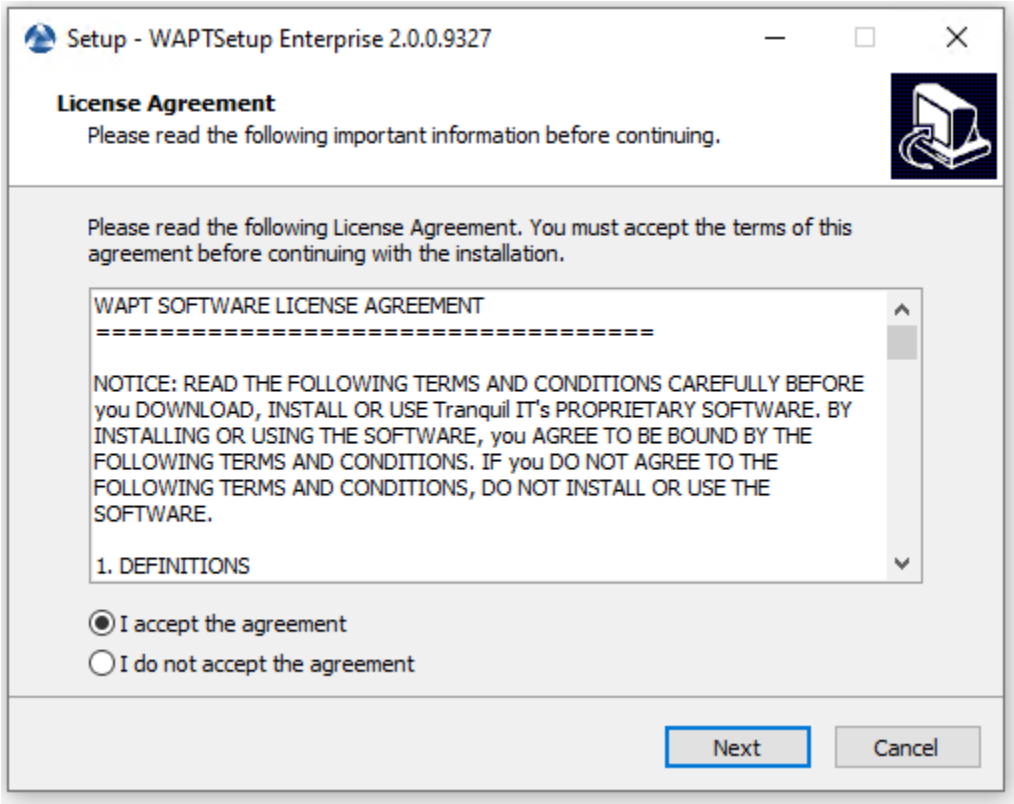


Fig. 1 – L'interface du serveur WAPT dans un navigateur web

— Choisissez la langue du programme d'installation de 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).

Tableau 1 – 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*.

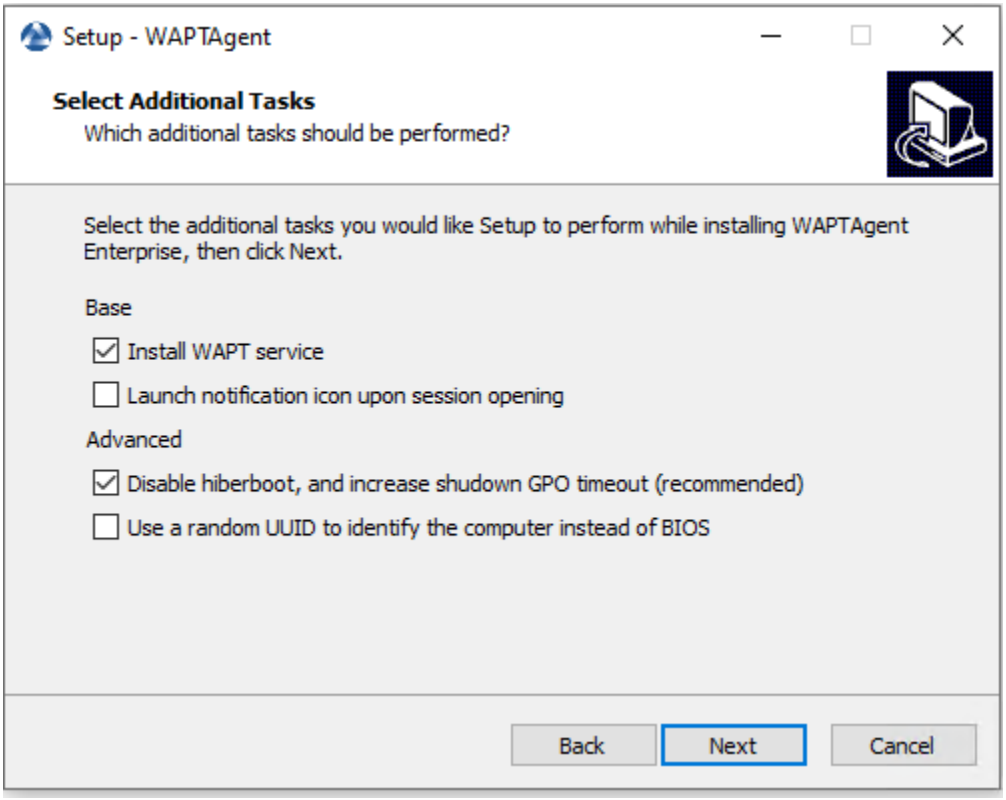


FIG. 2 – Choix des options du programme d'installation pour le déploiement de l'agent WAPT

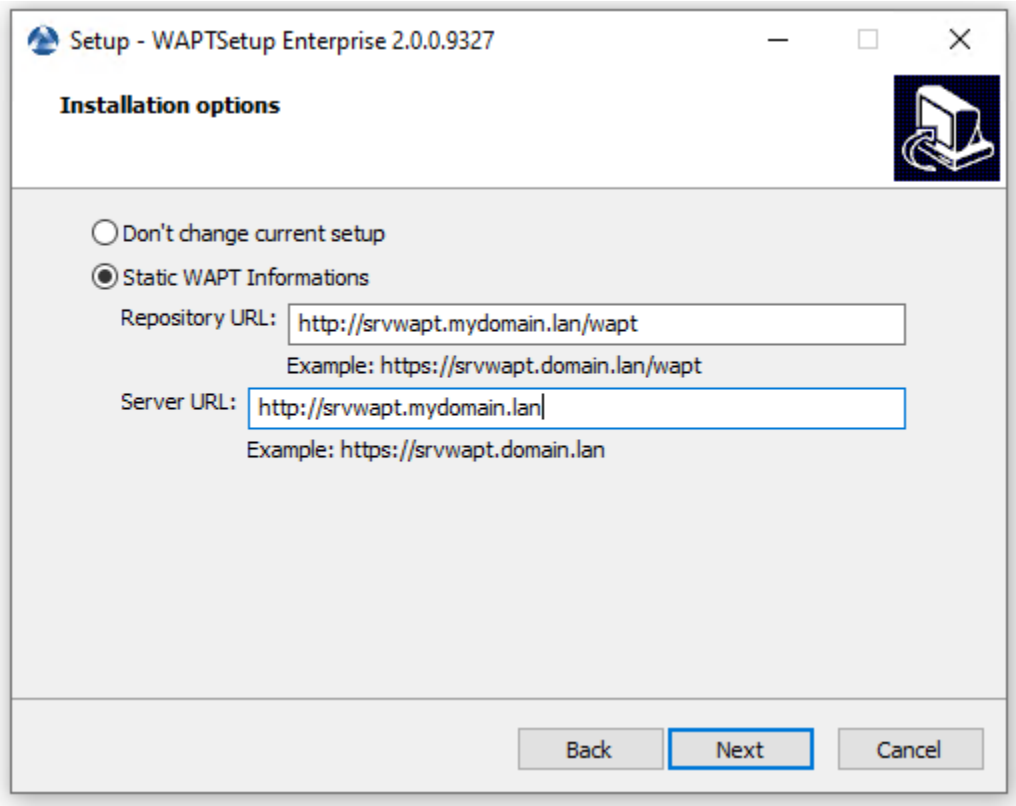
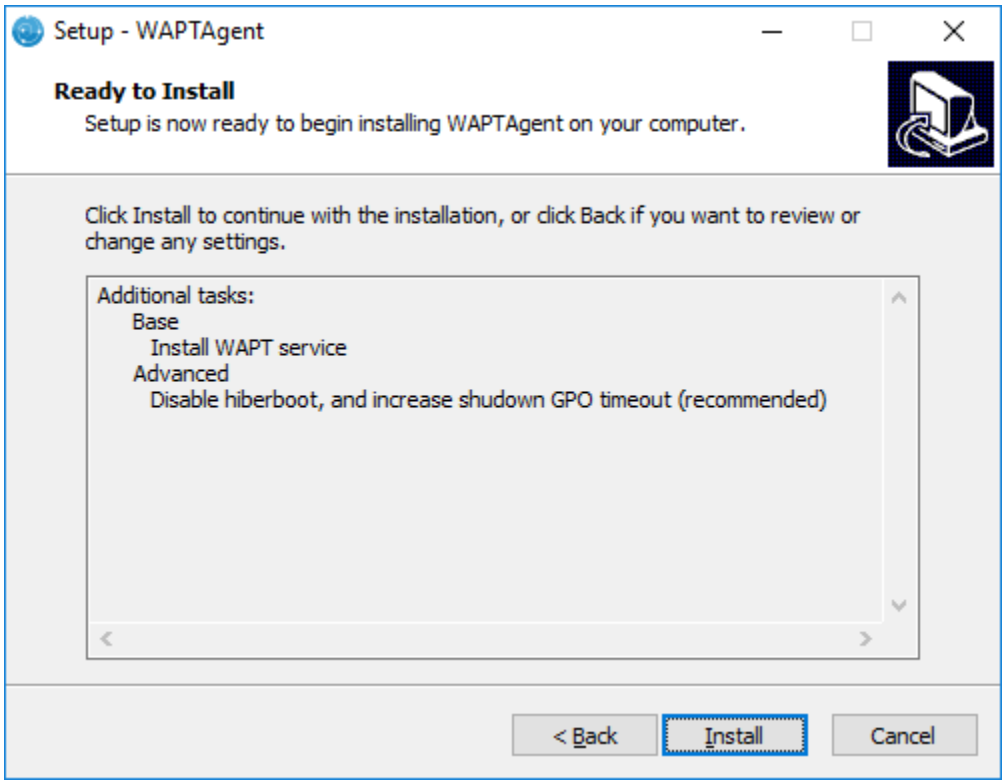
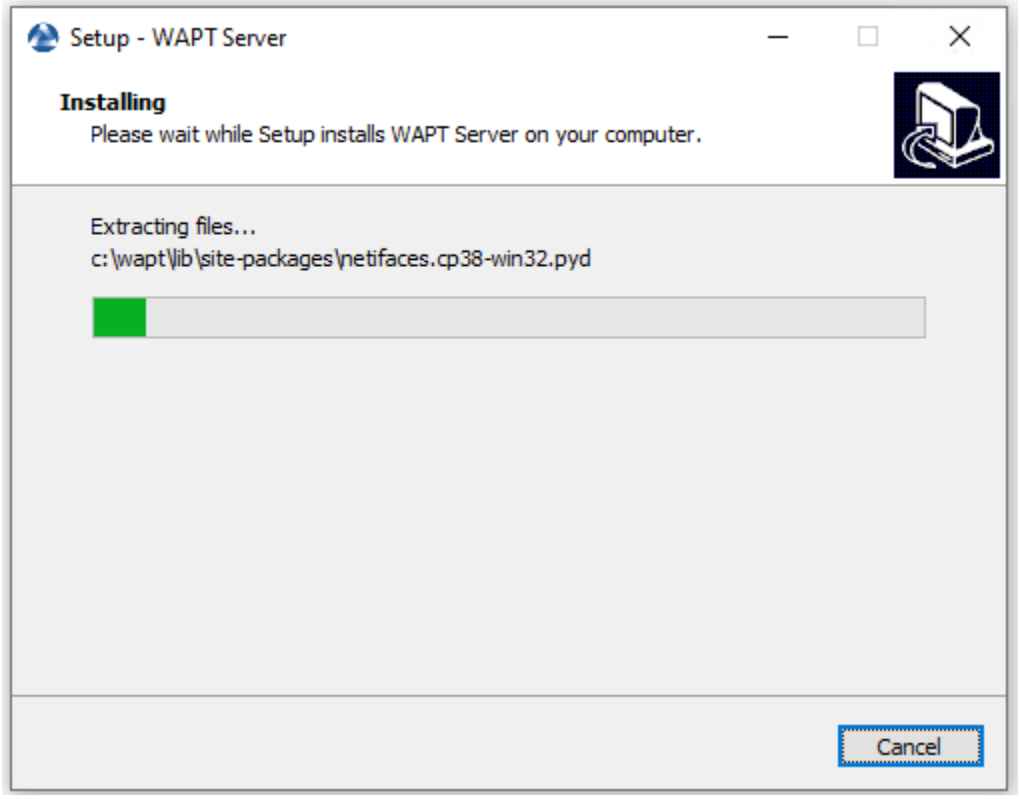


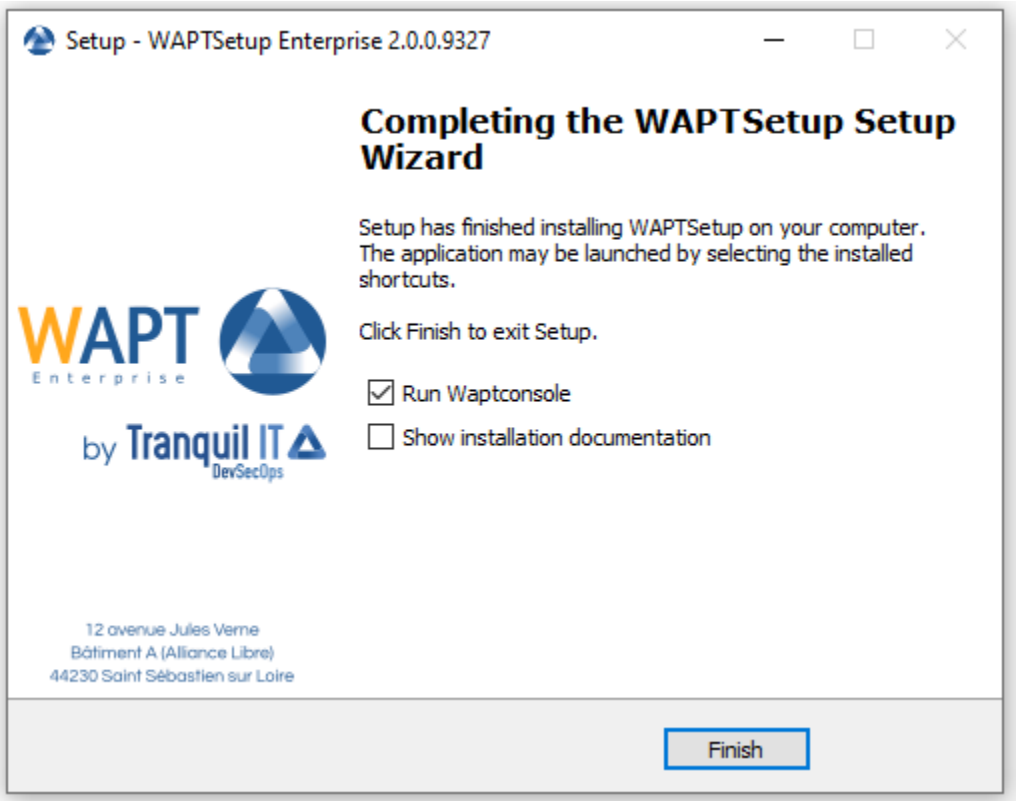
Fig. 3 – Choisir le référentiel WAPT et le serveur WAPT



— 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*.

17.1.2 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.

Tableau 2 – 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.

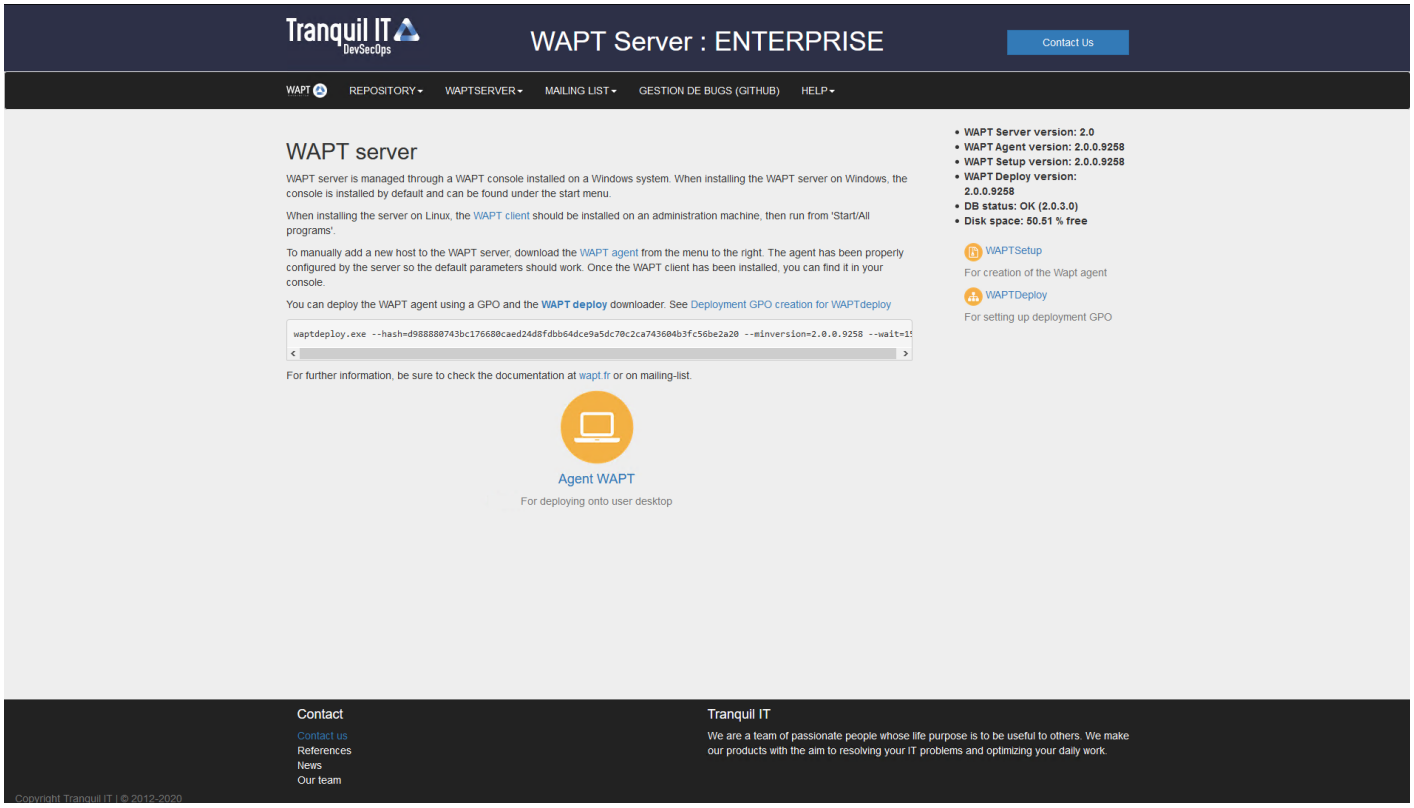


FIG. 4 – L'interface du serveur WAPT dans un navigateur web

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*.

FIG. 5 – Création d'une stratégie de groupe pour déployer l'agent WAPT

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

FIG. 6 – Recherche du fichier de l'utilitaire WAPT Deployment sur votre ordinateur

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

FIG. 7 – 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

---

FIG. 8 – Sélection du script de l'utilitaire WAPT Deployment

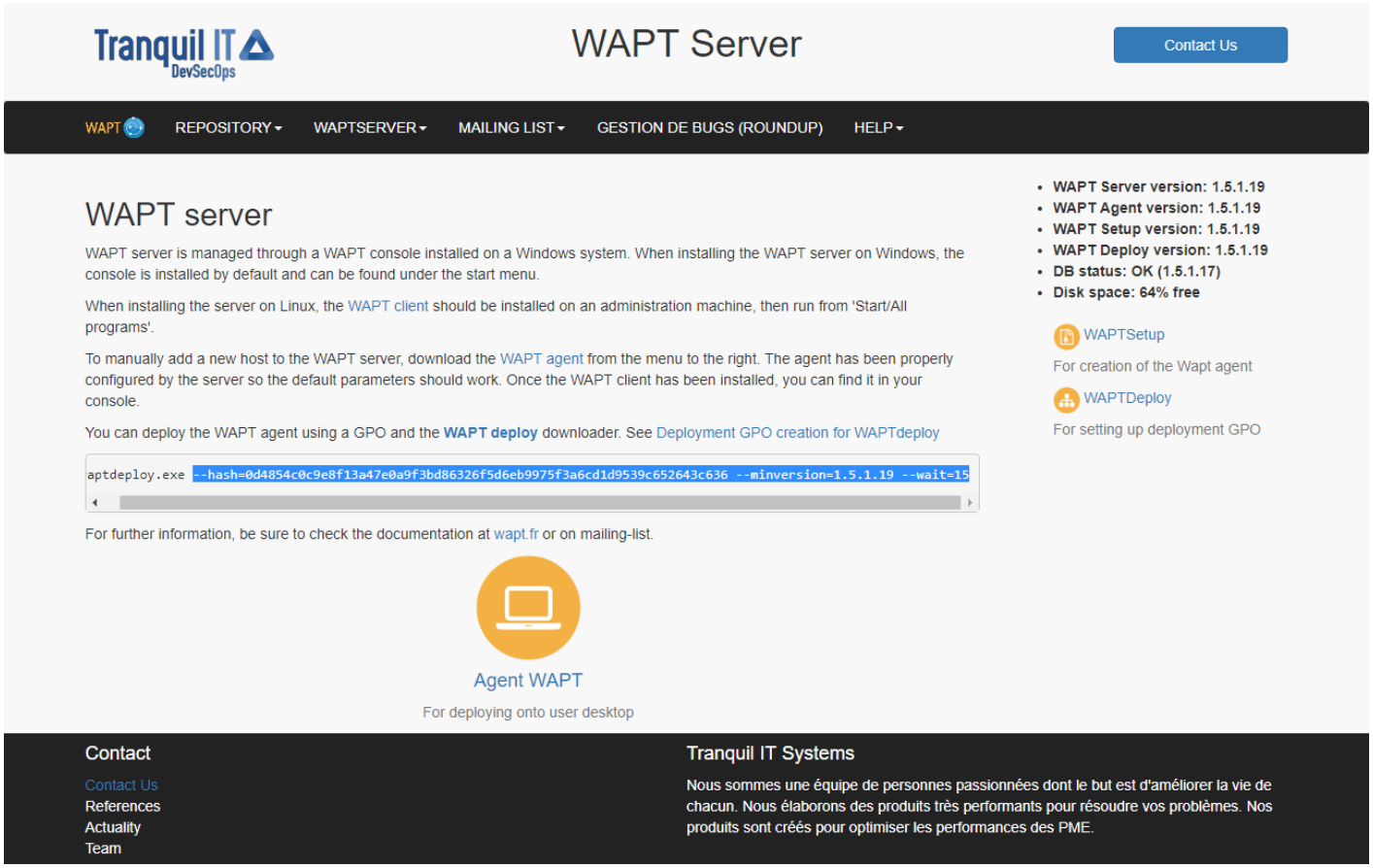


FIG. 9 – Console web du serveur WAPT

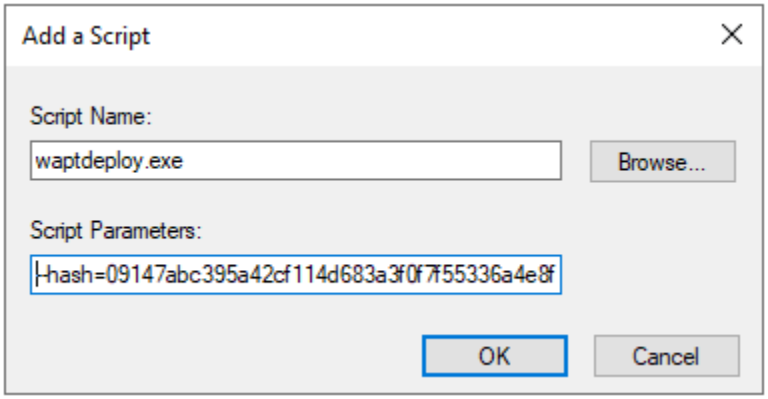


FIG. 10 – Ajout du script de l'utilitaire de déploiement WAPT à la GPO de démarrage

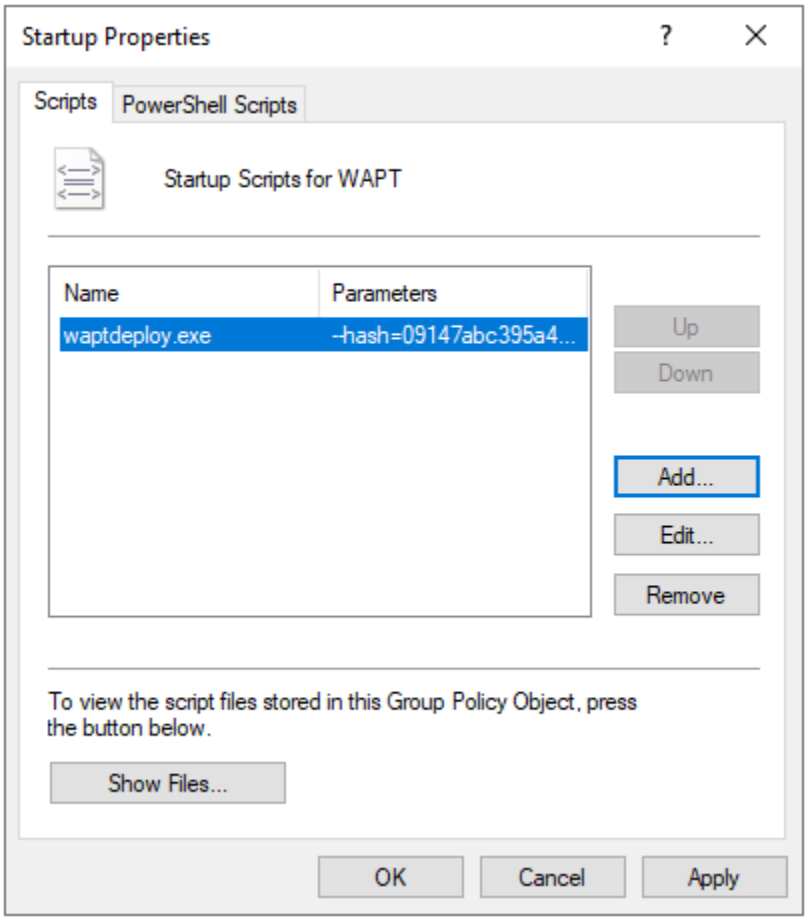


FIG. 11 – La GPO de l'utilitaire de déploiement de WAPT à déployer au prochain démarrage

Tableau 3 – 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 l'installateur **`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).

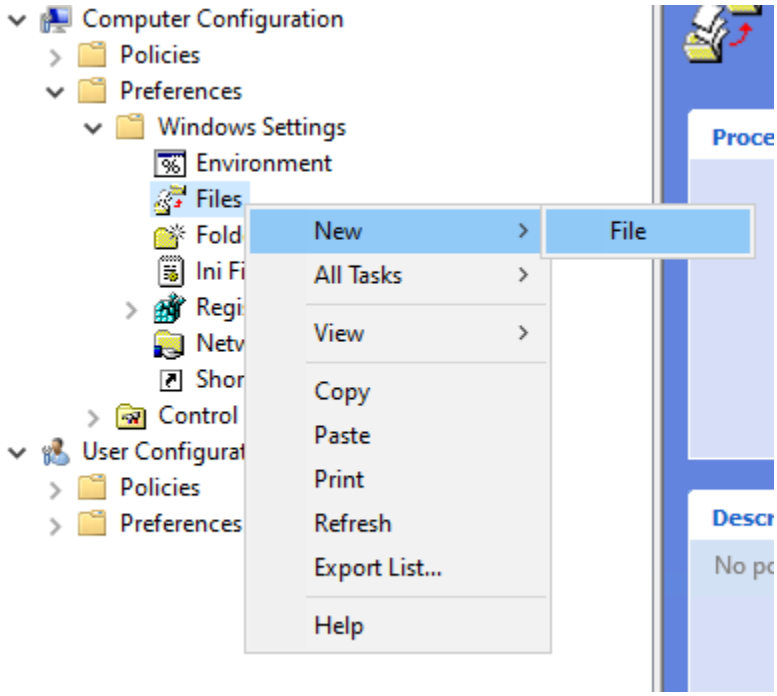
Avec une tâche planifiée

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.

TABLEAU 4 – 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`**.

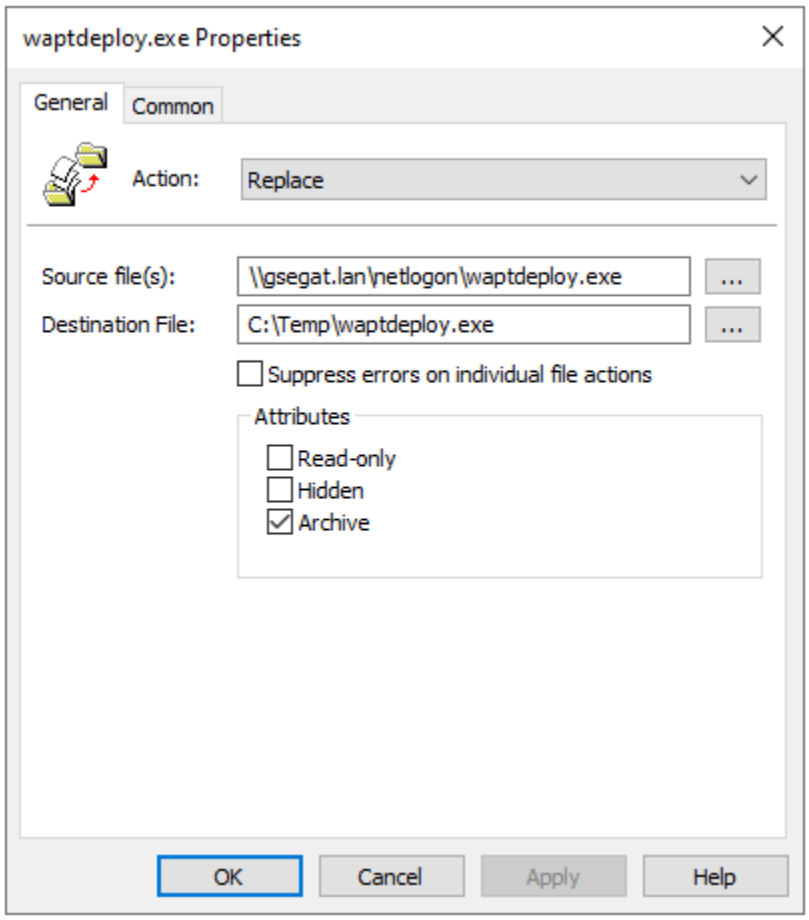
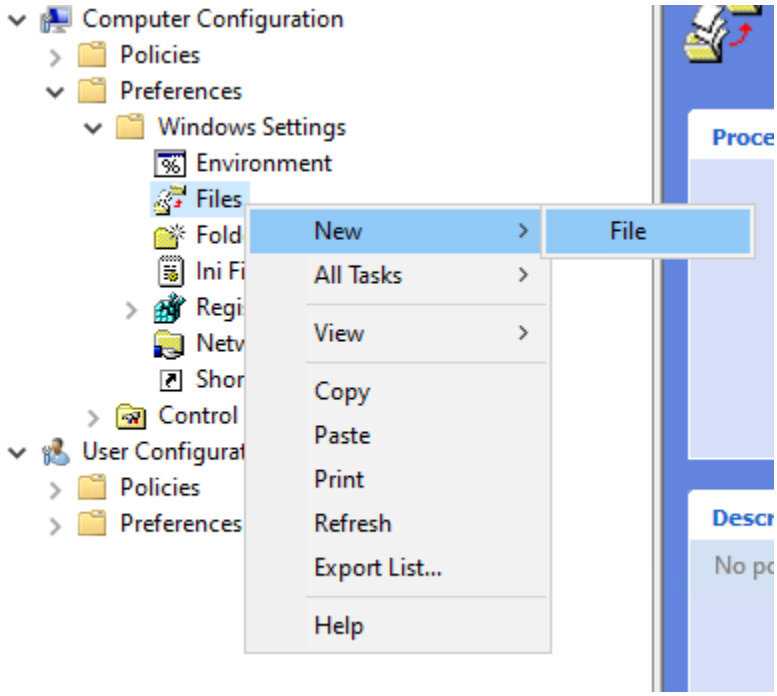


FIG. 12 – Progression de l'installation de l'agent WAPT



— Définir les paramètres.

TABLEAU 5 – 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*.
— Vérifiez que *Enabled* est coché, puis allez dans l'onglet *Actions*.

— Créez une nouvelle action *Démarrer un programme* pour `waptdeploy.exe`.

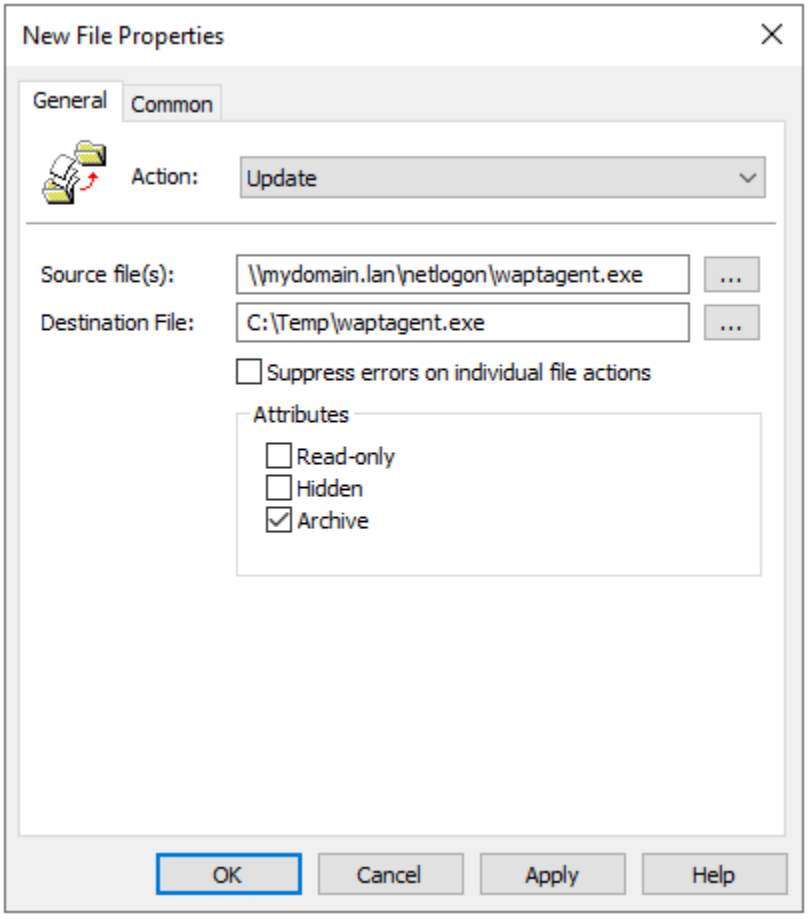


FIG. 13 – Préparation de la GPO de mise à jour WAPT

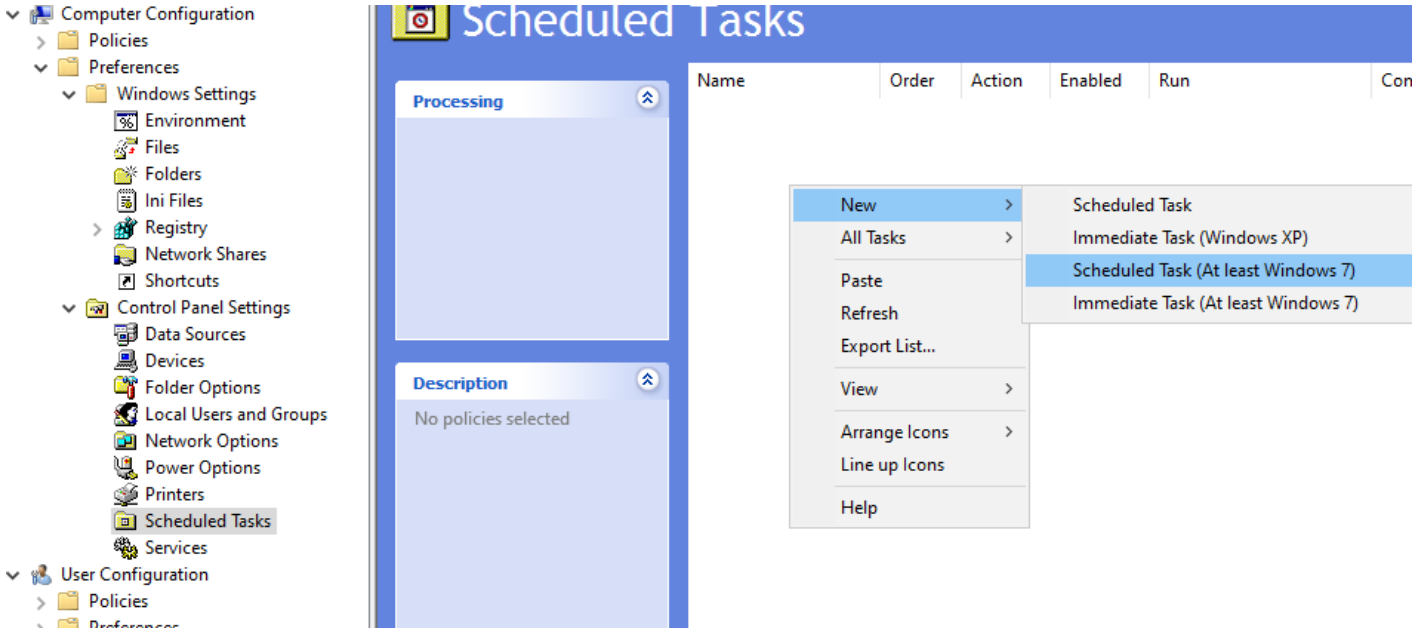


FIG. 14 – Créez la tâche planifiée pour la fenêtre des propriétés de l'utilitaire de déploiement WAPT dans RSAT

TABLEAU 6 – 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.

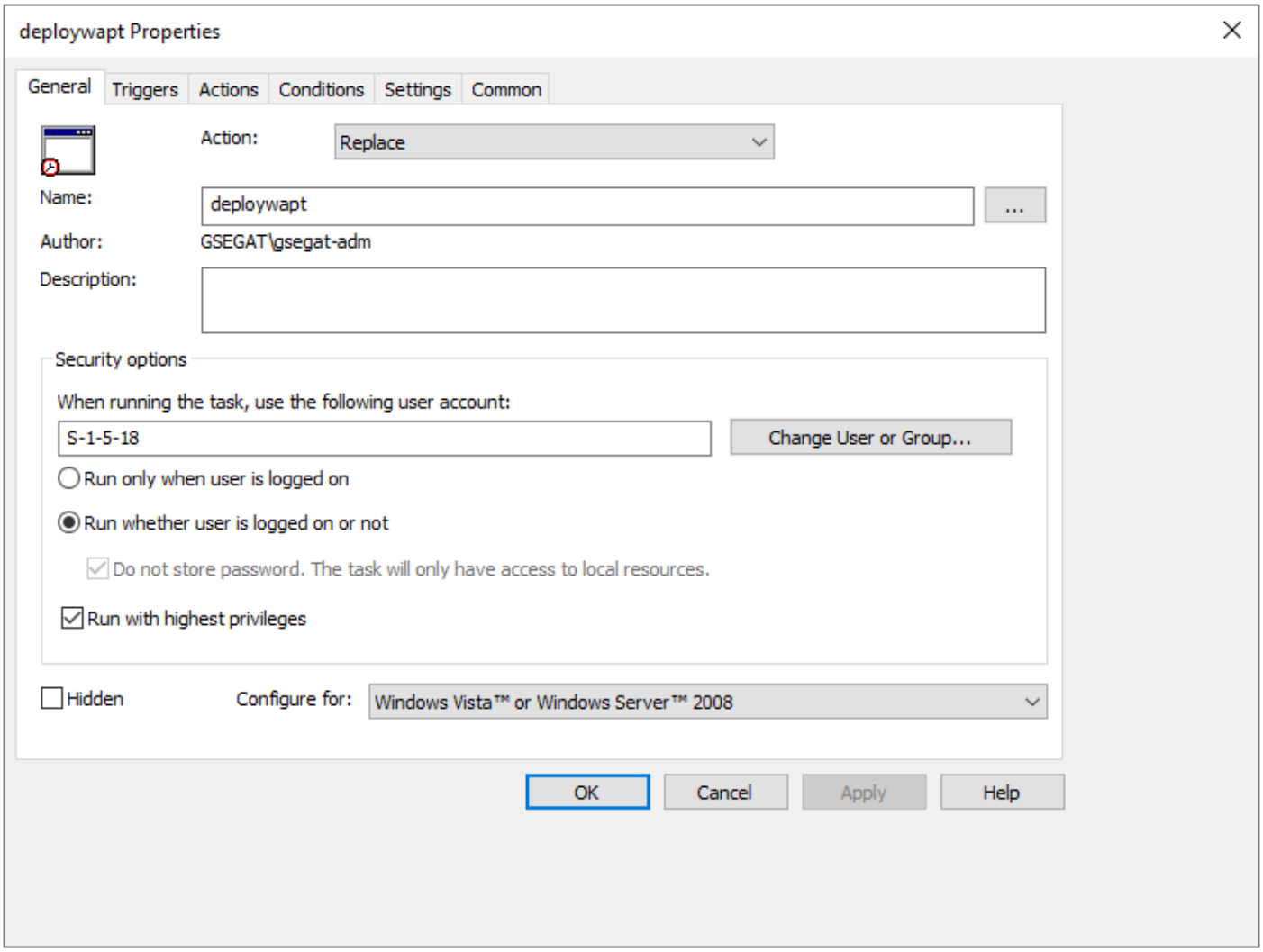


Fig. 15 – Onglet Général de la fenêtre Propriétés dans RSAT

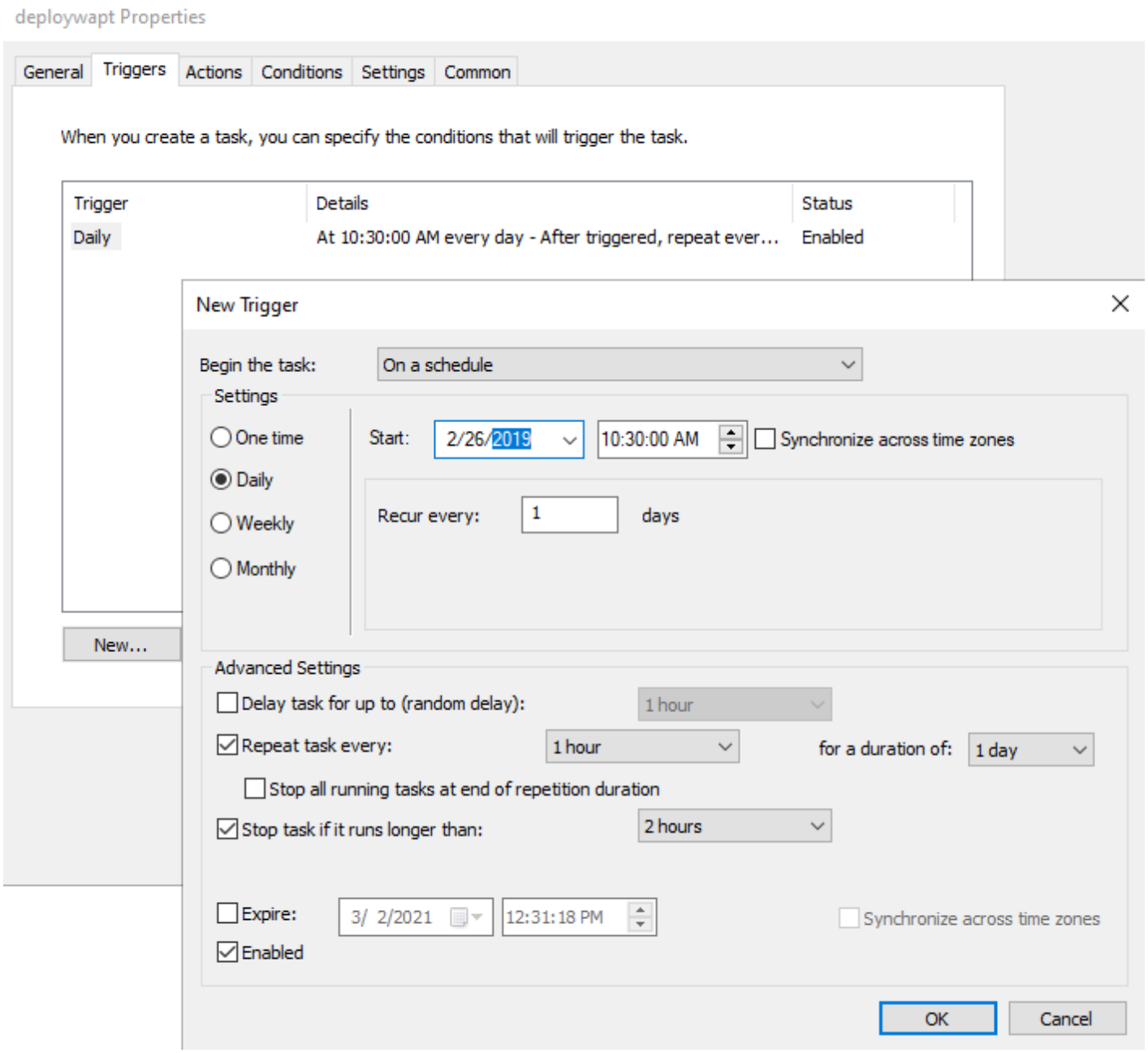


Fig. 16 – Onglet Déclencheur dans la fenêtre Propriétés dans RSAT

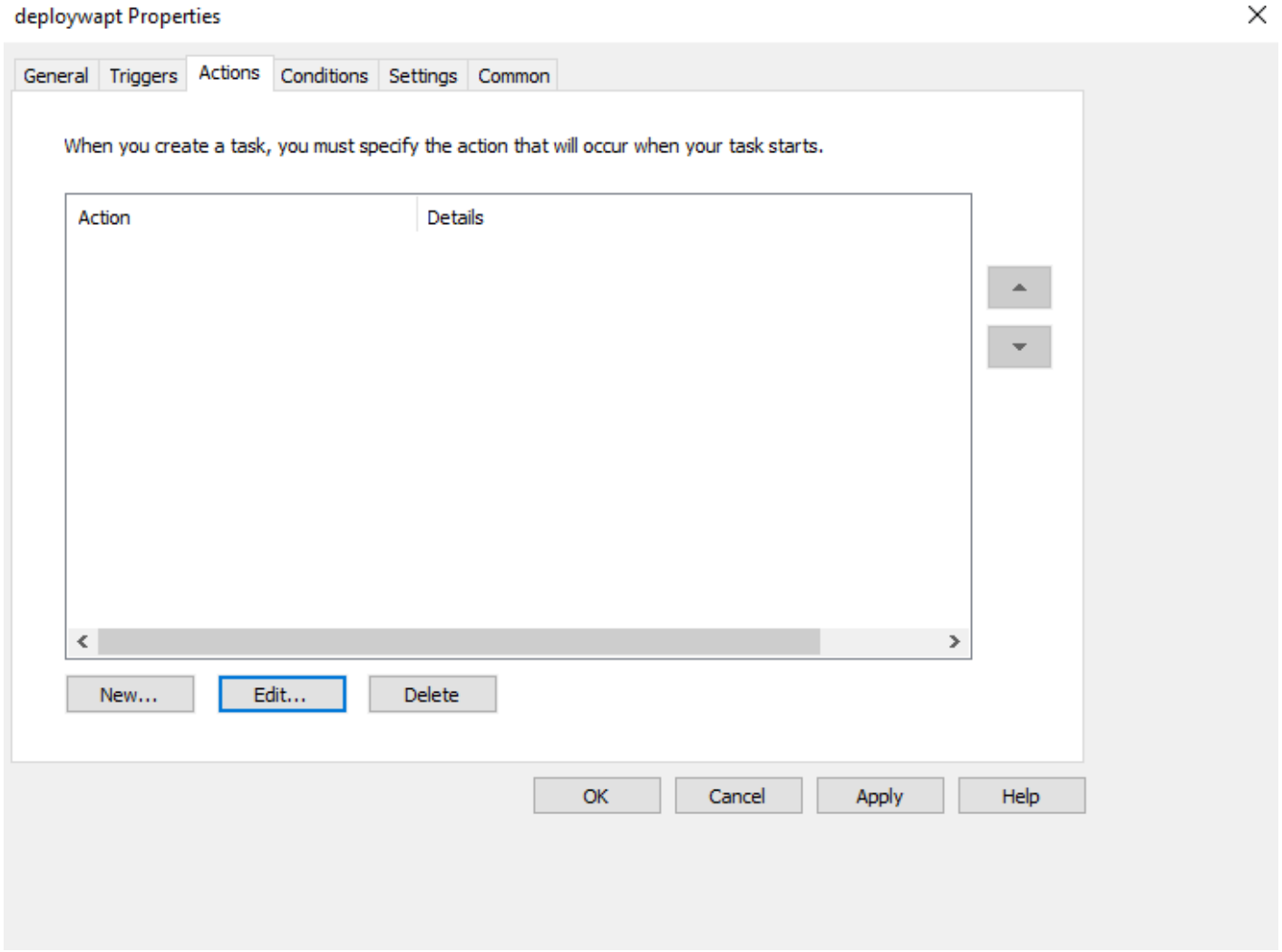


FIG. 17 – Onglet Actions dans la fenêtre Propriétés de RSAT

FIG. 18 – Onglet Actions dans la fenêtre Propriétés de RSAT

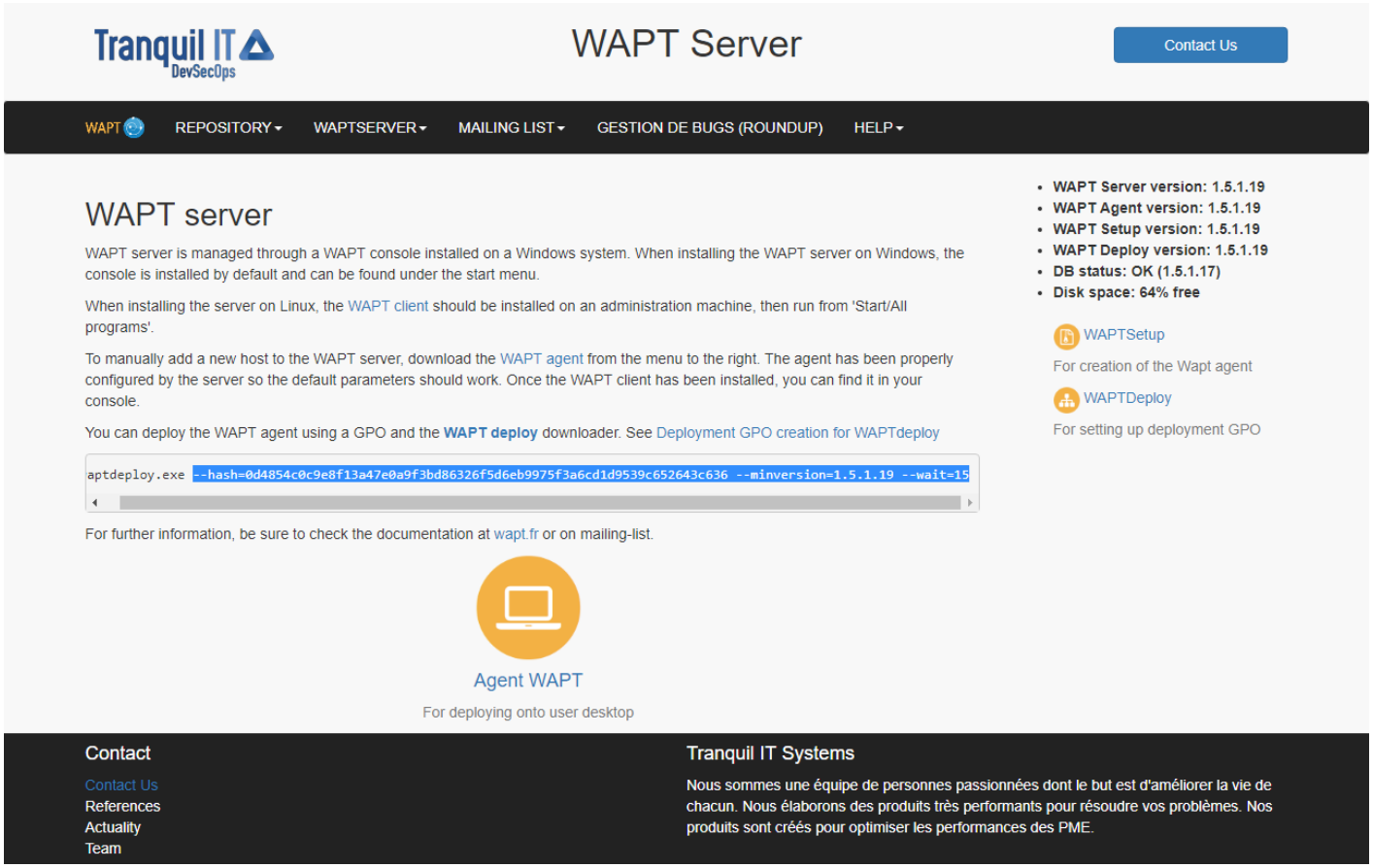


FIG. 19 – Console web du serveur WAPT

— 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
```

Tableau 7 – 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= 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.

17.2 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.

17.2.1 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.

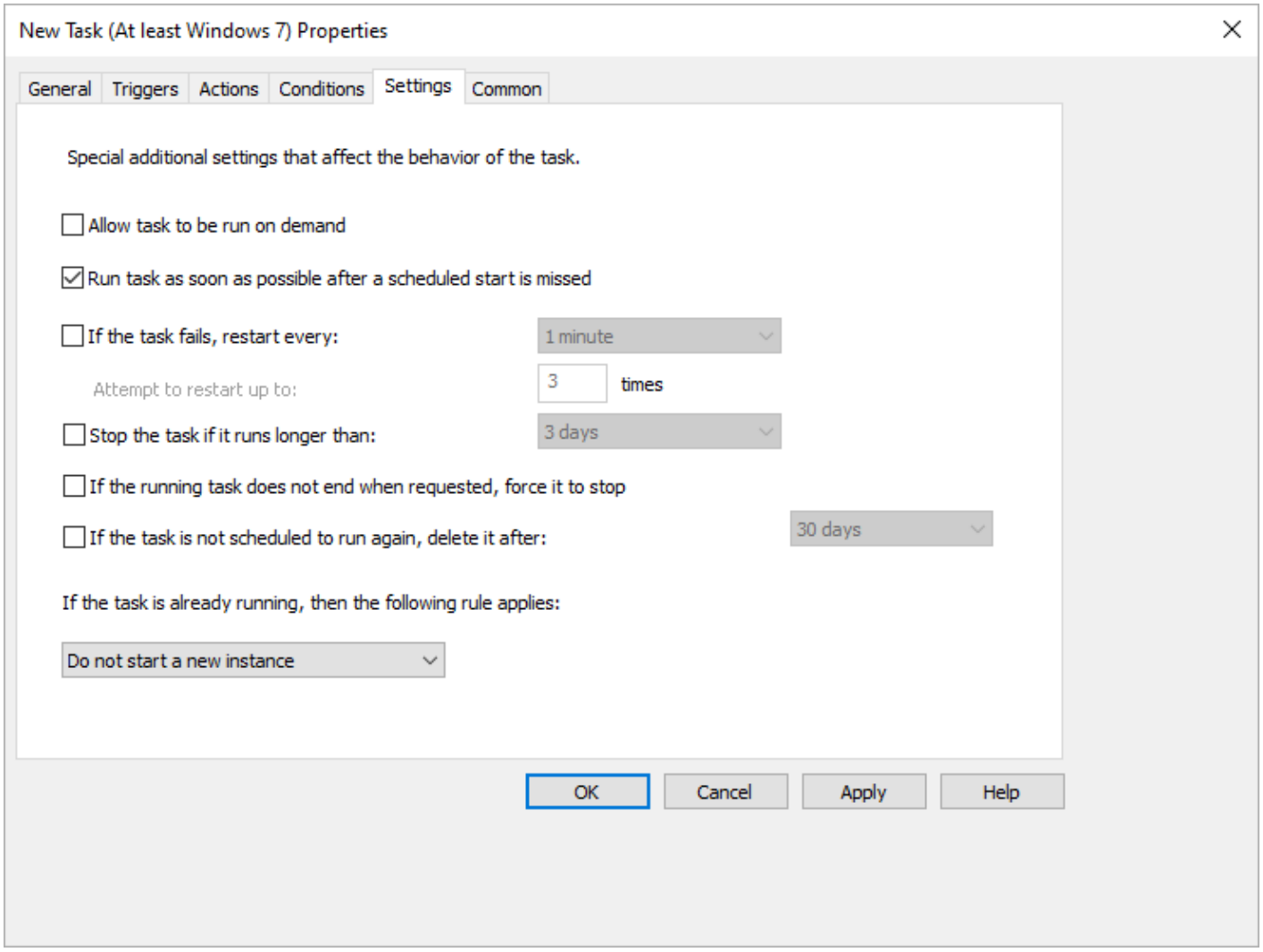


FIG. 20 – Onglet Paramètres dans la fenêtre Propriétés de RSAT

17.2.2 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.

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

- d'abord le packaging copie le nouveau fichier `waptagent.exe` sur l'ordinateur client et crée une nouvelle tâche planifiée qui exécutera **`waptagent.exe`** avec des drapeaux d'installation prédéfinis deux minutes après la création de la tâche planifiée. À ce moment-là, le packaging lui-même est installé et l'inventaire sur le serveur WAPT indique que l'installation du package 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 **`waptagent.exe`**. **`waptagent.exe`** arrête le service WAPT local, met à niveau l'installation WAPT locale, puis redémarre le service. La tâche planifiée est alors automatiquement supprimée et l'agent WAPT renvoie son inventaire au serveur WAPT. Maintenant 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.

CHAPITRE 18

Gestion de l'agent WAPT sous Linux et MacOS

18.1 Déploiement de l'agent WAPT sur Linux et MacOS

Le processus dépend de votre système d'exploitation :

Distributions basées sur Debian / Ubuntu

Indication : L'agent WAPT pour Debian a été testé sur Debian 8, 9, 10 et 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-2.2/ $(lsb_release -cs) main" > /
↪etc/apt/sources.list.d/wapt.list

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

RedHat 7 based distributions

Indication : The WAPT Agent for Redhat based system has been tested on Redhat 7 and derivatives.

— Mettre à jour la distribution sous-jacente.

```
yum update
```

— Récupérez la clé .gpg et configurez le référentiel WAPT.

```
wget -q -O /tmp/tranquil_it.gpg "https://wapt.tranquil.it/redhat7/RPM-GPG-KEY-TISWAPT-7"; rpm --
↪import /tmp/tranquil_it.gpg

cat > /etc/yum.repos.d/wapt.repo <<EOF
[wapt]
name = WAPT Server Repo
baseurl = https://wapt.tranquil.it/redhat7/wapt-2.2/
enabled = True
gpgcheck = True
EOF
```

— installer l'agent WAPT en utilisant yum :

```
yum install tis-waptagent
```

RedHat 8 based distributions

Indication : The WAPT Agent for Redhat based system has been tested on Redhat 8 and derivatives.

— Mettre à jour la distribution sous-jacente.

```
yum update
```

— Récupérez la clé .gpg et configurez le référentiel WAPT.

```
wget -q -O /tmp/tranquil_it.gpg "https://wapt.tranquil.it/redhat8/RPM-GPG-KEY-TISWAPT-8"; rpm --
↪import /tmp/tranquil_it.gpg

cat > /etc/yum.repos.d/wapt.repo <<EOF
[wapt]
name = WAPT Server Repo
baseurl = https://wapt.tranquil.it/redhat8/wapt-2.2/
enabled = True
gpgcheck = True
EOF
```

— installer l'agent WAPT en utilisant yum :

```
yum install tis-waptagent
```

MacOS

Indication : L'agent WAPT n'a été testé que sur l'architecture **Intel** (les processeurs Apple Silicon M1 seront bientôt pris en charge) :
- High Sierra (10.13) ;
- Mojave (10.14) ;
- Catalina (10.15) ;
- Big Sur (11.x) ;
- Monterey (12.x).

- Téléchargez et installez l'agent WAPT (note : la chaîne de hachage peut changer, pour obtenir la dernière version, pointez votre navigateur sur l'url https://wapt.tranquil.it/wapt/releases/wapt-2.2/) :

```
curl -o tis-waptagent-2.2.0.11586-macos-9c22a4fb.pkg   https://wapt.tranquil.it/wapt/releases/wapt-
↪2.2/tis-waptagent-2.2.0.11586-macos-9c22a4fb.pkg
sudo installer -target / -pkg tis-waptagent*.pkg
```

18.1.1 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
```

18.1.2 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.

18.1.3 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.

Enregistrement de

— Enfin, exécutez la commande suivante pour enregistrer votre hôte Linux avec le serveur WAPT.

```
sudo wapt-get register
```

Redémarrer l'agent

— 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

La console WAPT n'est pas actuellement disponible sur linux.
— L'installation des mises à jour à l'arrêt.
— la console WAPT ;
— Toutes les fonctions spécifiques a Windows.

Particularités de la fonctionnalité du domaine

Les tests ont été effectués avec **sssd** sur un contrôleur de domaine Samba et Windows Active Directory et une authentification kerberos.
— Pour intégrer une machine dans le domaine Active Directory, vous pouvez choisir de 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 l'hôte est membre.

Attention : Nous avons remarqué que la requête LDAP de kerberos ne fonctionne pas si le reverse DNS record n'est pas configuré correctement pour vos contrôleurs de domaine. Ces enregistrements doivent donc être créés s'ils n'existent pas.

18.2 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.

18.2.1 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.

CHAPITRE 19

Désinstallation de l'agent WAPT des clients

19.1 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
```

TABLEAU 1 – 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)
```

19.1.1 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()
```

19.2 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
```

19.3 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
```

Indication : 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 :

Tableau 2 – Possibilités de mise à niveau WAPT disponibles

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

Note : Pour toutes les anciennes versions, veuillez vous référer à la documentation 2.1

Avertissement : Sur WAPT 2.1 *l'activation de la licence* est modifiée.

CHAPITRE 20

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

WAPT Community n'est plus pris en charge par la version WAPT 2.2. Si vous voulez passer de WAPT 1.8.2 Community à WAPT 2.2 Discovery, vous pouvez le faire. Veuillez noter que WAPT Discovery est limité à 300 ordinateurs clients maximum.

Pour mettre à niveau WAPT Community vers WAPT Enterprise, suivez la documentation standard 1.8.2 to 2.2 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, elle se rabattra sur l'édition Discovery. Si vous avez plus de 300 ordinateurs clients dans votre serveur en mode Découverte, la console sera limitée et vous devrez supprimer des entrées d'ordinateurs dans l'inventaire afin de passer sous la limite des 300 ordinateurs.

CHAPITRE 21

Mise à niveau de WAPT de 2.0 ou 2.1 à 2.2

Choisissez votre distribution

Debian / Ubuntu

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

```
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

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

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

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

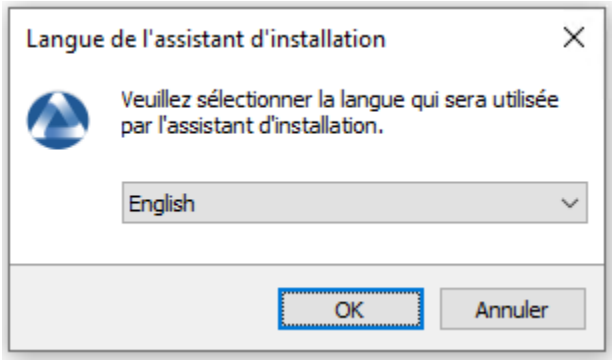
— Une fois terminé, votre serveur est prêt.

Windows

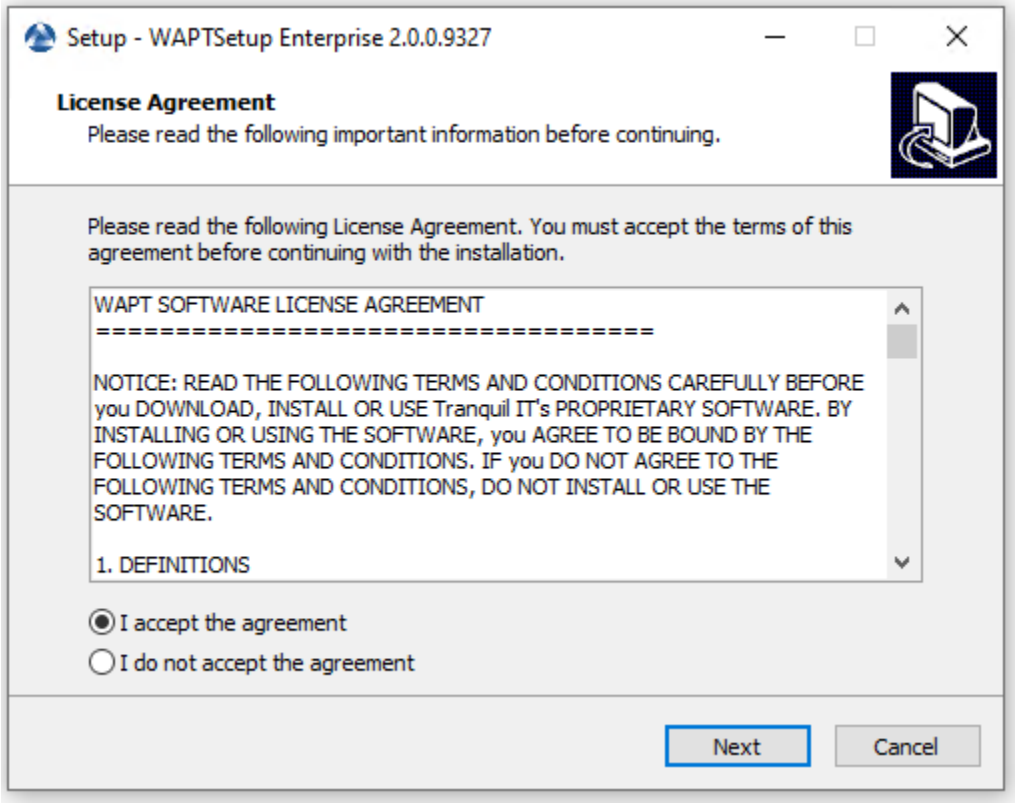
— Téléchargez et exécutez `waptserversetup.exe`.

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

— Choix de la langue pour 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).

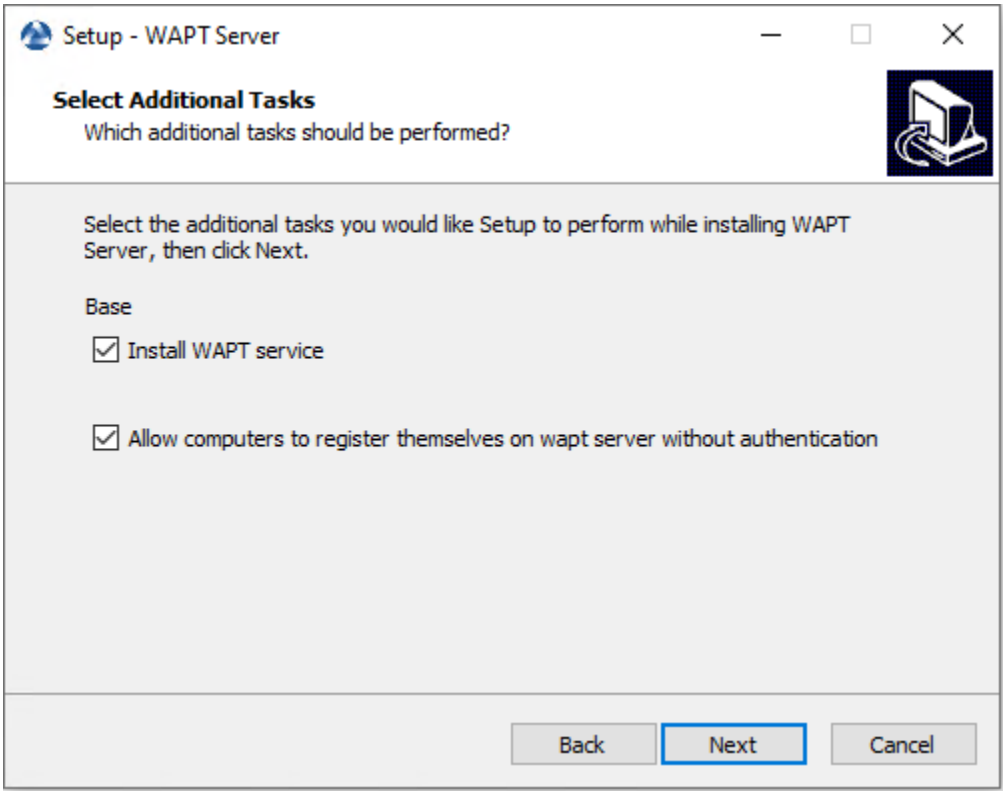
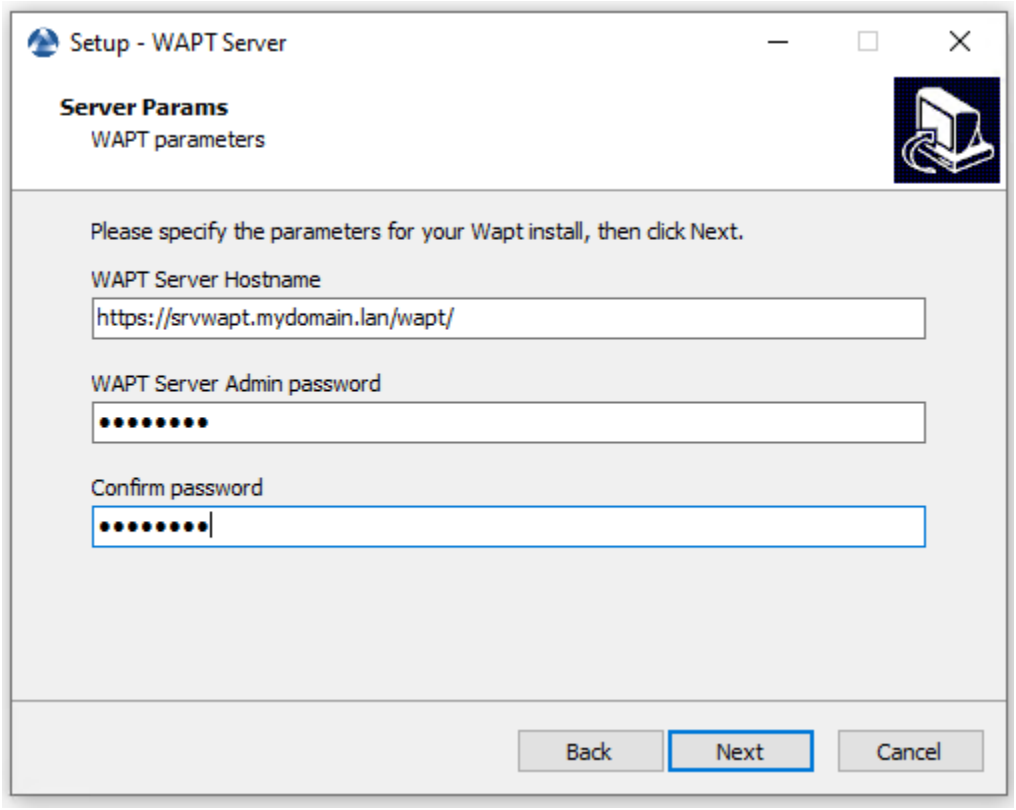
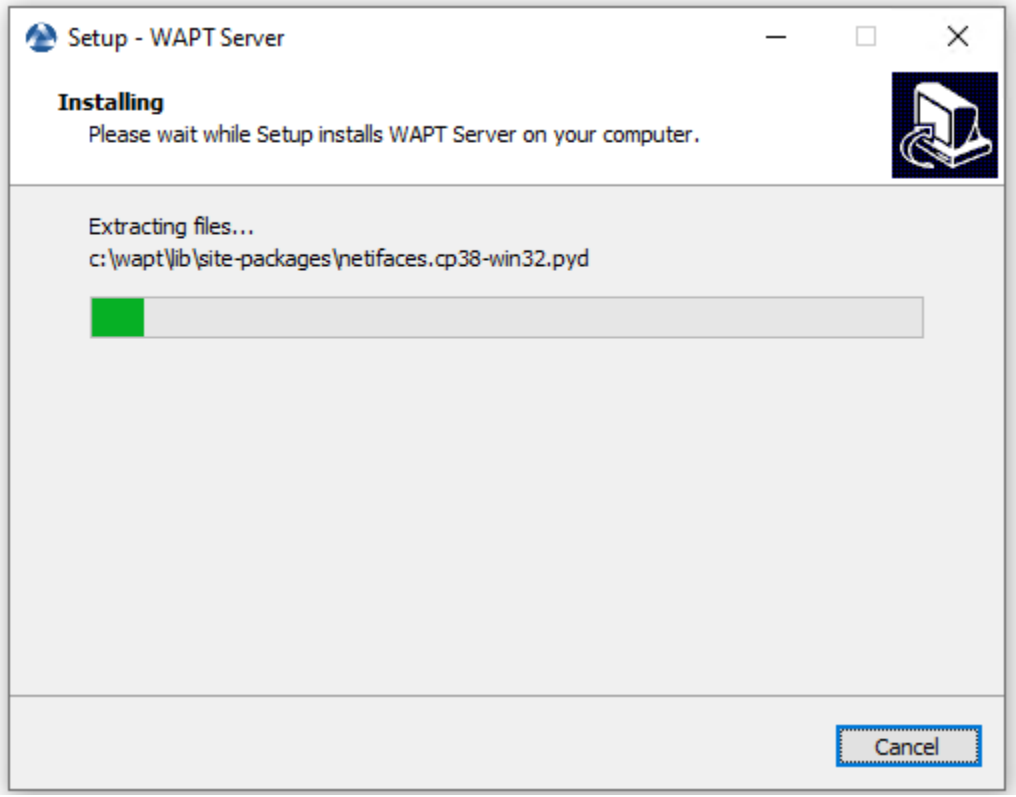


FIG. 1 – Choix des options du programme d'installation pour le déploiement du serveur WAPT



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



— Cliquez sur *Terminer* pour fermer la fenêtre.

Le serveur WAPT sur votre Windows est prêt.

CHAPITRE 22

Mise à niveau de WAPT de 2.0 ou 2.1 à 2.2

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

Choisissez votre distribution

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-2.2/ $(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.

```
cat > /etc/yum.repos.d/wapt.repo <<EOF
[wapt]
name=WAPT Server Repo
baseurl=https://wapt.tranquil.it/centos$(lsb_release -rs | cut -f1 -d.)/wapt-2.2/
enabled=1
gpgcheck=1
EOF

wget -q -O /tmp/tranquil_it.gpg "https://wapt.tranquil.it/centos$(lsb_release -rs | cut -f1 -d.)/
↪RPM-GPG-KEY-TISWAPT-$(lsb_release -rs | cut -f1 -d.)"; rpm --import /tmp/tranquil_it.gpg
```

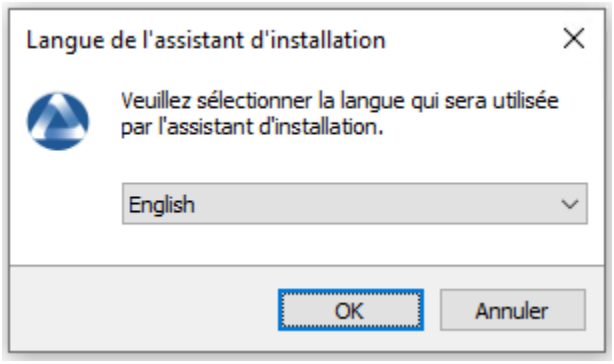
— Et enfin, mettre à jour le serveur WAPT

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

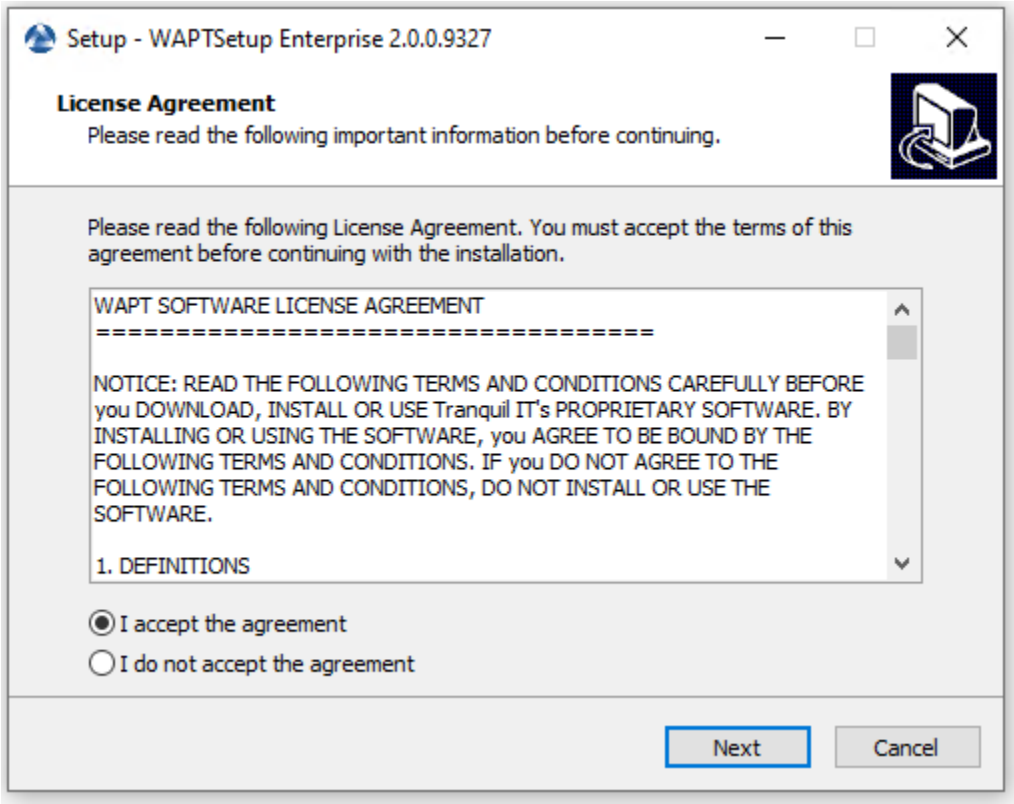
Windows

— Téléchargez et exécutez `waptserversetup.exe`.

— Choix de la langue pour 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.
— Modifiez le mot de passe du serveur WAPT si nécessaire, puis appuyez sur *Suivant*.

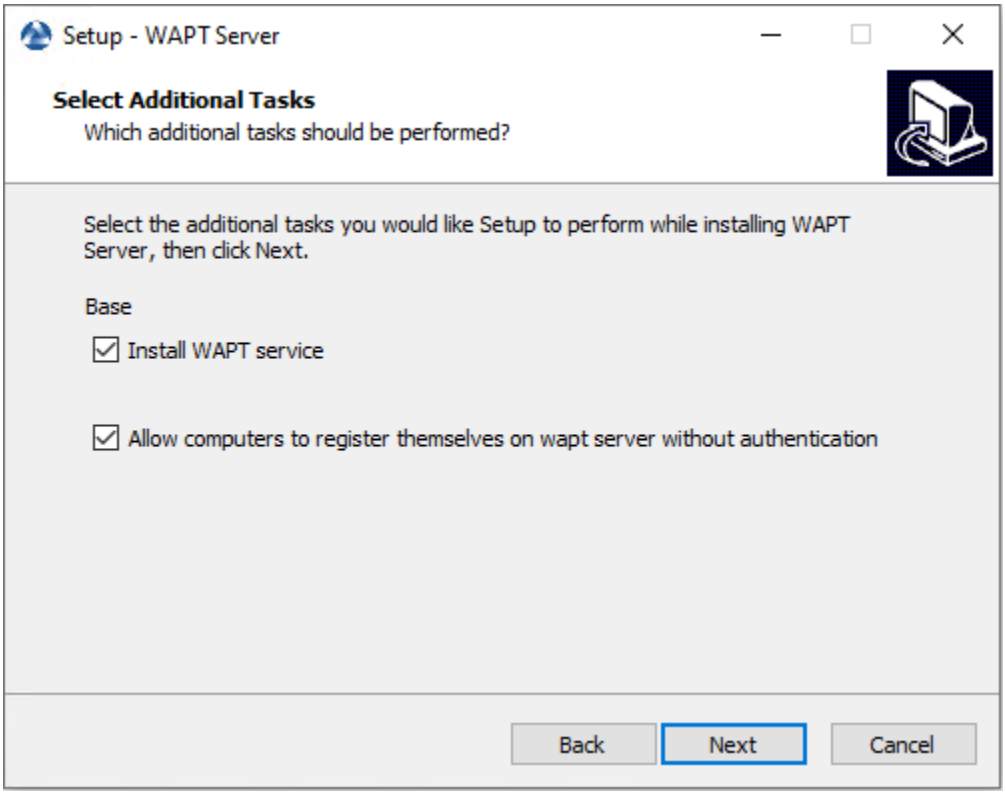
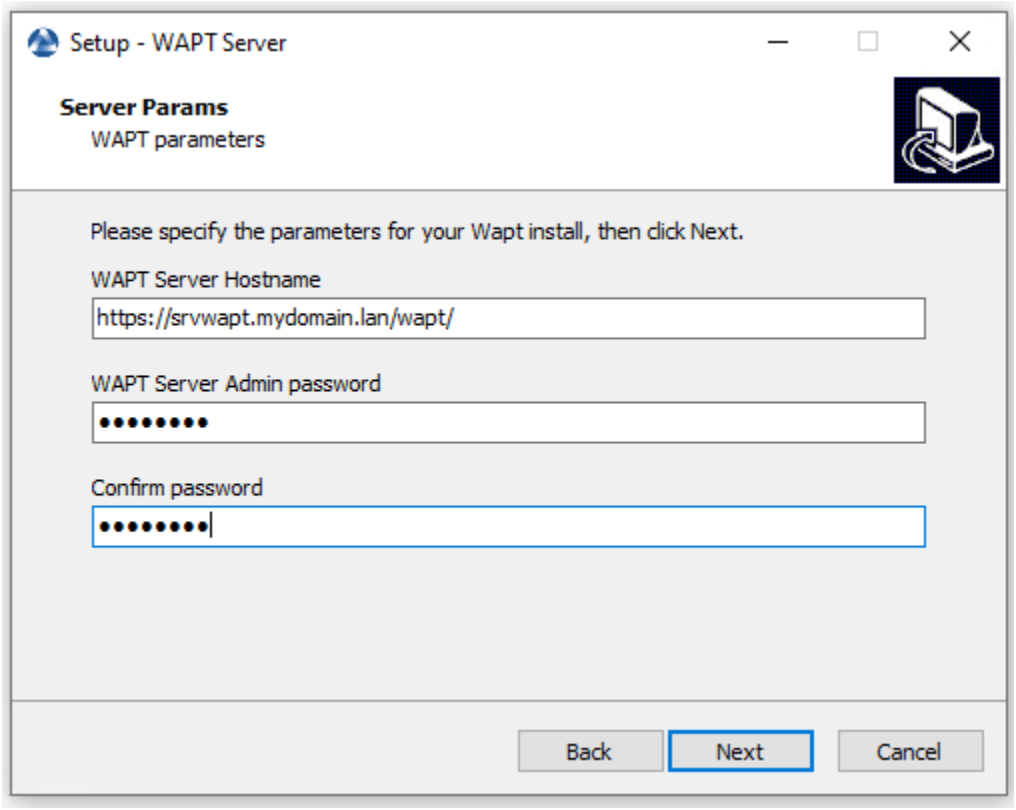
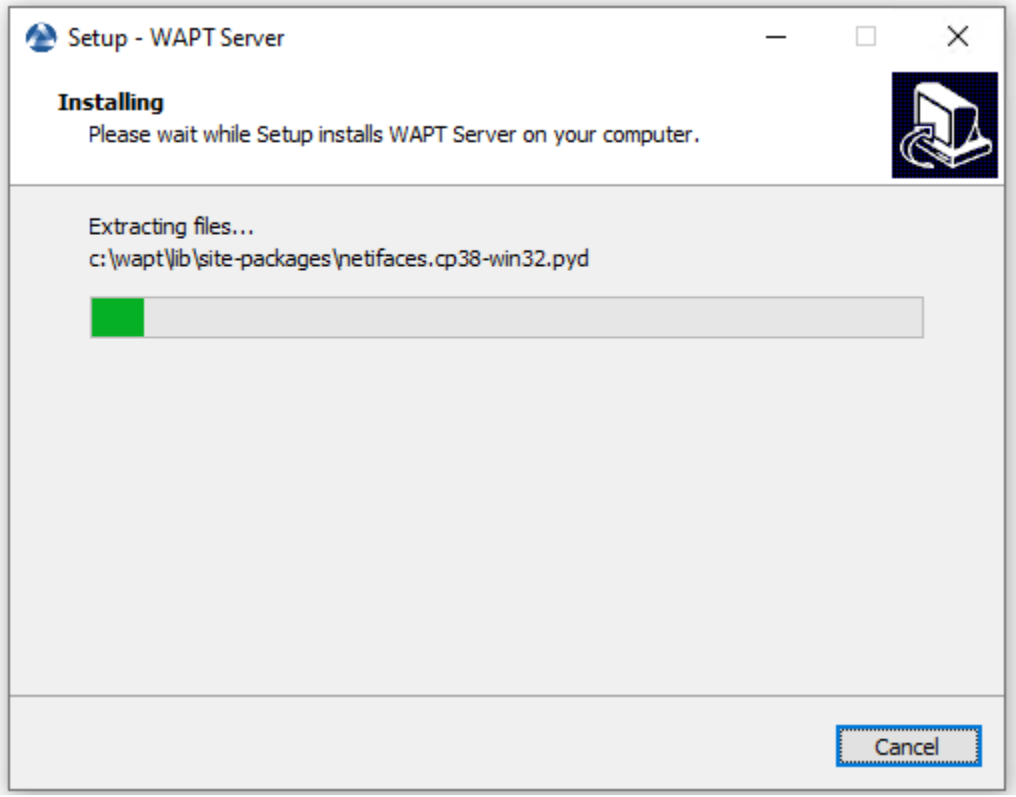


FIG. 1 – Choix des options du programme d'installation pour le déploiement du serveur WAPT



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



— Cliquez sur *Terminer* pour fermer la fenêtre.

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.

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

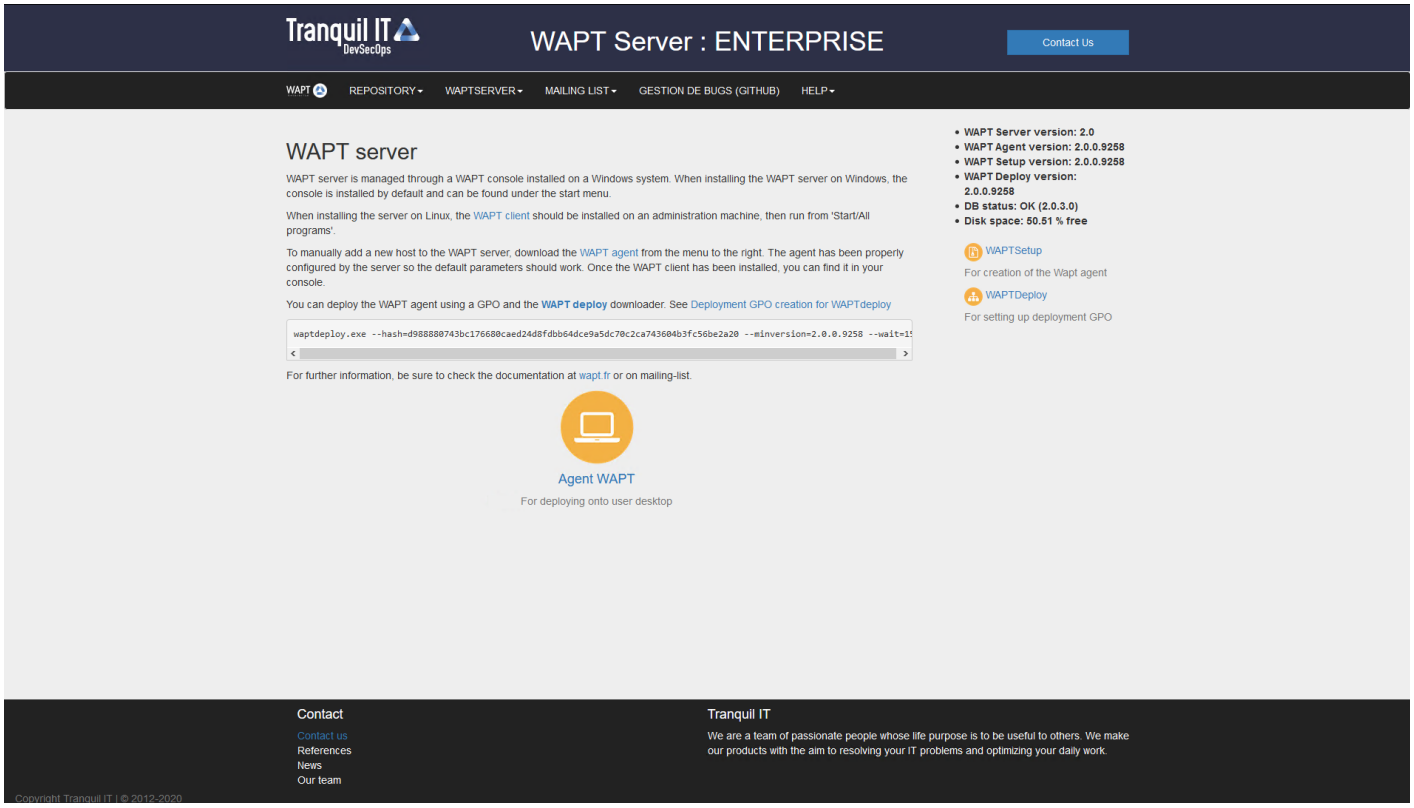


Fig. 2 – L'interface du serveur WAPT dans un navigateur web

CHAPITRE 23

Mise à niveau de WAPT de 1.8.2 à 2.2

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

Avertissement : Sur WAPT 2.1 *l'activation de la licence* est modifiée.

Choisissez votre distribution

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. **WAPTServer 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-2.2/ $(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*
— Re-signez tous vos paquets WAPT.
  — *En graphique*, ou
  — *En ligne de commande*.

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.

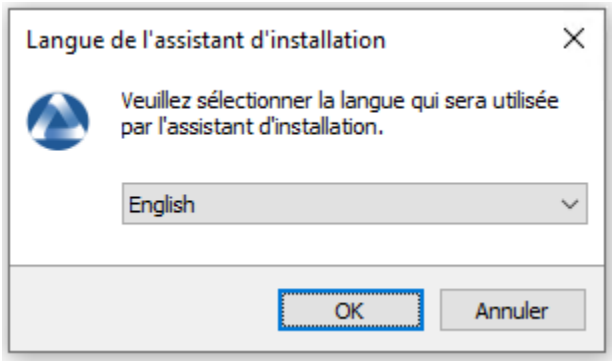
```
cat > /etc/yum.repos.d/wapt.repo <<EOF
[wapt]
name=WAPT Server Repo
baseurl=https://wapt.tranquil.it/centos$(lsb_release -rs | cut -f1 -d.)/wapt-2.2/
enabled=1
gpgcheck=1
EOF

wget -q -O /tmp/tranquil_it.gpg "https://wapt.tranquil.it/centos7/RPM-GPG-KEY-TISWAPT-7"; rpm --
↪import /tmp/tranquil_it.gpg
```

— Et enfin, mettre à jour le serveur WAPT

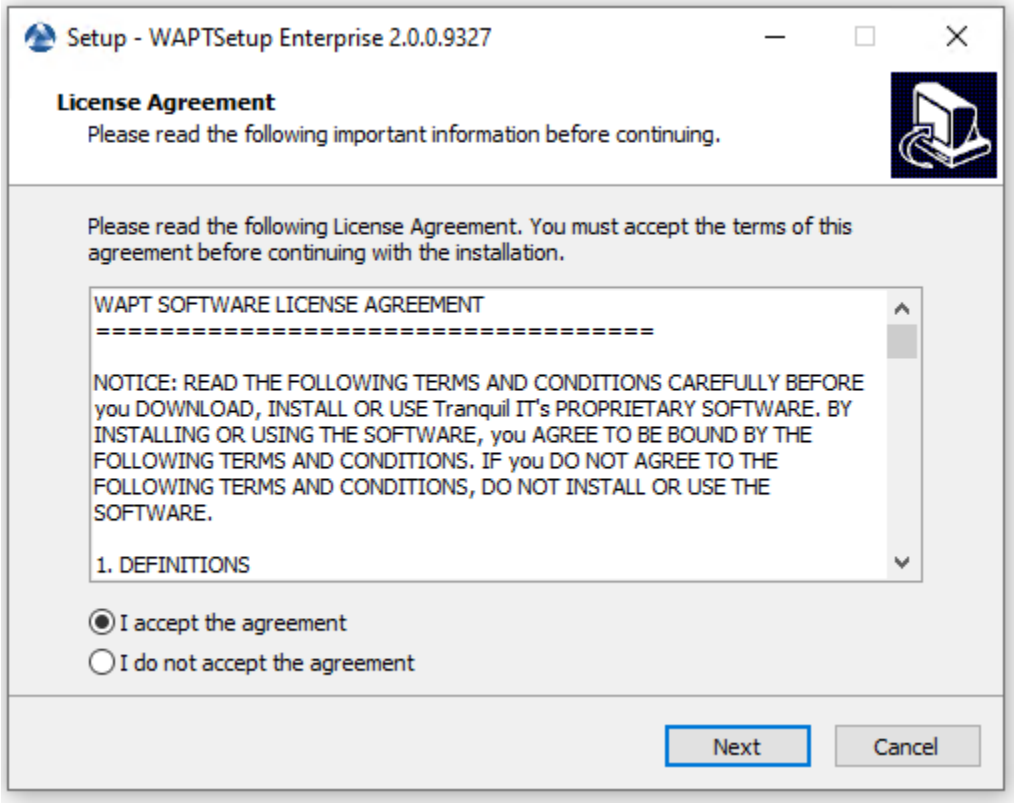
```
yum install tis-waptserver tis-waptsetup cabextract -y
```

— Lancer l'étape de post-configuration *étape de post-configuration*
— Re-signez tous vos paquets WAPT.
  — *En graphique*, ou
  — *En ligne de commande*.

Windows
— Téléchargez et exécutez `waptserversetup.exe`.
— Choix de la langue pour 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.
— Modifiez le mot de passe du serveur WAPT si nécessaire, puis appuyez sur *Suivant*.

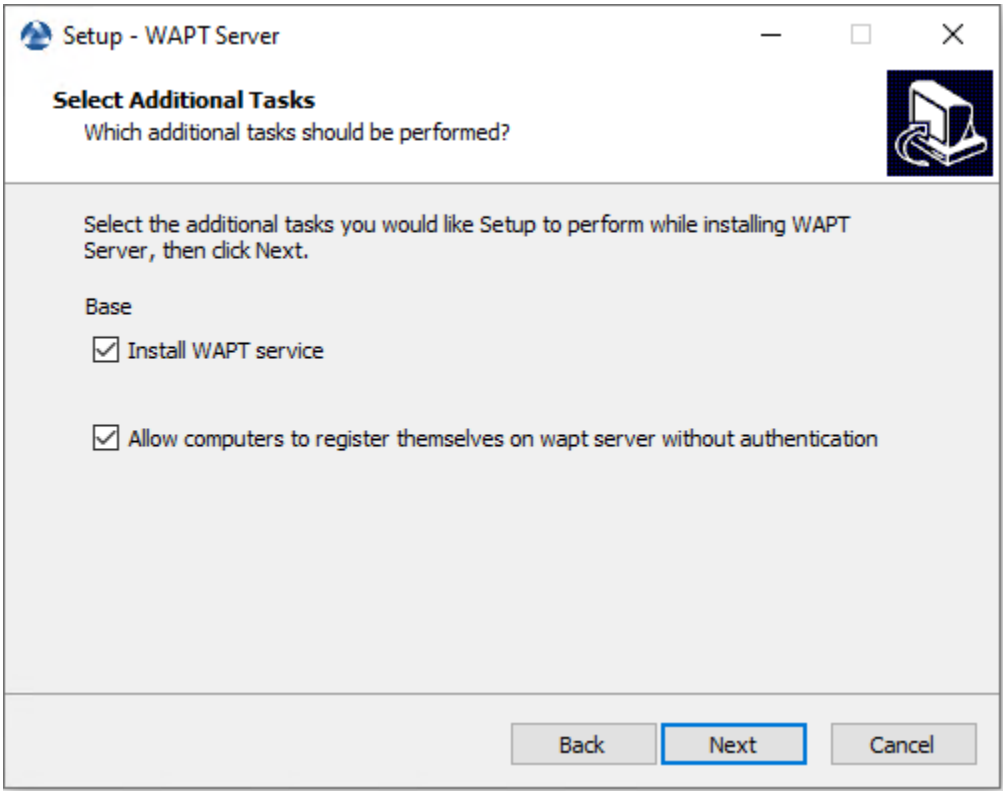
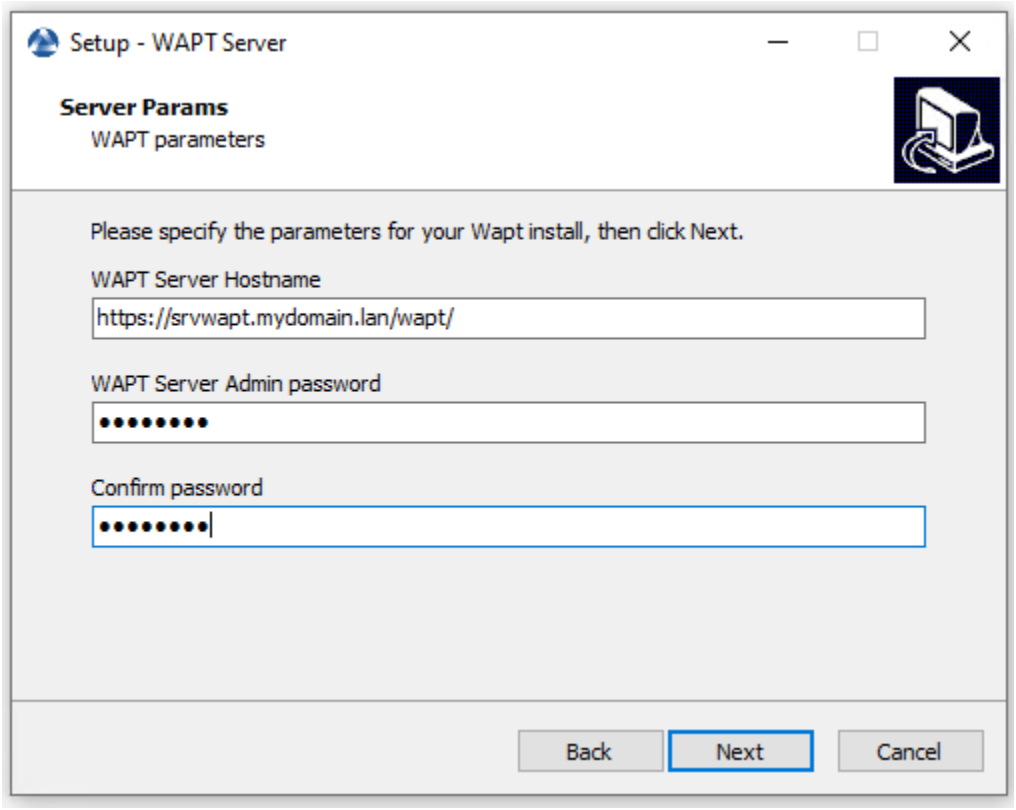
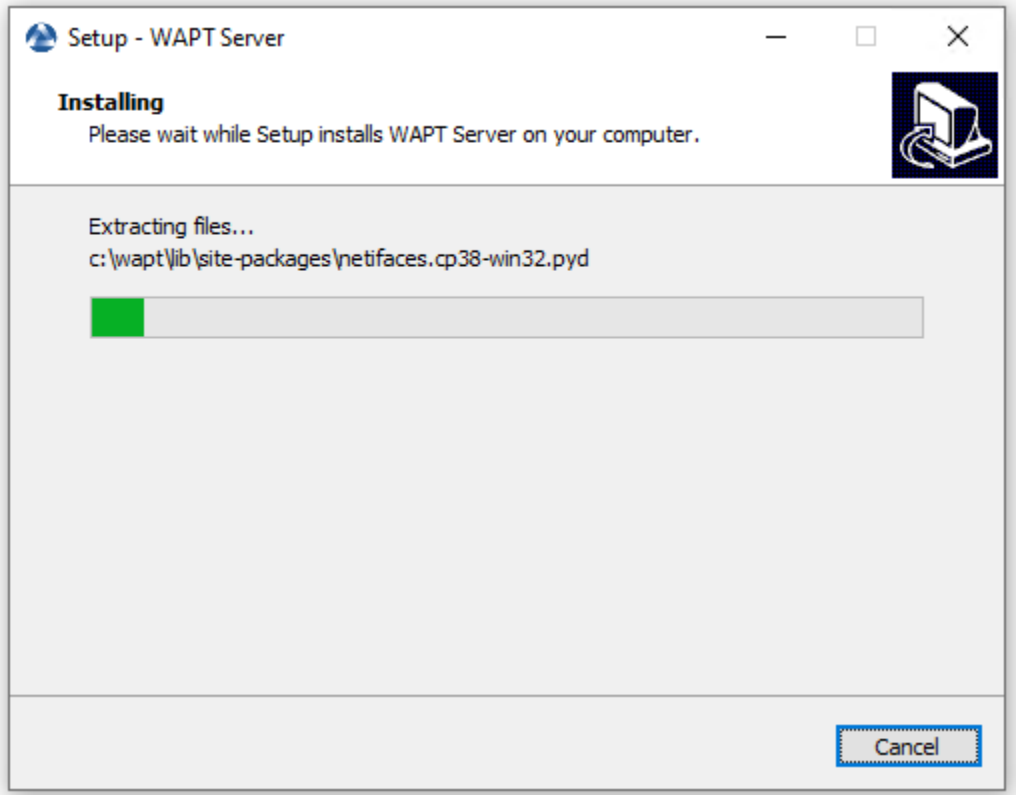


FIG. 1 – Choix des options du programme d'installation pour le déploiement du serveur WAPT



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



— Cliquez sur *Terminer* pour fermer la fenêtre.

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.

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

— Re-signez tous vos paquets WAPT.

— *En graphique*

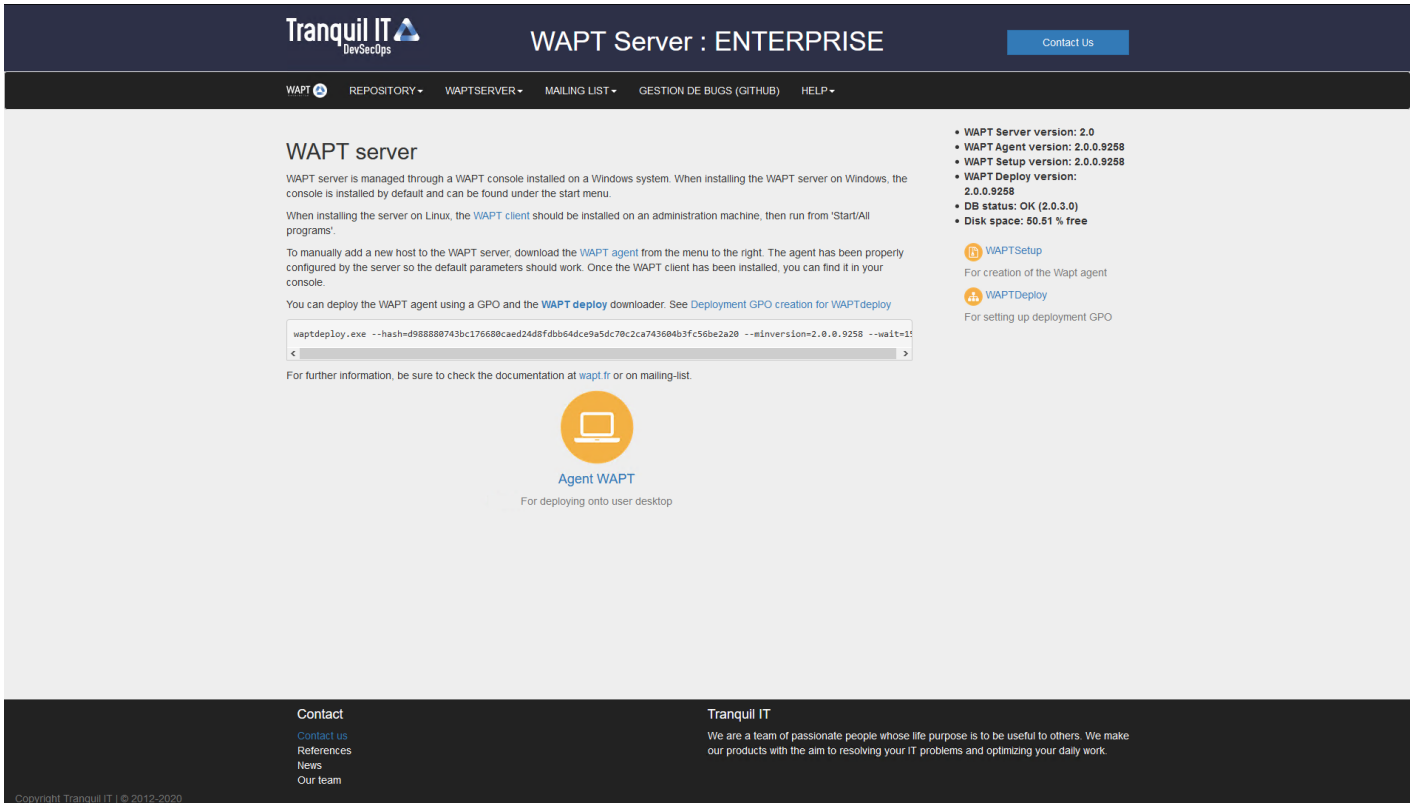


FIG. 2 – L'interface du serveur WAPT dans un navigateur web

CHAPITRE 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/
```

— 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
```

CHAPITRE 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
```

CHAPITRE 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*.

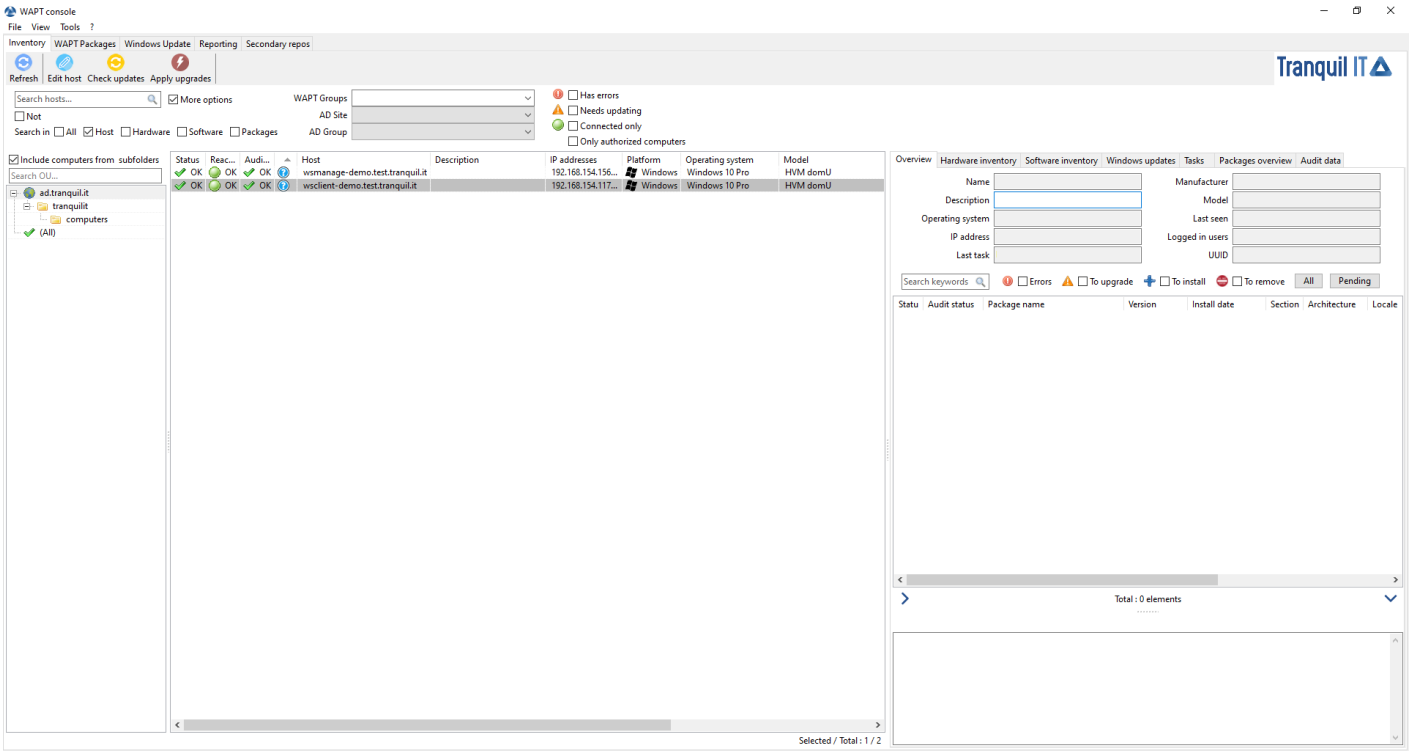


Fig. 1 – Méthode pour ajouter un paquet WAPT à la machine

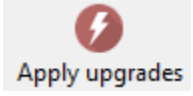
26.2 Vérifier les mises a jour sur l'hote

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*.

26.3 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

26.4 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.

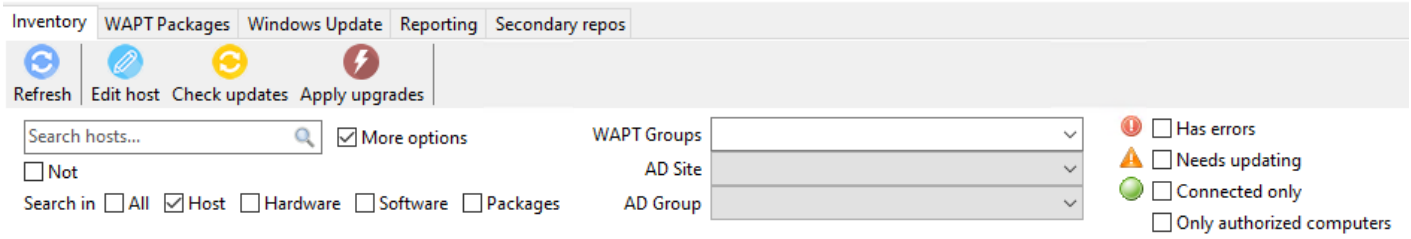


FIG. 2 – Les fonctionnalités de recherches avancées dans la console WAPT

TABLEAU 1 – 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 authorisés par le certificat de l'utilisateur en cours
WAPT Group	Filter les hôtes basés sur leur appartenance / dépendance à un paquet groupe WAPT
AD Site	Filter les hôtes basés sur leur appartenance / dépendance à Site et Services Active Directory
AD Group	Filter les hôtes basés sur leur appartenance / dépendance à un groupe Active Directory

Indication : Les filtres fonctionnent avec des expressions régulières.

26.5 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.

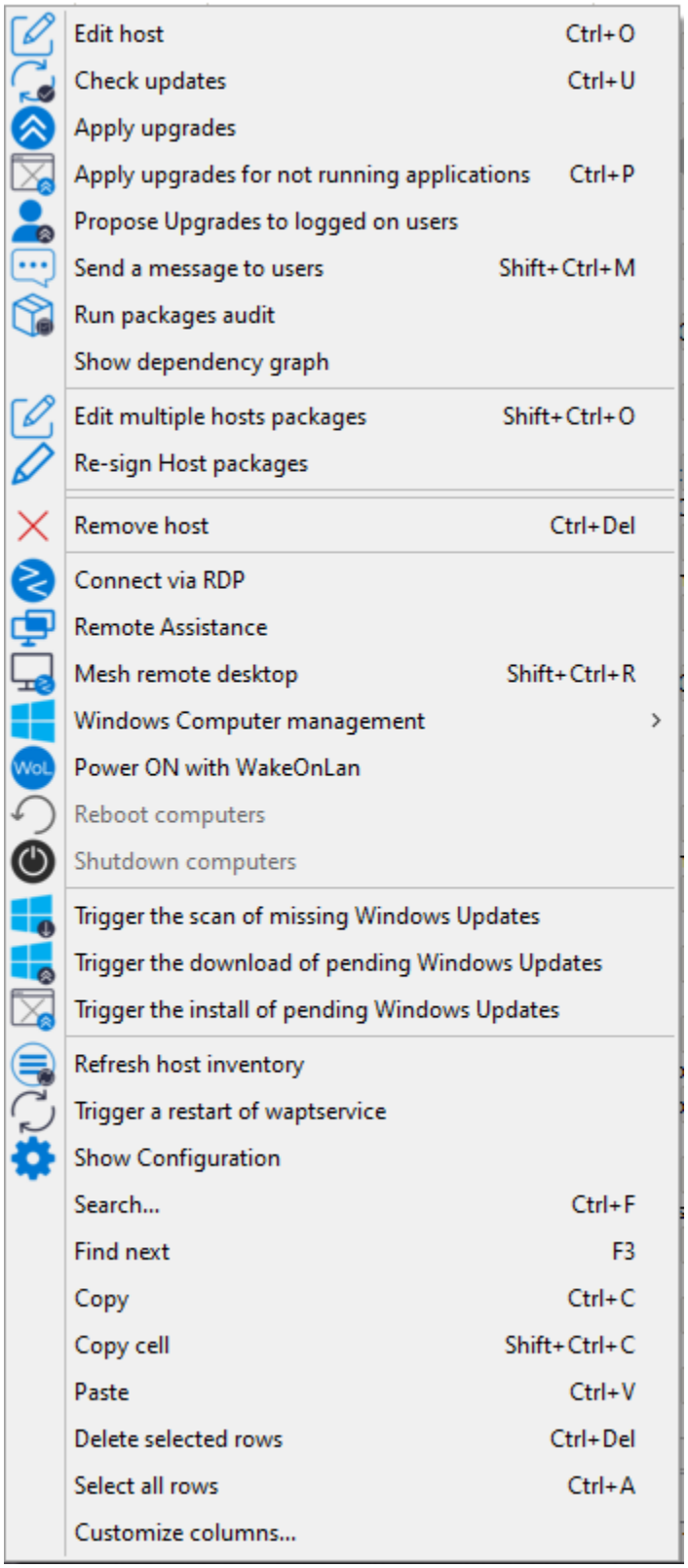
Cliquez sur le bouton *Refresh* ou appuyez sur `F5` sur le clavier.

FIG. 3 – 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*).

26.6 Comment effectuer des actions sur les hôtes ?



Certaines actions ne sont pas disponibles lorsque vous sélectionnez plusieurs hôtes.

Tableau 2: 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

26.7 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 packages pour maintenir son référentiel 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 référentiel 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*.

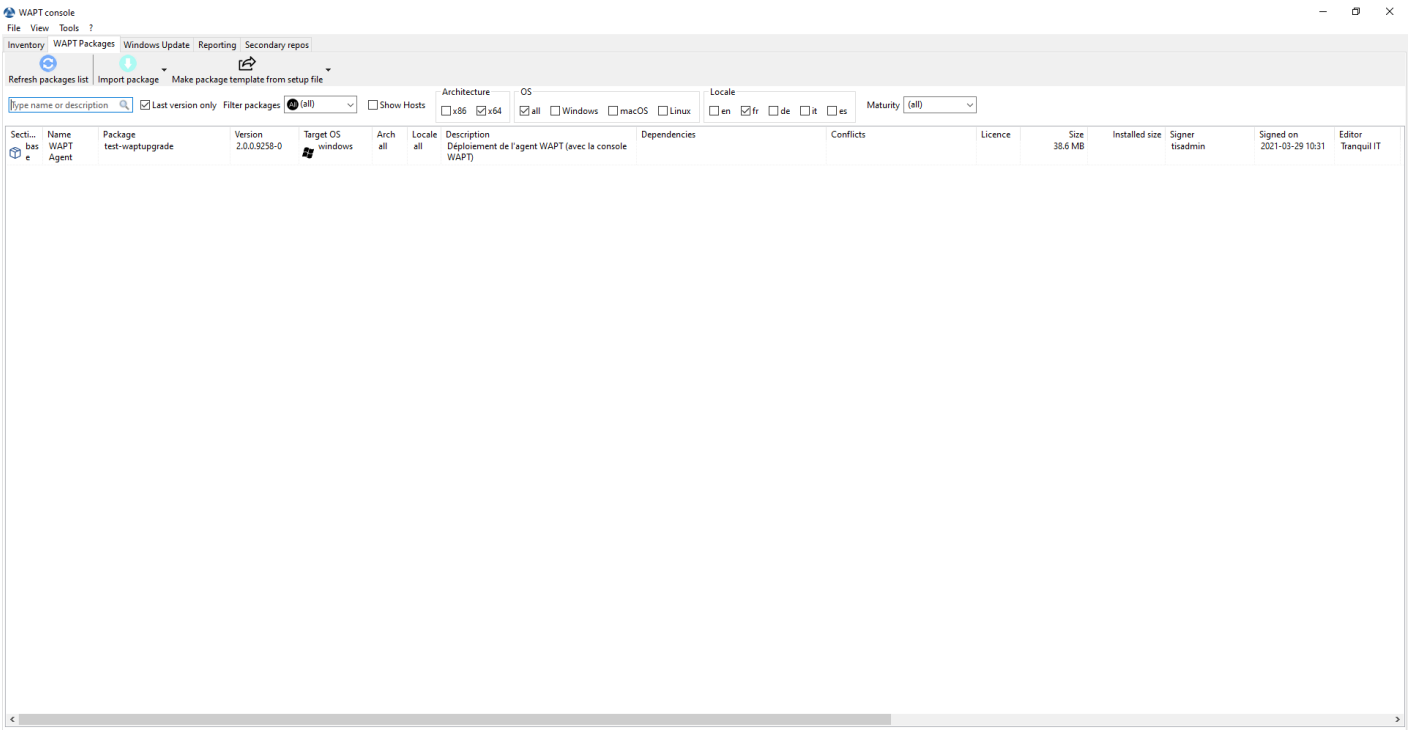


FIG. 4 – Les logiciels disponibles sont affichés dans la console WAPT

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*.

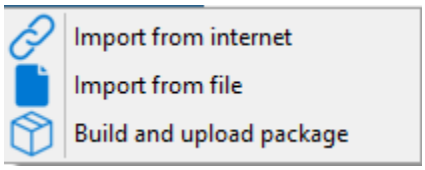
26.7.1 Importer un paquet depuis un dépôt externe sur Internet

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 présent, pour ajouter un autre dépôt cf paramètres de configuration des dépôts externes ;

Note : 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.

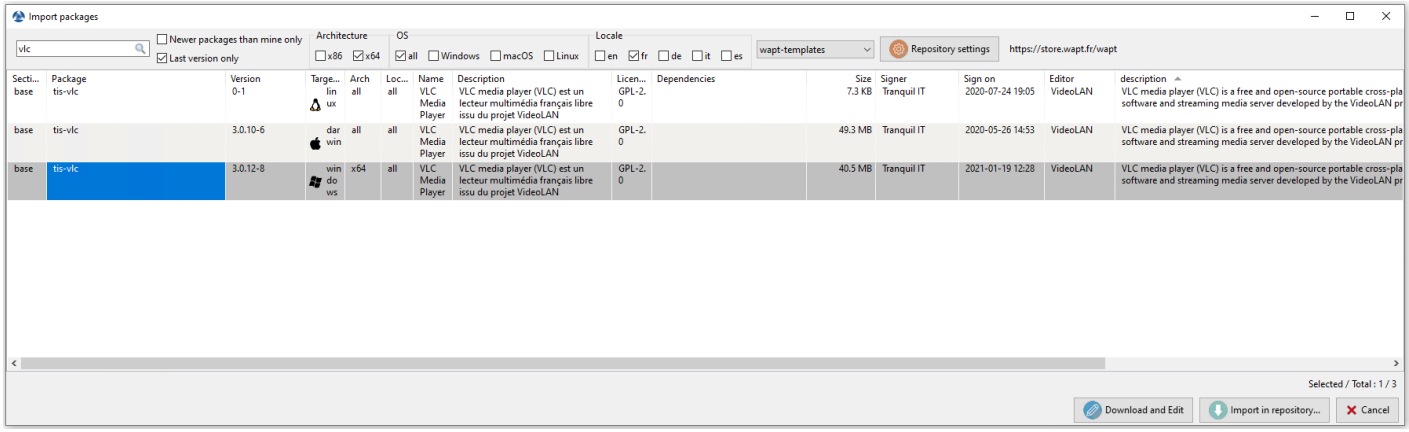
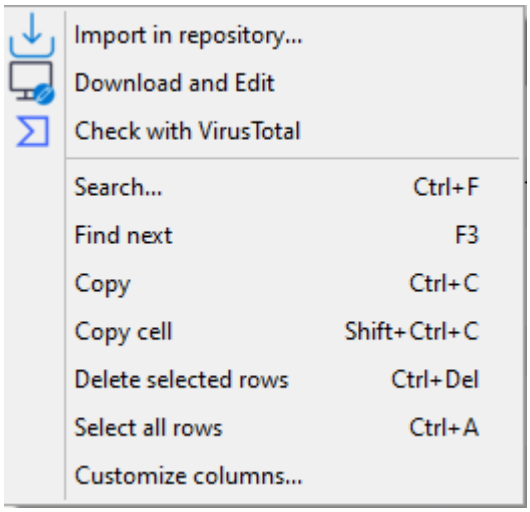


FIG. 5 – Le paquet WAPT importé s'affichera dans votre dépôt WAPT local

— Il y a 2 méthodes pour importer un paquet WAPT :
  — pour importer un paquet, sélectionner un paquet puis *Clic-droit* → *Importer* ;



— ou en bas a droit de la console *Importer dans le dépôt* :

— Valider l'import dans votre dépôt local.

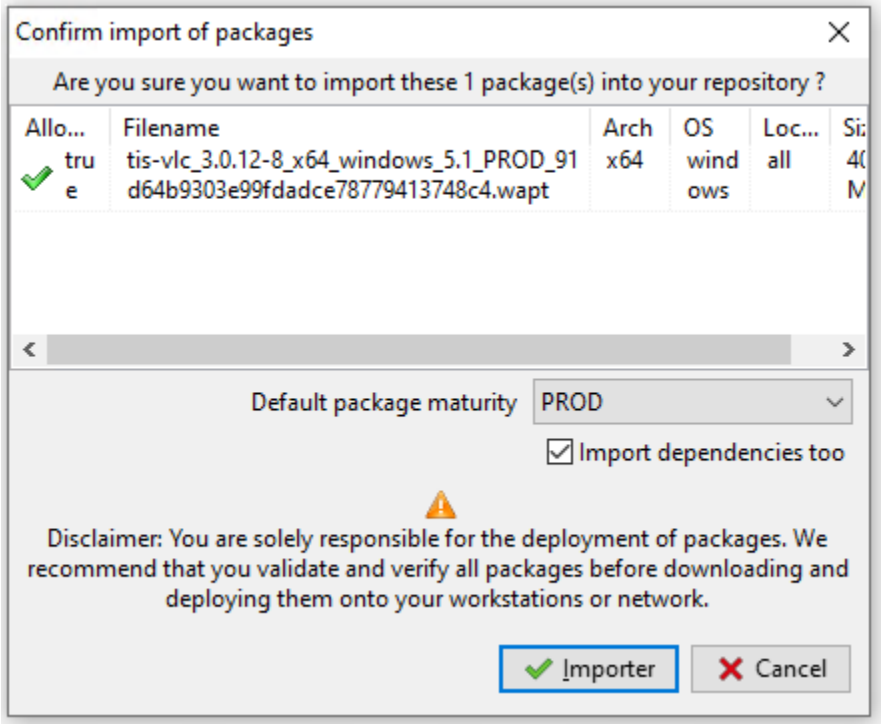
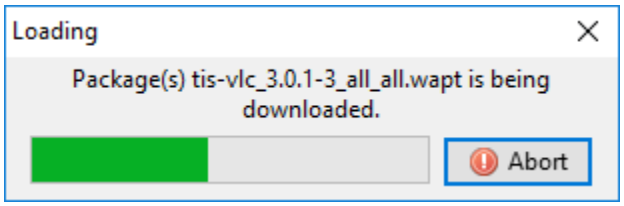


FIG. 6 – Boîte de dialogue pour préparer et confirmer l'importation d'un packaging WAPT dans un référentiel WAPT

Note : Il est possible de *changer la maturité d'un paquet WAPT* avant d'importer le paquet dans votre dépôt privé.

— Le téléchargement du paquet commence.



— Puis, entrer le mot de passe de votre clé privé.

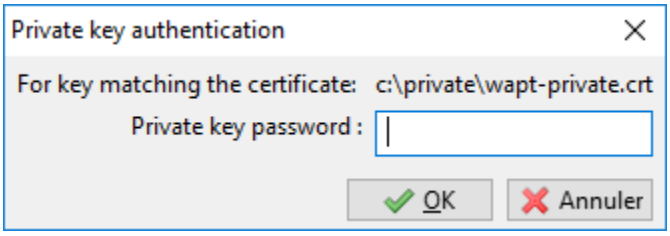


FIG. 7 – Entrez le mot de passe pour déchiffrer la clé privée

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.

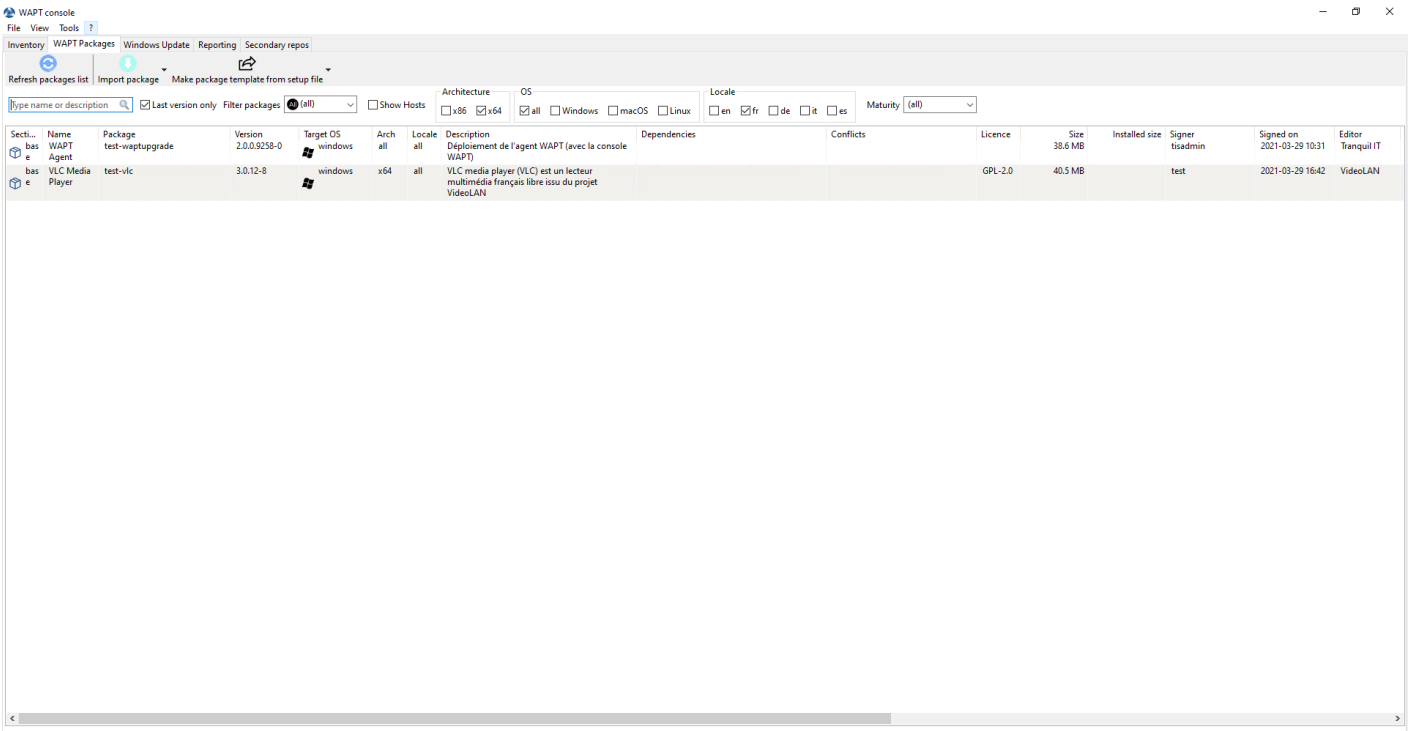


FIG. 8 – La console WAPT affiche le paquet importé

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*.

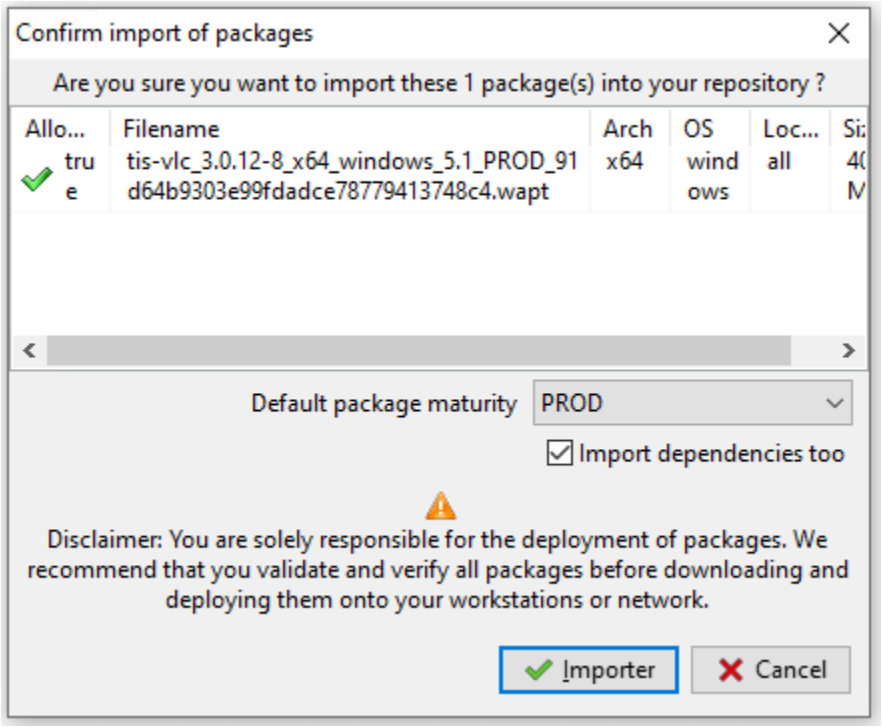
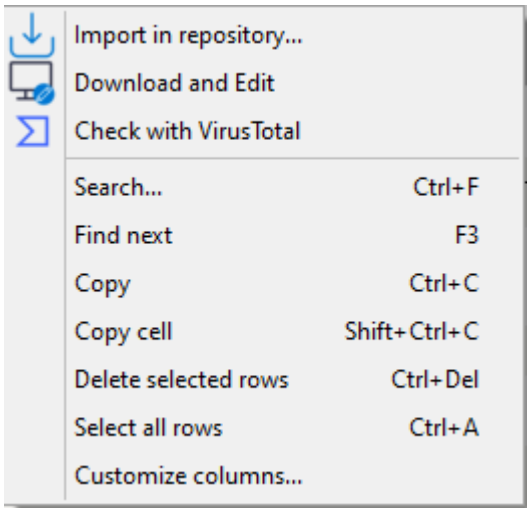


FIG. 9 – Boîte de dialogue pour préparer et confirmer l'importation d'un packaging WAPT dans un référentiel WAPT

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 ;

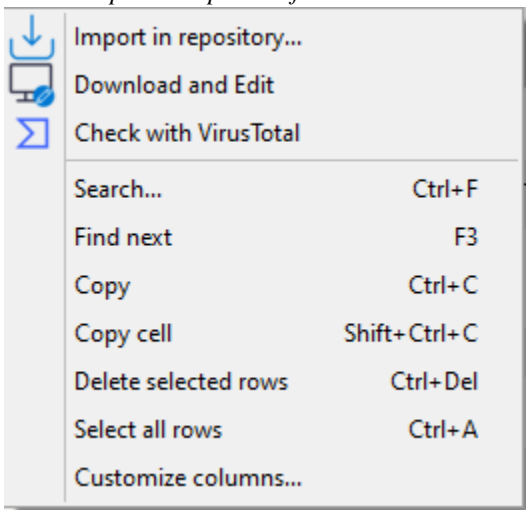
PyScripter,s'il est installé, va ouvrir les fichiers `control` et `setup.py` du paquet WAPT.

Pour plus d'information, visitez la documentation sur *créer un paquet WAPT*.

26.8 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*.



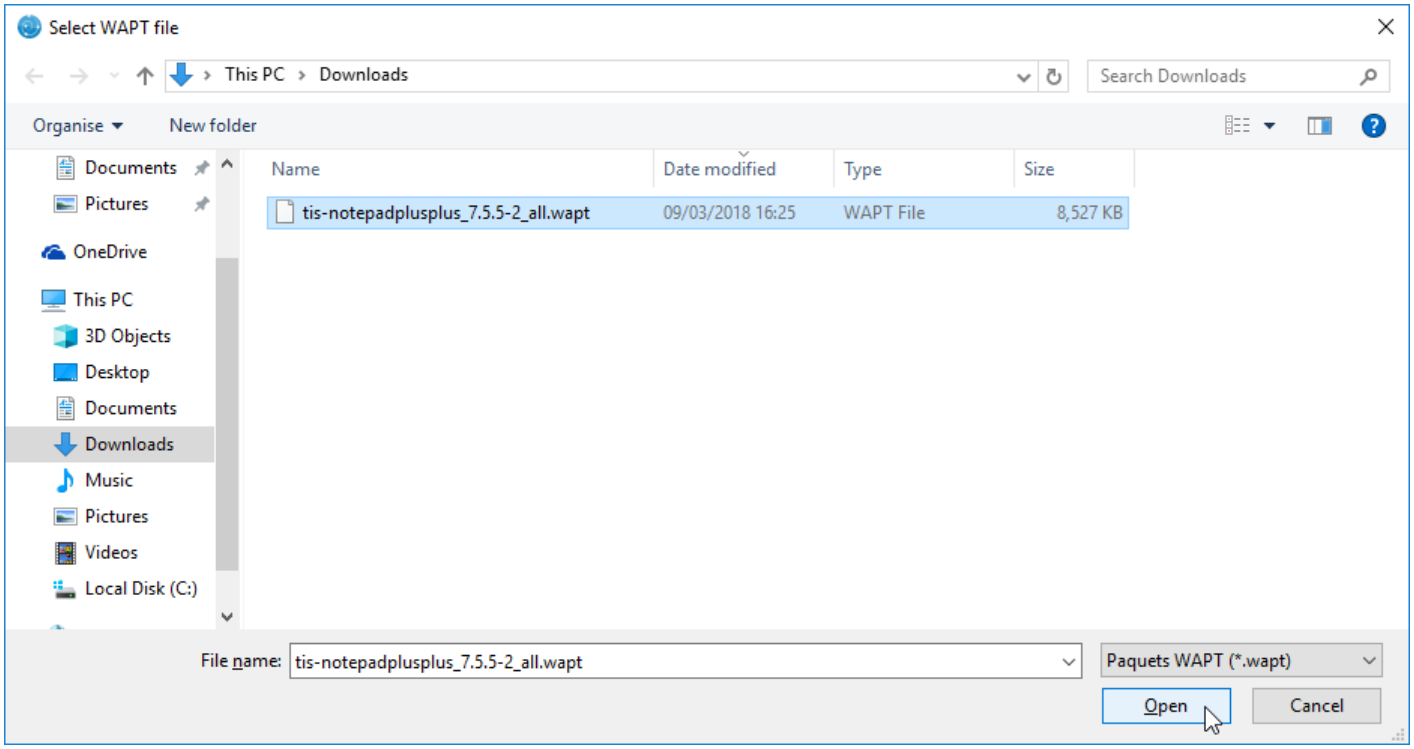
— Sélectionner le fichier à importer.

— 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.

Note : Il n'est pas possible de changer la maturité avant d'importer dans ce cas.



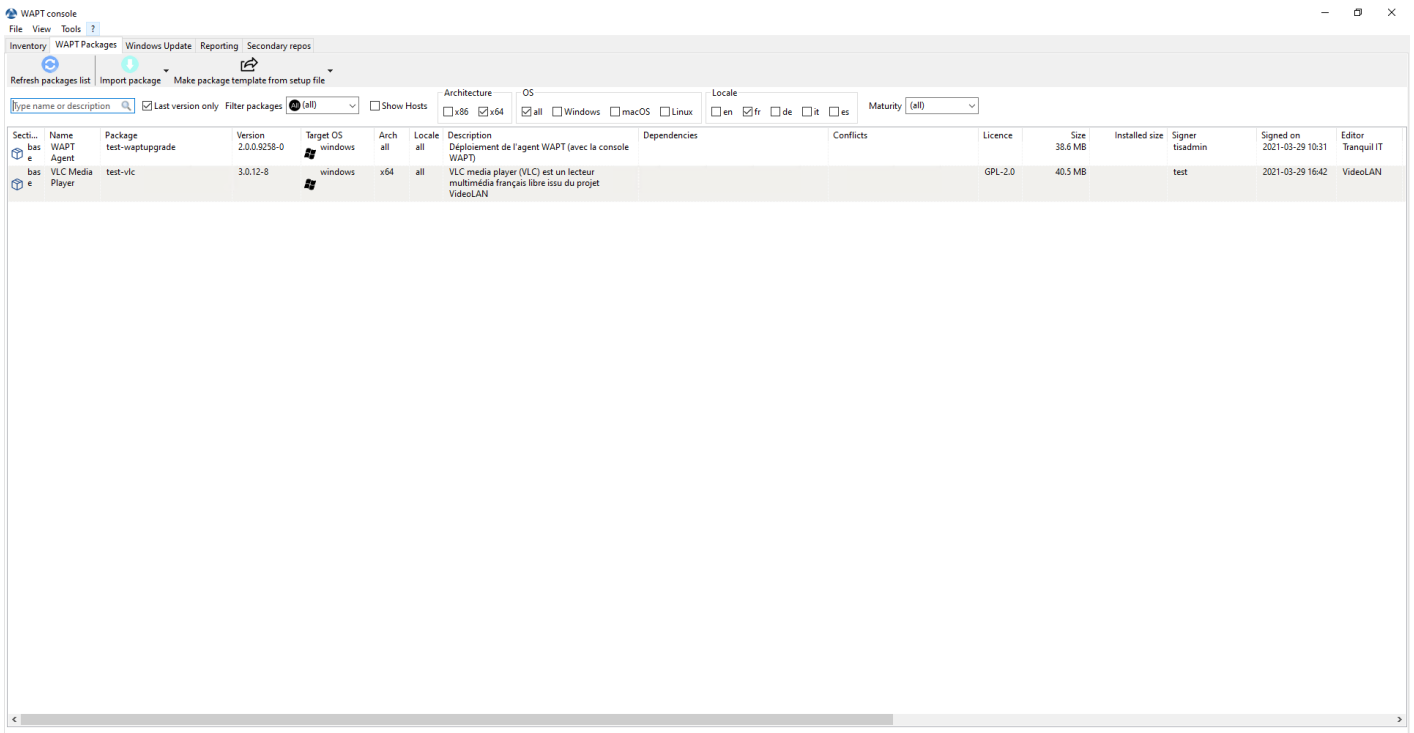


FIG. 10 – La console WAPT affiche le paquet importé

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.

26.9 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.

Tableau 3 – 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 (système d'exploitation).
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.

26.9.1 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*).

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.

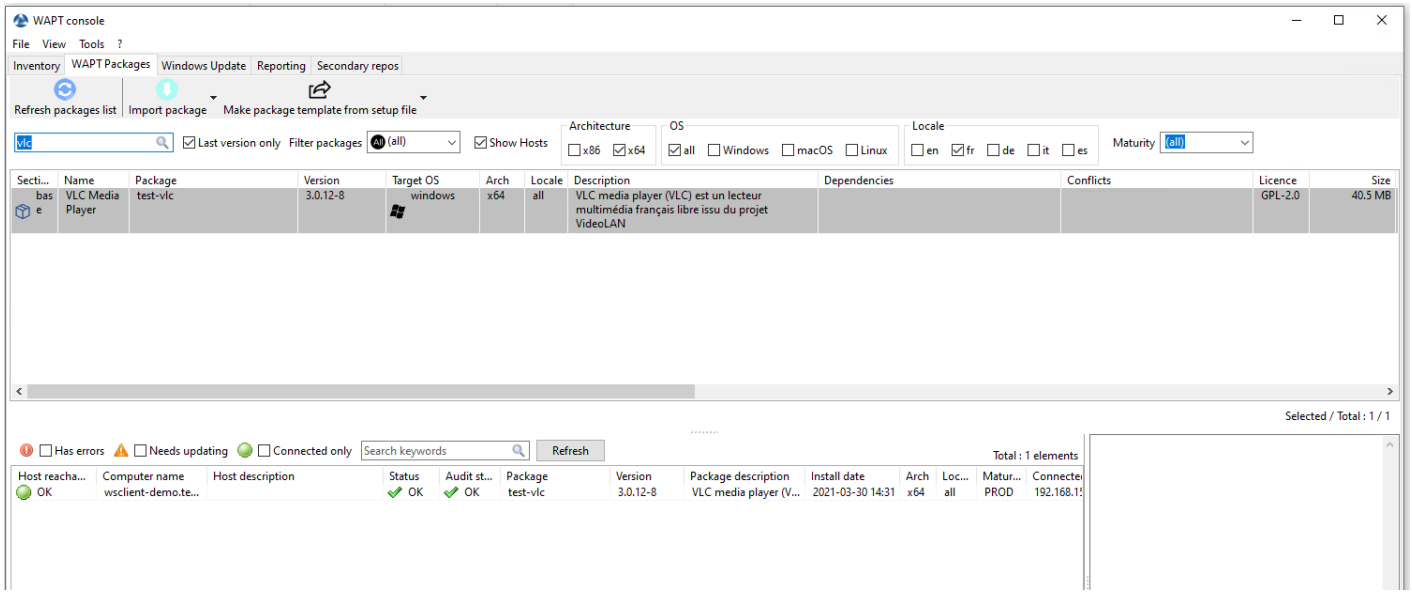


FIG. 11 – Faire une recherche basé sur un paquet WAPT

26.9.2 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*.

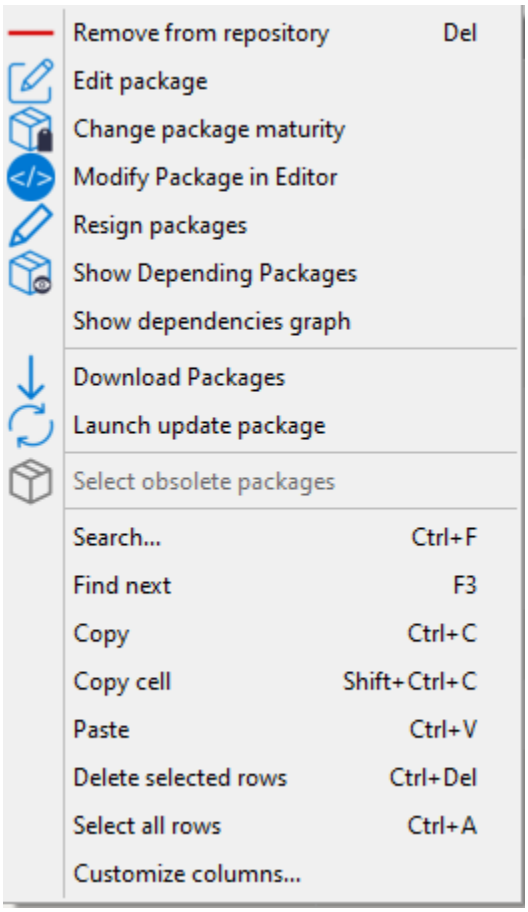


TABLEAU 4 – Changer la maturité d'un paquet WAPT

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 les paquets WAPT.

Note : 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 ✅.

Indication : Vous pouvez changer la maturité de multiples paquets en une fois

Avertissement : 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.

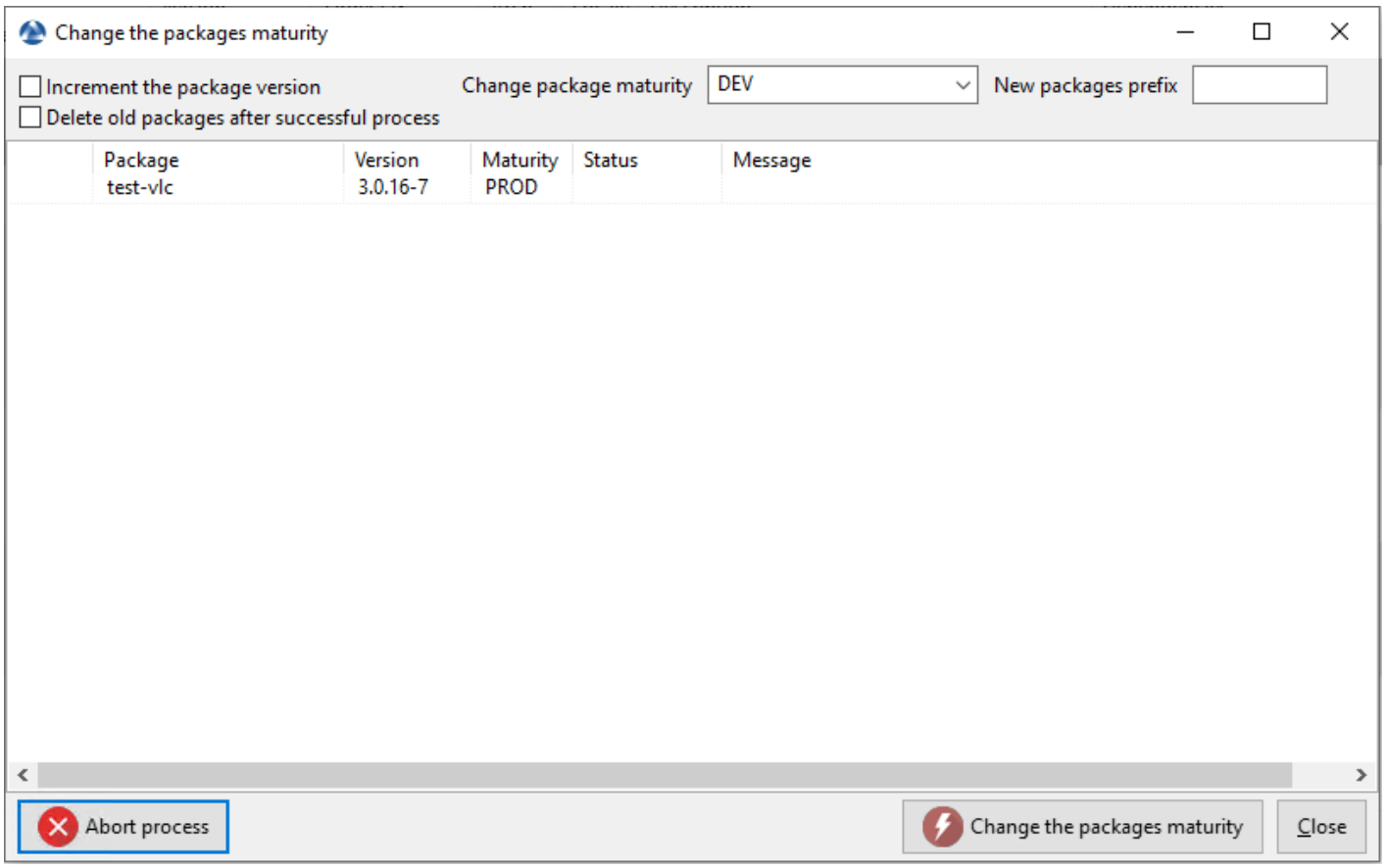
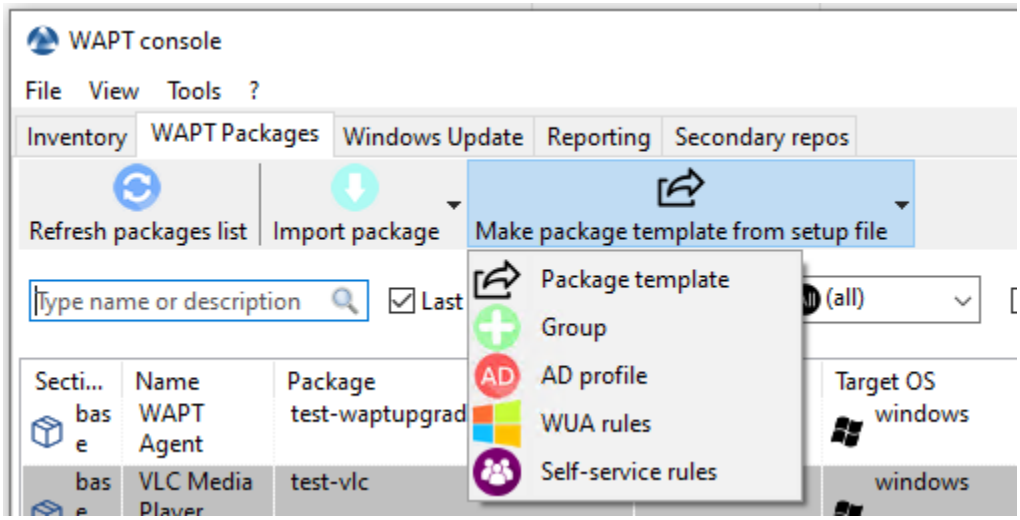


FIG. 12 – Changer la maturité d'un paquet WAPT

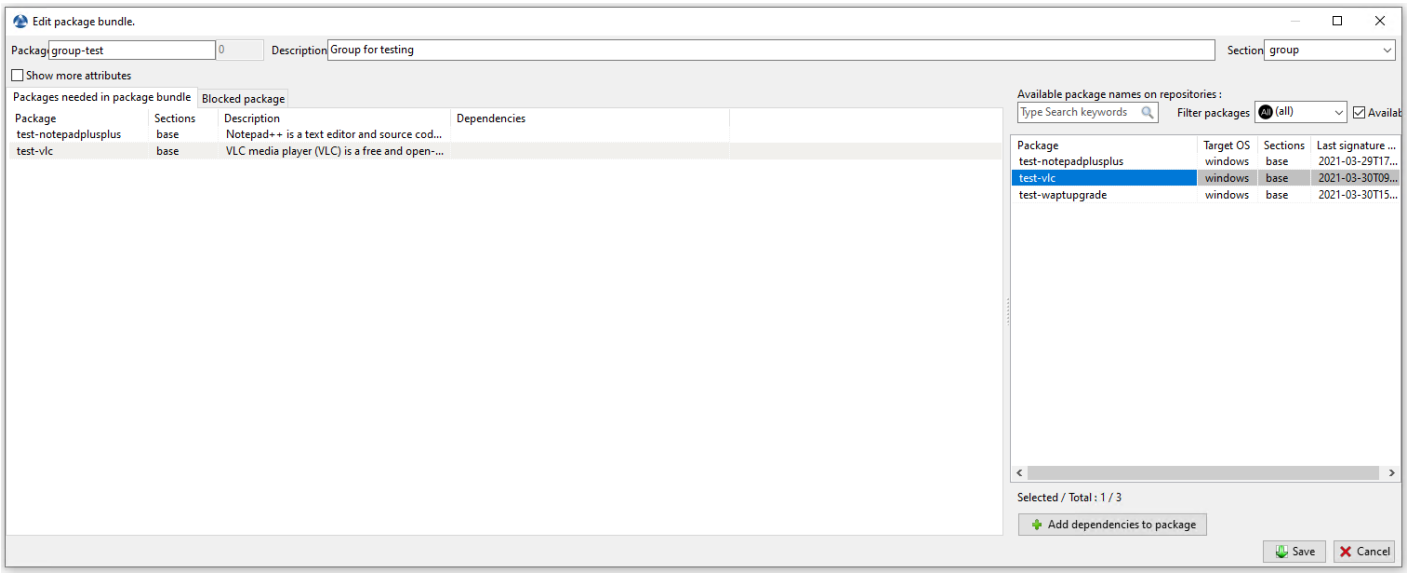
26.9.3 Créer un groupe de paquet

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 au groupe en les glissant et déposant ou bien en faisant un clic-droit sur le nom du paquet, et ajoutez-le au paquet.



— Cliquer sur *Enregistrer* pour enregistrer le paquet groupe.

Indication : Pour désinstaller un paquet, il est possible de l'ajouter comme interdit à un groupe de paquet. Le paquet interdit, si

installé, sera supprimé avant que d'autres paquets soient installés.

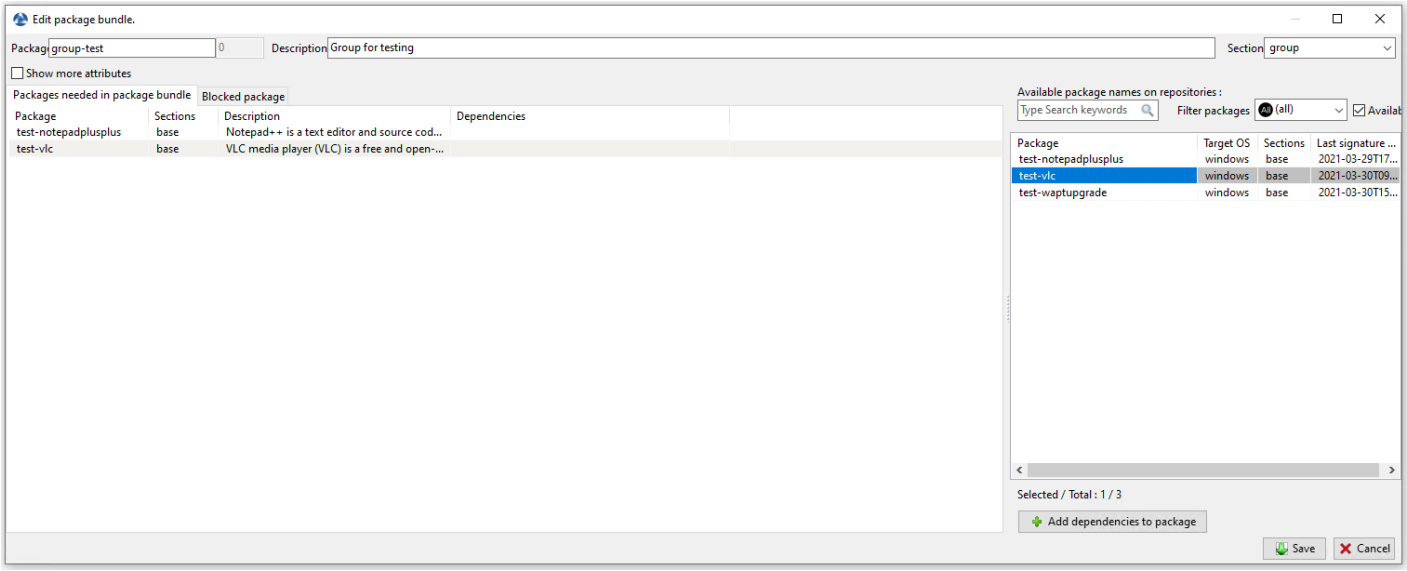


FIG. 13 – Ajouter un paquet interdit à la machine

26.9.4 Supprimer un paquet WAPT

Pour supprimer un paquet depuis le dépôt, *Clic-droit → Supprimer du dépôt*.

Indication : Vous pouvez sélectionner plusieurs paquets a la fois pour les supprimer.

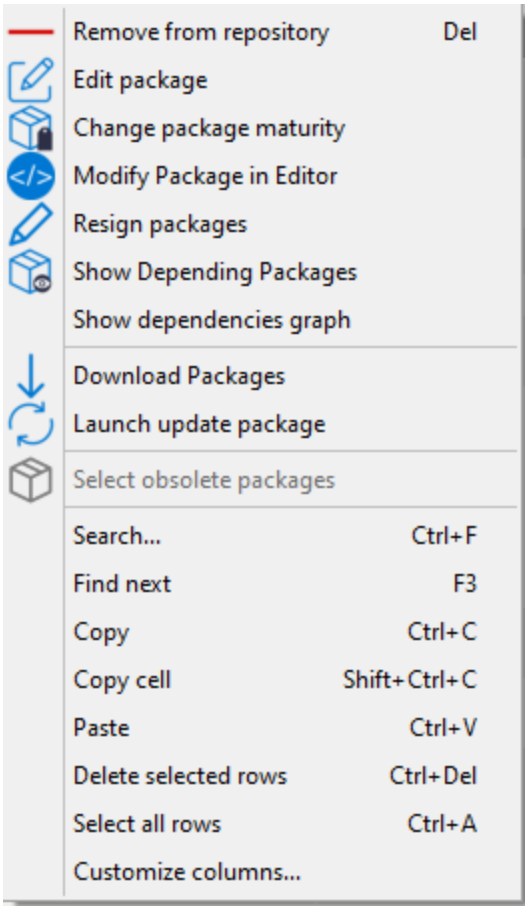
26.9.5 Editer un paquet WAPT

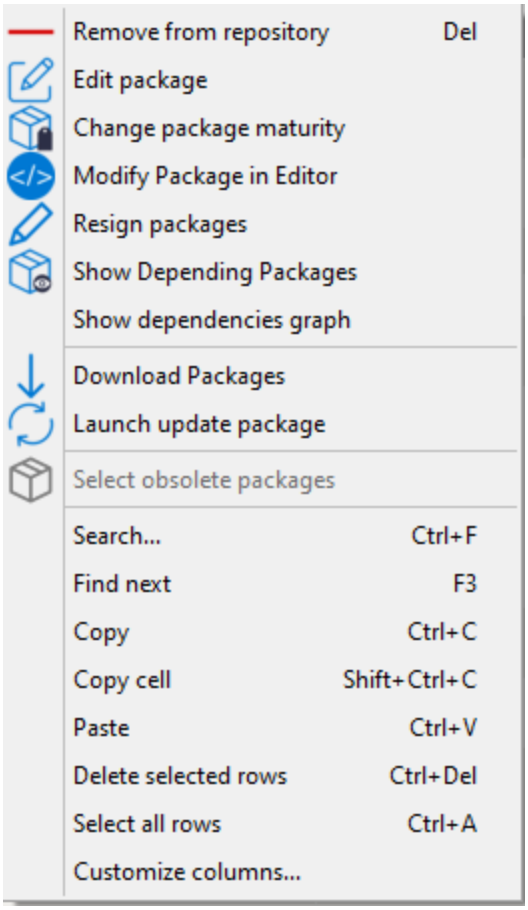
Pour éditer un paquet, *clique-droit → Editer le paquet*.

The WAPT package will be downloaded locally in **the base package development folder**, set in the WAPT Console settings.

Si **`PyScripter`** est installé, il ouvrira automatiquement les fichiers `control` et `setup.py`.

Une fois édité vous pouvez téléverser *depuis la console wapt*.


Remove from repository Del
Edit package
Change package maturity
Modify Package in Editor
Resign packages
Show Depending Packages
Show dependencies graph
Download Packages
Launch update package
Select obsolete packages
Search... Ctrl+F
Find next F3
Copy Ctrl+C
Copy cell Shift+Ctrl+C
Paste Ctrl+V
Delete selected rows Ctrl+Del
Select all rows Ctrl+A
Customize columns...


Remove from repository Del
Edit package
Change package maturity
Modify Package in Editor
Resign packages
Show Depending Packages
Show dependencies graph
Download Packages
Launch update package
Select obsolete packages
Search... Ctrl+F
Find next F3
Copy Ctrl+C
Copy cell Shift+Ctrl+C
Paste Ctrl+V
Delete selected rows Ctrl+Del
Select all rows Ctrl+A
Customize columns...

26.9.6 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.

CHAPITRE 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 paquet profile Active Directory.

Cela automatise l'instalaltion du logiciel WAPT ainsi que ses paquets de configuration sur des hôtes, basé sur leur appartenance aux Groupes de SecuritéOrdinateur Active Directory.

Important : Les groupes de Securité Ordinateur Active Directory contiennent des Ordinateurs et non pas des Utilisateurs.

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.

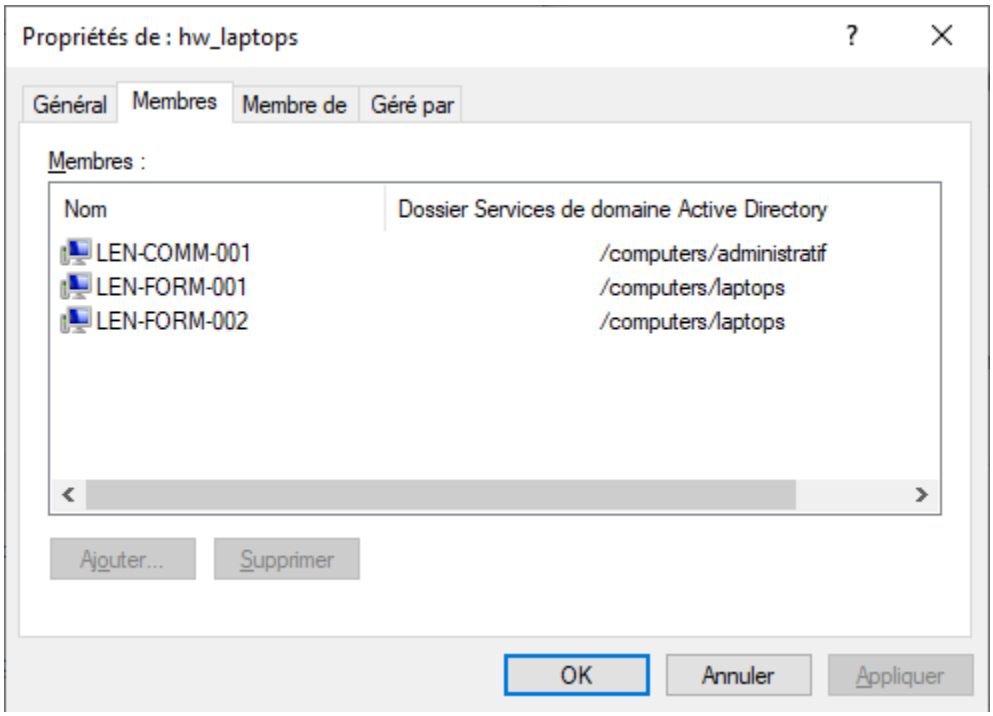
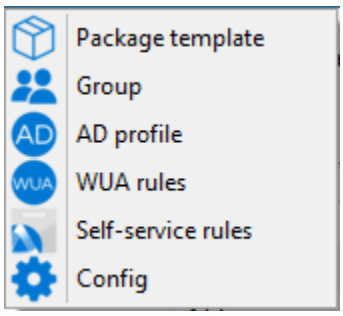


FIG. 1 – Fenêtre montrant le groupe Computers dans Active Directory

27.1.2 Création de paquets de regroupement de profils 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 :

— The *profile* AD group name and the *profile* package **MUST** be all lower case.

Exemple :

— AD Security group : `hw_laptops` ;

— WAPT profile bundle : `hw_laptops`.

Une fenêtre s'ouvre et on vous demande de choisir quels paquets doivent être contenus dans le paquet **profile** tout juste créé.

Enregistrez le paquet *profil* et il sera téléchargé sur le serveur WAPT.

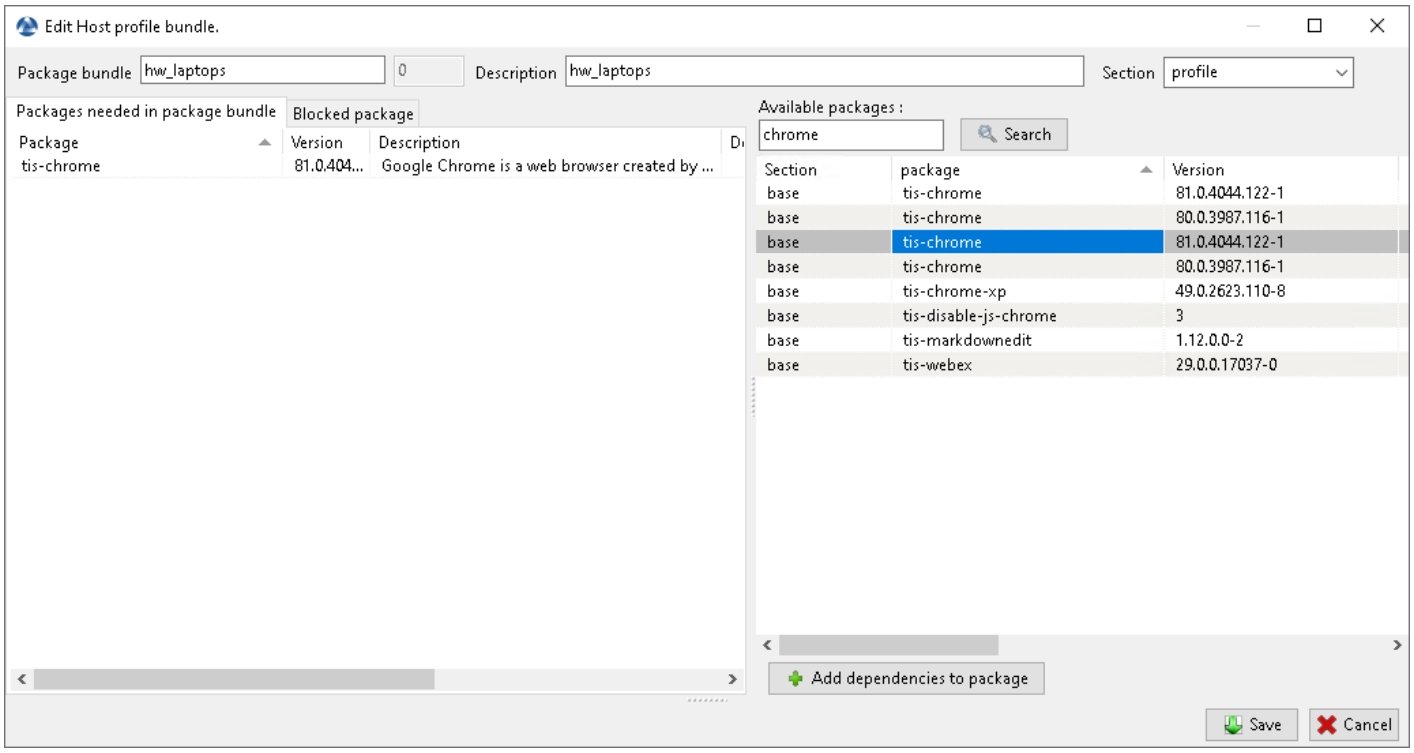


FIG. 2 – Ajout de paquets WAPT à un paquet *profile*

27.2 Utiliser des Unités Organisationnelles dans WAPT

27.2.1 Principe de fonctionnement

WAPT Enterprise propose la fonctionnalité de paquets d'unité organisationnelle.

Il automatise les installations de logiciels en fonction de votre organisation Active Directory. Il s'agit d'une fonctionnalité très puissante si elle est utilisée correctement.

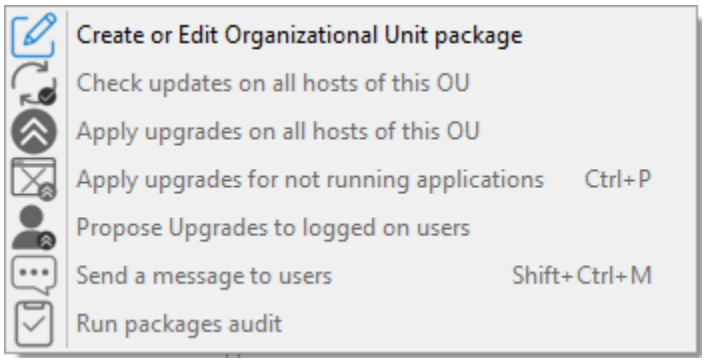
L'agent WAPT connait son emplacement dans l'arborescence Active Directory, il connaît donc pour cette raison la hiérarchie des Unités Organisationnelles qui le concerne, par exemple :

```
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 d'Unité Organisationnelle est défini à chaque niveau, l'agent WAPT téléchargera automatiquement les paquets et les configurations qui sont attachés à chaque niveau. En utilisant l'héritage, WAPT appliquera les paquets 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 vil vous ai demandé de choisir les paquets qui doivent être inclus dans le paquet **unit**.

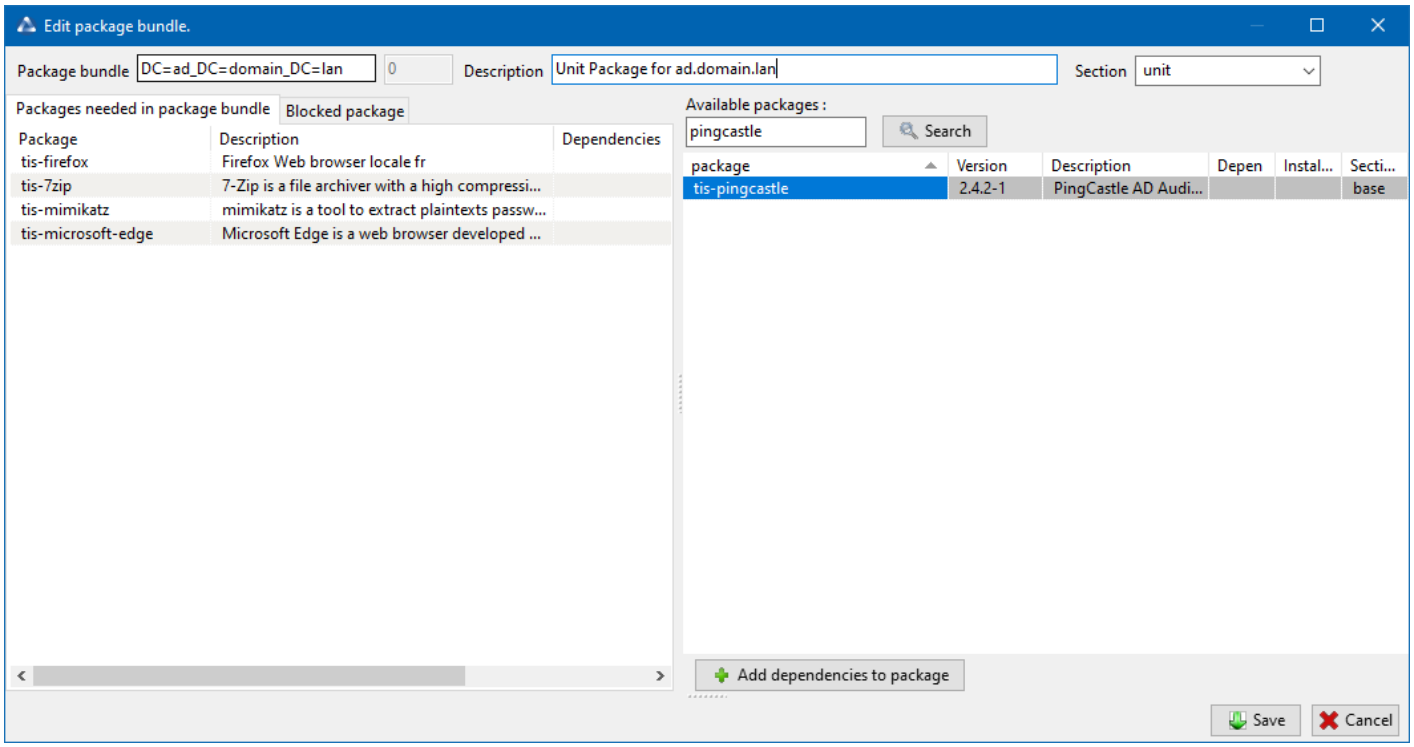


FIG. 3 – Ajouter des paquets au paquet unit

Sauvegarder le paquet et il sera déployé sur tous les hôtes appartenant à cette OU.

27.2.3 Les actions possibles avec les Unités Organisationnelles

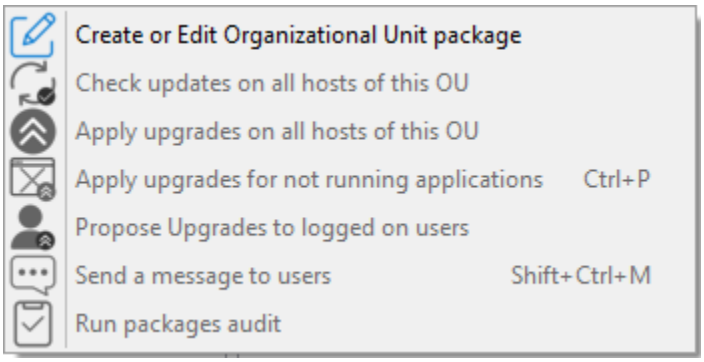
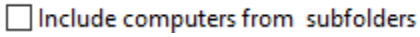


Tableau 1 – Créer ou éditer des paquets d'Unité Organisationnelles

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 tous les hôtes apparaissent dans la console WAPT sous la bonne unité organisationnelle, qu'ils soient joints à un domaine AD ou non, WAPT vous permet de spécifier une *fausse* Unité Organisationnelle 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 = []
```

(suite sur la page suivante)

(suite de la page précédente)

```
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 ...).
- Continuez à suivre ce chemin dans les champs DN/computer_ad_dn dans la grille d'inventaire.

27.3 Ajouter des plugins dans la Console

Pour ajouter des plugins, aller dans *Outils*, Préférences et onglet Outils externes.

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 :

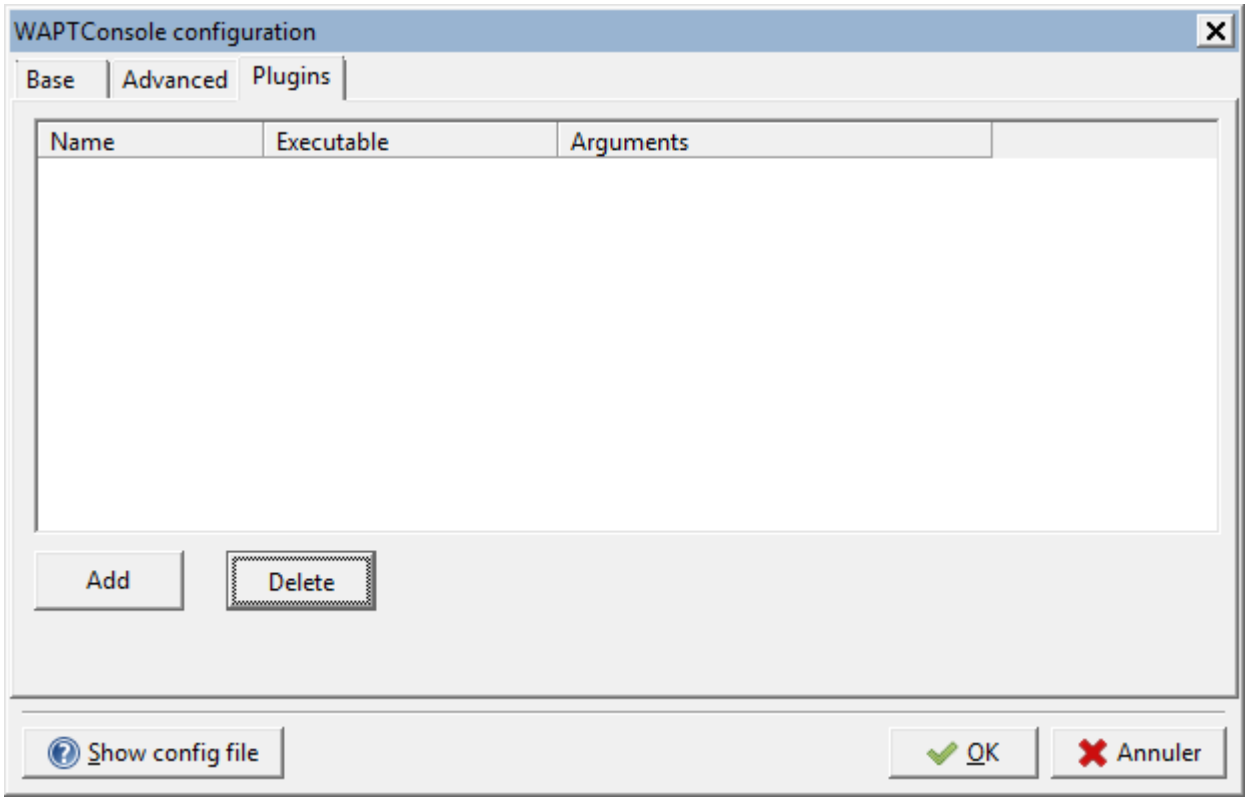


FIG. 4 – Créer un outil externe personnalisé pour WAPT

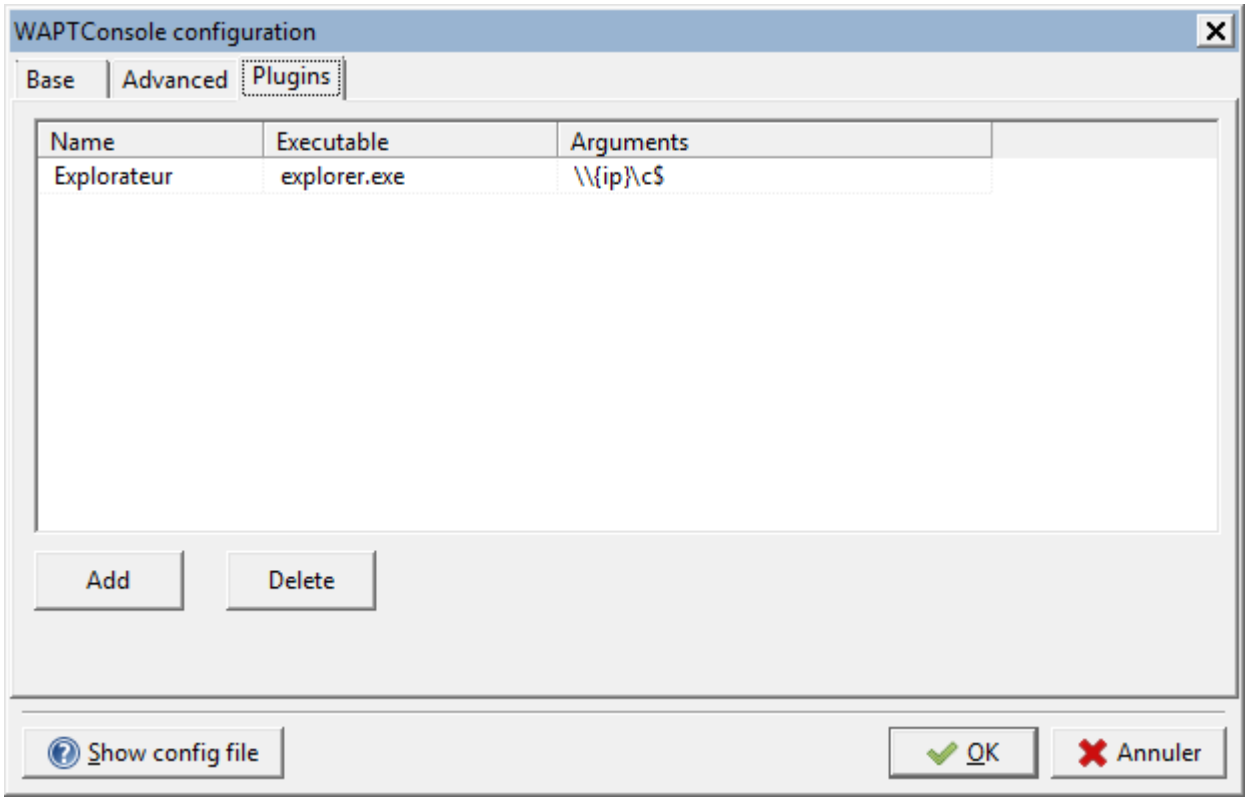


FIG. 5 – Créer un outil externe personnalisé pour WAPT

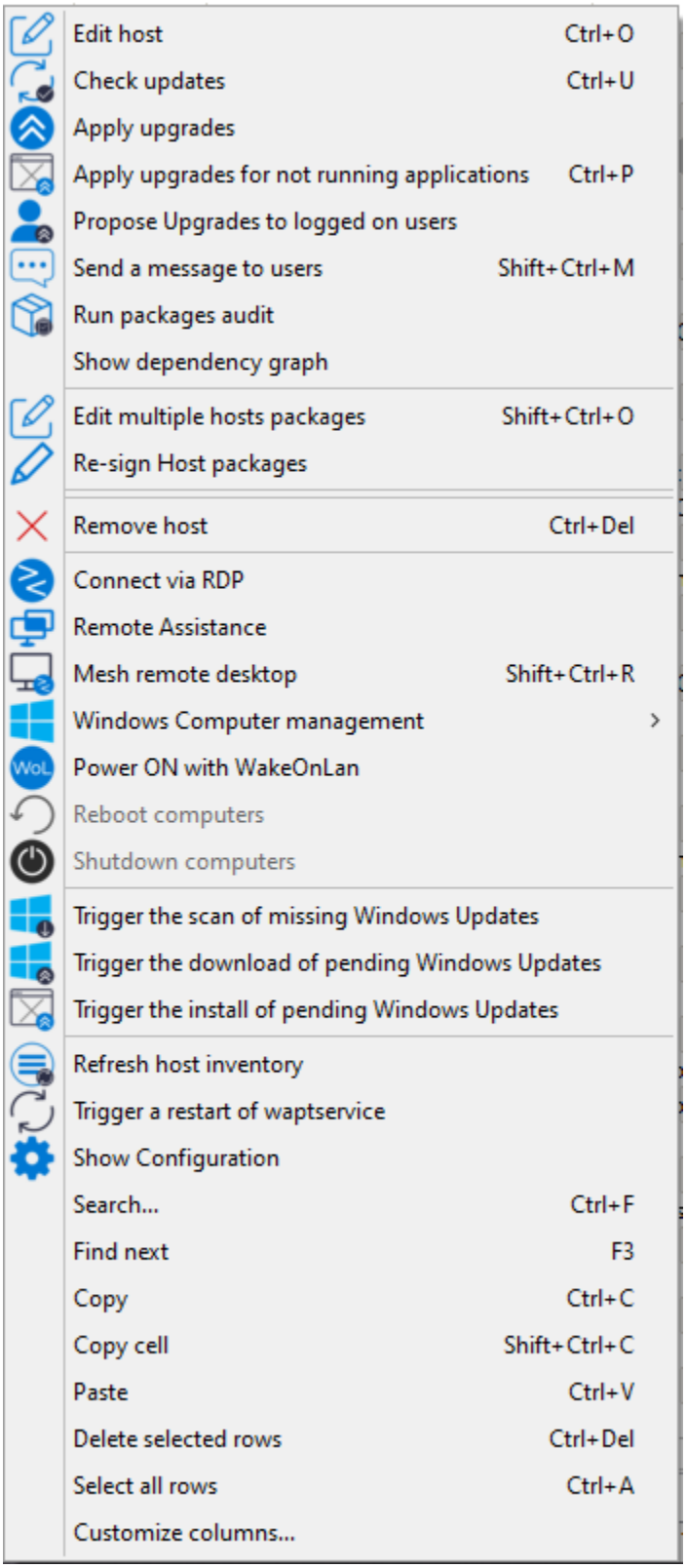
27.4 Re-Signer tous les paquets avec la console WAPT

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).

Indication : Utilisez le certificat Administrateur pour re-signer les paquets.

27.4.1 Fenêtre de re-signature des paquets WAPT

— 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.

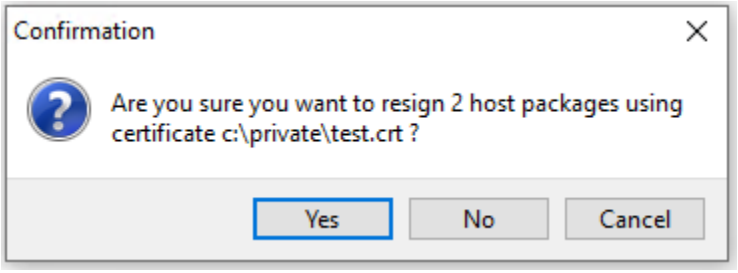


Fig. 6 – Confirmez la re-signature sur les hôtes sléectionnés.

— Puis, entrez votre clé privée.

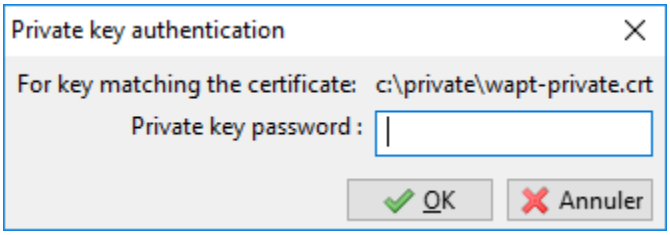


Fig. 7 – Entrer le mot de passe pour déchiffrer la clé privée

— Les paquets *hôtes* sélectionnés sont désormais tous re-signé avec la nouvelle méthode cryptographique exigée par Python3.

27.4.2 Re-signature d'autres types de packaging WAPT

— 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 ce procédé, ce qui prend du 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.

Éditez le paquet et suivez *cette procédure*.

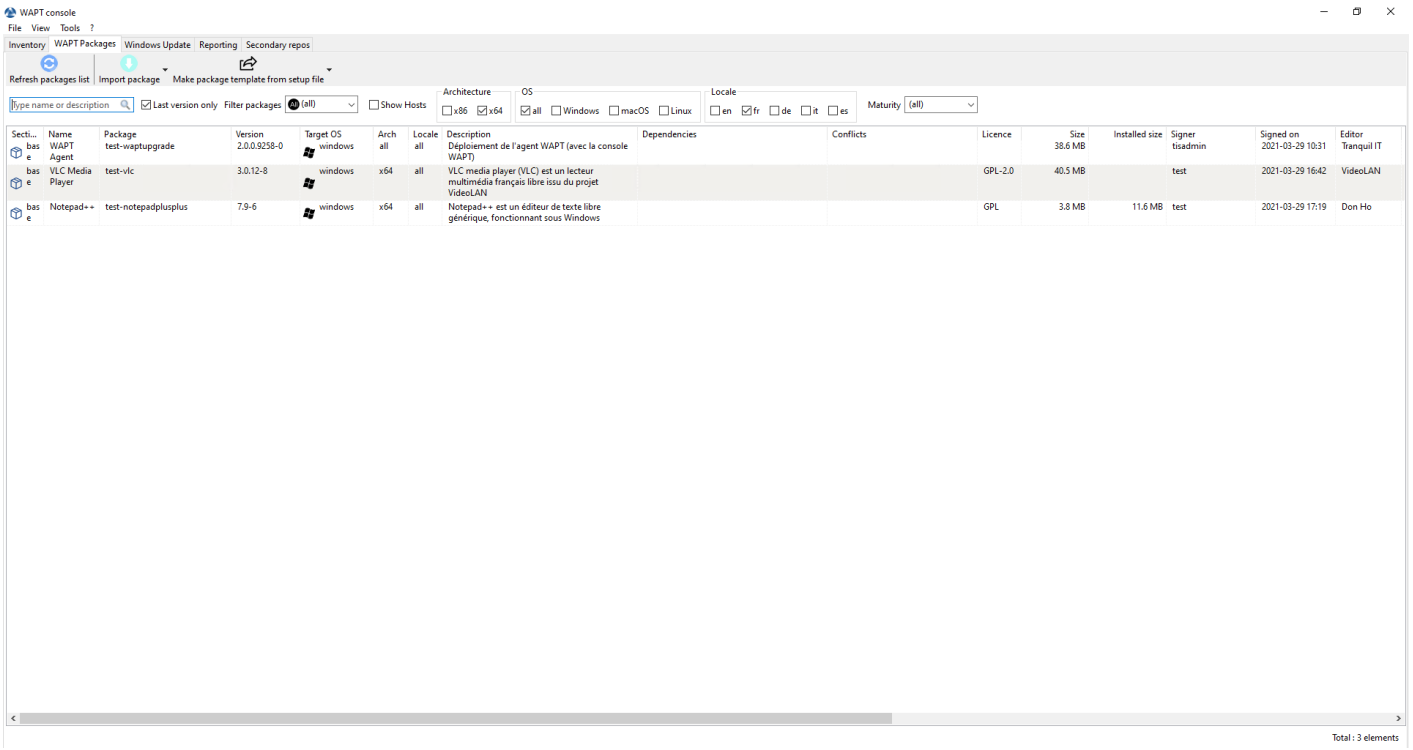


FIG. 8 – Les dépôts disponibles sur la console WAPT

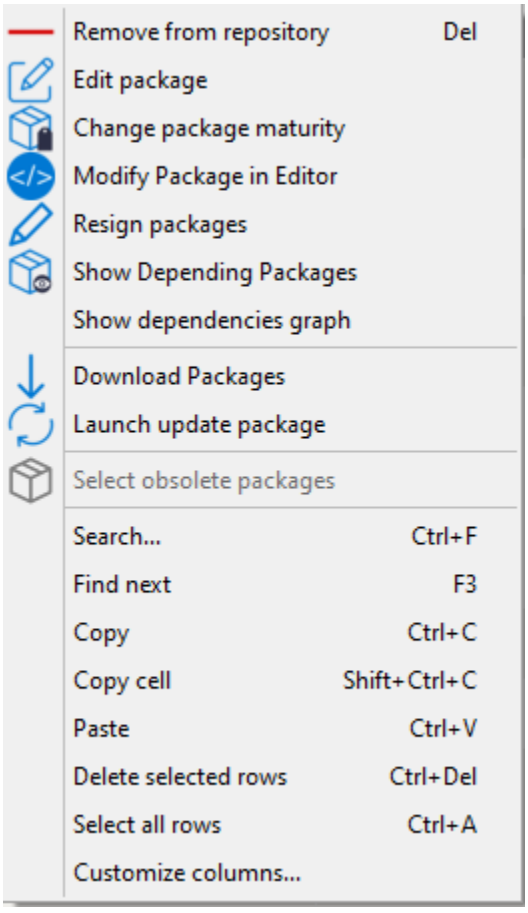


FIG. 9 – Options de menu pour les référentiels

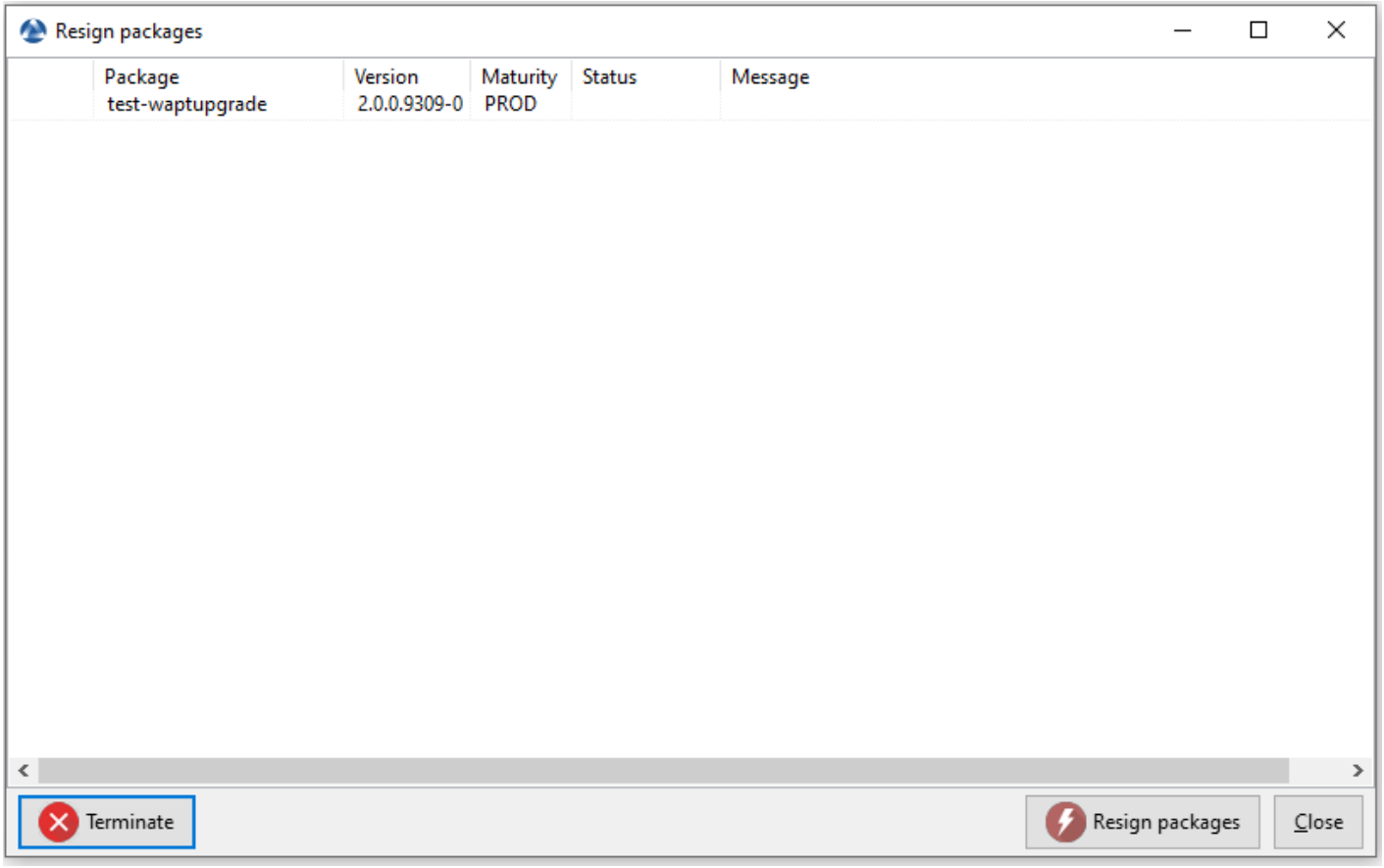


FIG. 10 – Fenêtre de re-signature des paquets WAPT

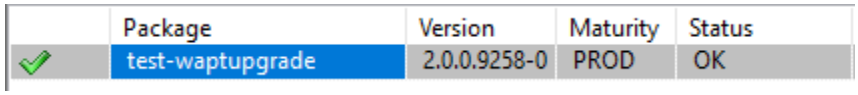


FIG. 11 – Le processus de signature s'est correctement terminé

CHAPITRE 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 sevreur WSUS.

Note : **L'interne de WAPTWUA est basé sur l'API WUA (Windows Update Agent).

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 : WAPTWUA n'est pas actuellement compatible avec l'utilisation de l'interface graphique du magasin 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.

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.

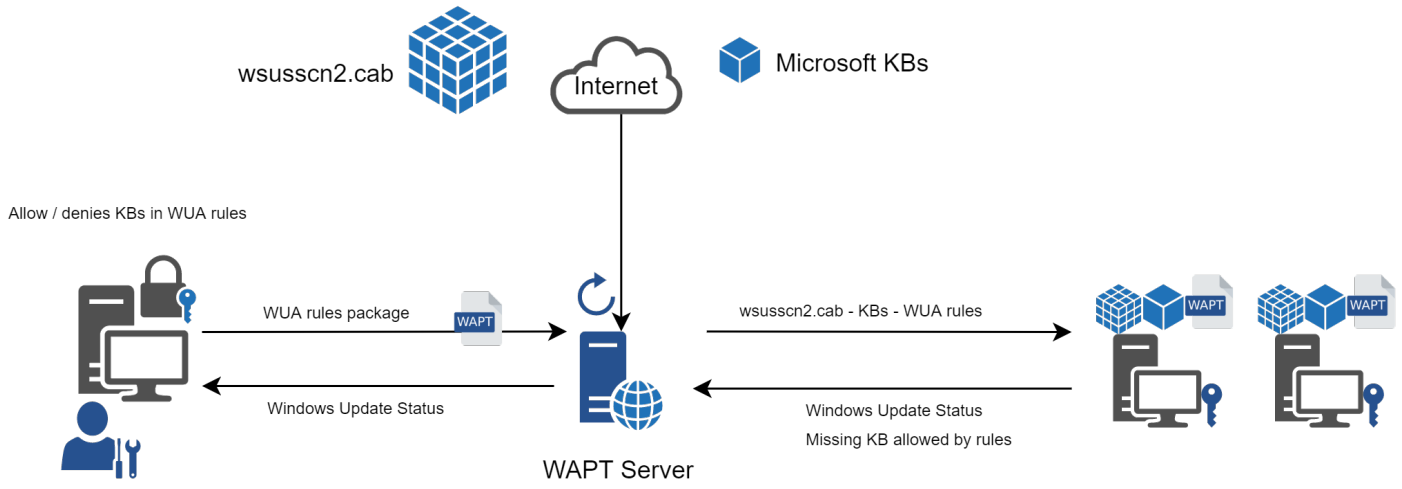


FIG. 1 – Diagramme de flux des mises à jour Windows WAPT

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 :

Tableau 1 – Les options de configuration dans la section `[waptwua]` dans le `wapt-get.ini`

Options (Valeurs par défaut)	Description	Exemple
`enabled` (par défaut `False`)	Activer ou désactiver WAPTWUA sur cette machine.	`enabled` = True
`direct_download` = False	Télécharger les mises à jour directement depuis les serveurs Microsoft.	`direct_download` = True
`default_allow` = False	Configuré si la mise à jour est autorisée ou pas par défaut.	`default_allow` = True
`download_scheduling` = 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` = 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` = False	Définit si les mises à jour sont déclenchées lors de l'arrêt de l'hôte.	`install_at_shutdown` = True
`install_delay` (par défaut `None`)	Configure un délai d'installation entre la publication dans le dépôt et l'installation.	`install_delay` = 15d
`allowed_severities` = None	Définit une liste de criticité qui sera automatiquement accépté durant un scan WAPT windows update. ex : *Important*, *Critical*, *Moderate*.	`allowed_severities` = Important

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]` :

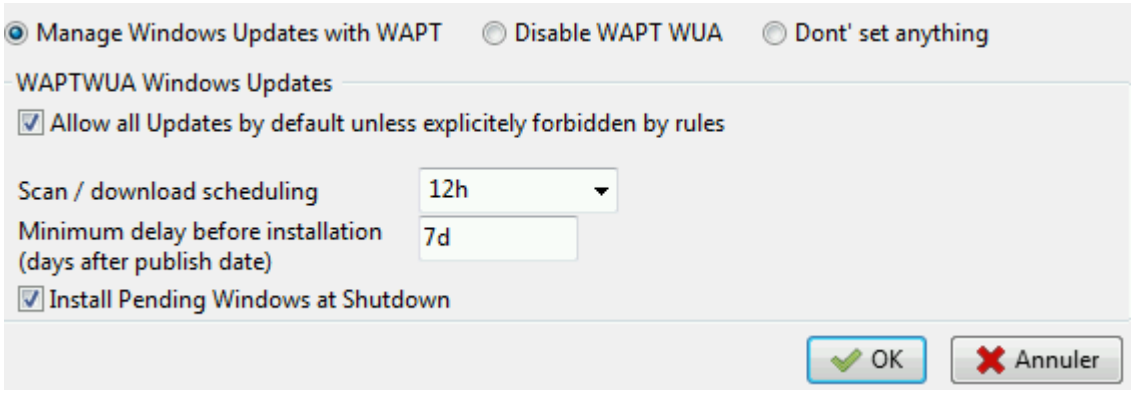


FIG. 2 – Options de menu pour l'agent de mise à jour Windows WAPT

```
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')
  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`.

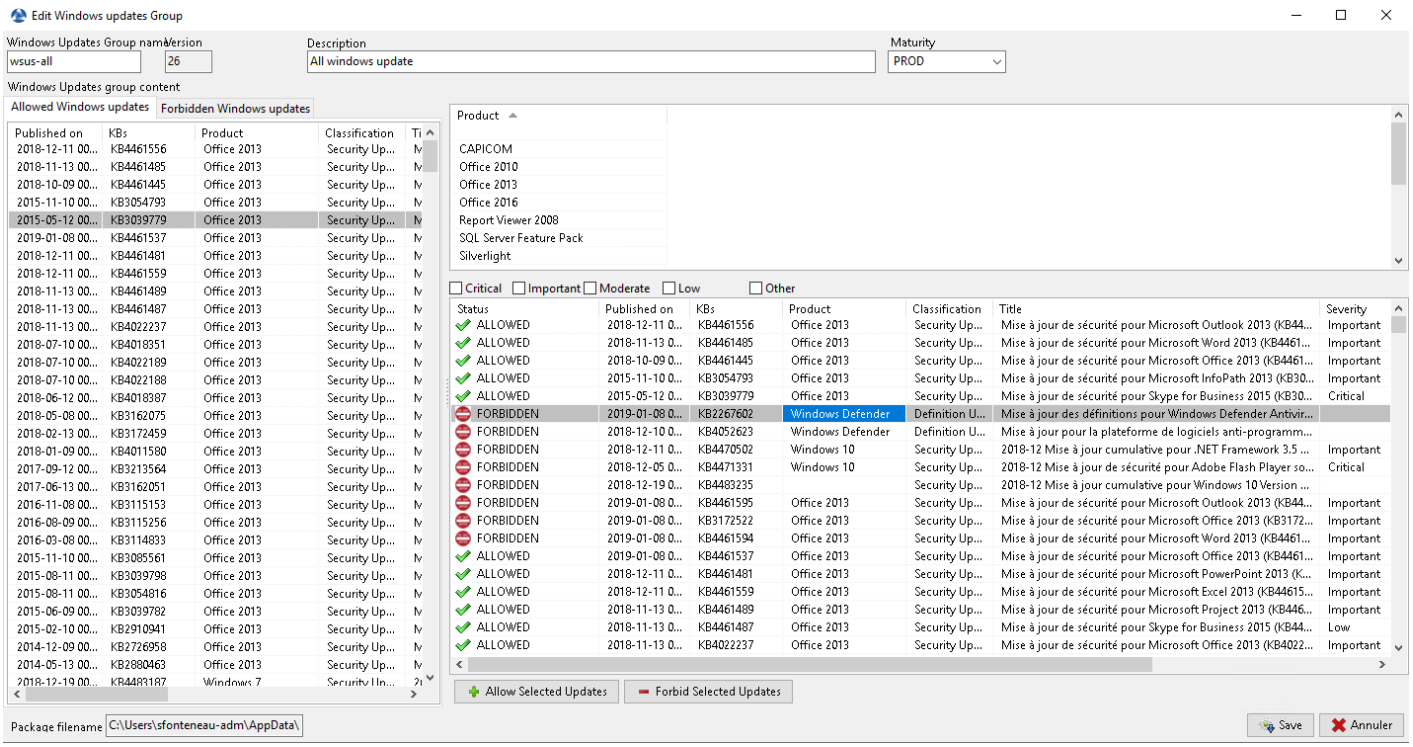


FIG. 3 – Création de paquets de regroupement de profils dans la console WAPT

— 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.

L'onglet liste des Windows Updates

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 :

— Criticté ;
— 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* .

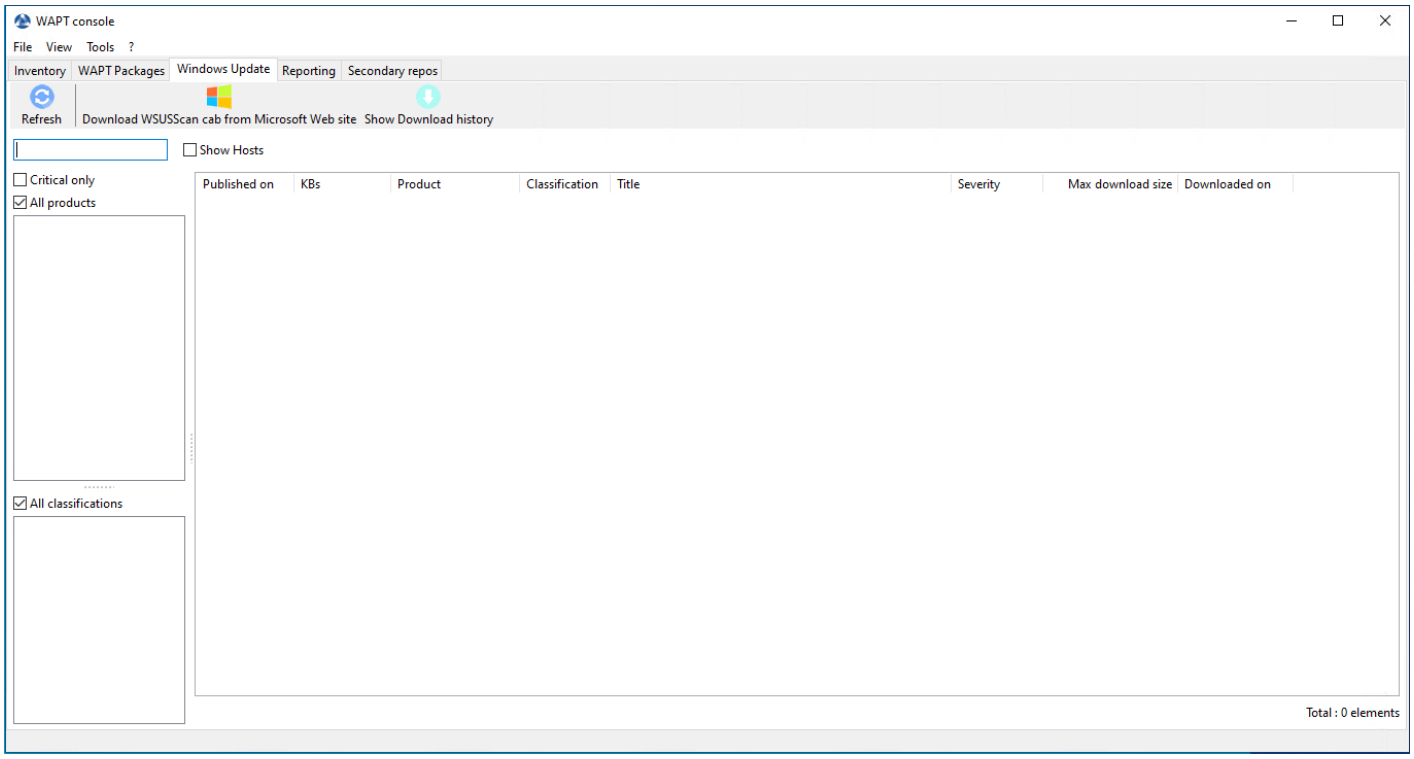


FIG. 4 – Les mises à jour Windows en attente affichées dans la console WAPT

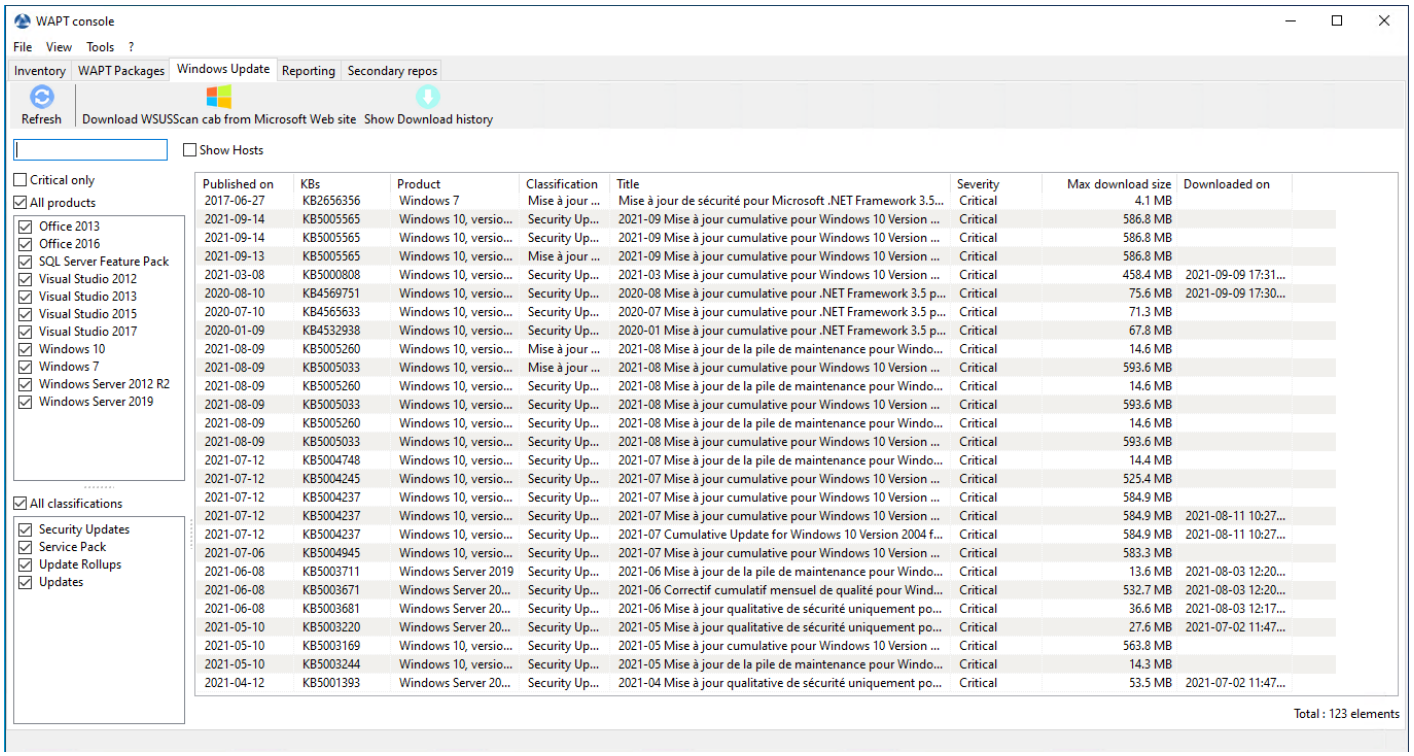


FIG. 5 – Les mises à jour Windows en attente affichées dans la console WAPT

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 l'hôte, elle est automatiquement supprimée sur le Server WAPT entre 2h30 et 3h30 tous les jours.

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 dans votre serveur WAPT.

Redémarrez le serveur WAPT

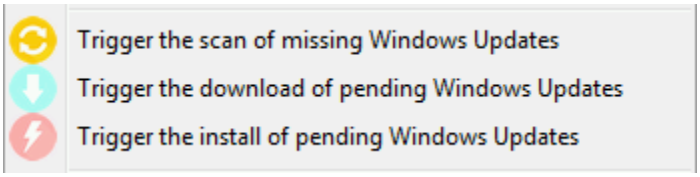
Il est possible de supprimer manuellement du serveur WAPT tout fichier Windows Update qui n'est plus nécessaire.

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.

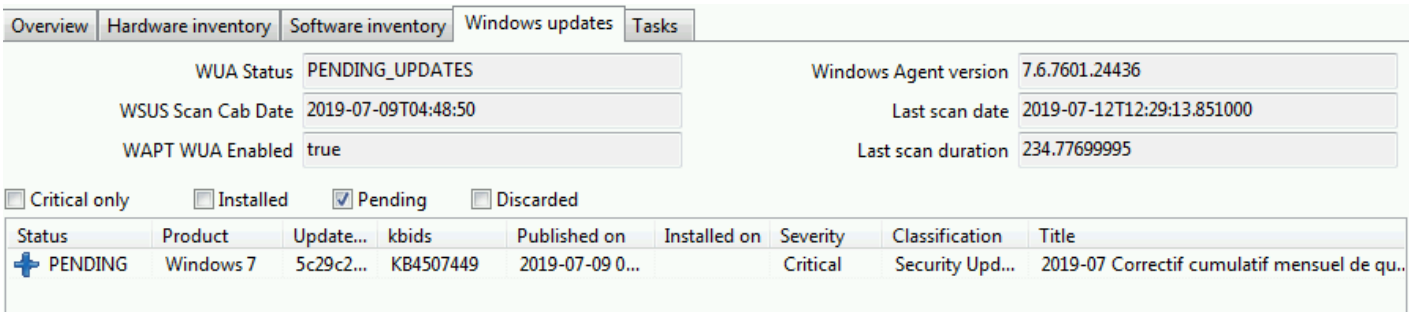


FIG. 6 – Les mises à jour Windows en attente affichées dans la console WAPT

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.

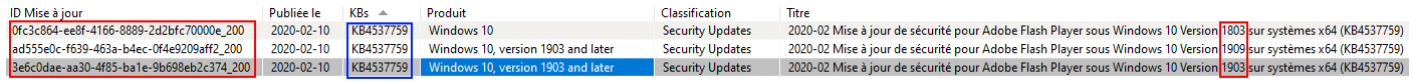


FIG. 7 – Liste montrant les KB en double dans la console WAPT

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():
  with EnsureWUAServRunning():
    run('wusa /uninstall /KB:4023057')
```

CHAPITRE 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 Donner l'état actuel de l'hôte au serveur.

Tableau 1 – 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.
2. Exécutez l'agent FusionInventory et extrayez le rapport dans un fichier `.xml`.

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.

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.

CHAPITRE 30

Utiliser des paquets profile 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`.

30.1.1 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.

Pour créer une nouvelle requête, cliquez sur *Reporting* → *Mode conception* → *Nouvelle requête*.

Indication :

- Pour renommer une requête, appuyez sur la touche `F2`.
- Dans l'encadré du haut, vous pouvez écrire votre requête SQL.

Pour éditer / modifier / Sauvegarder vos requêtes :

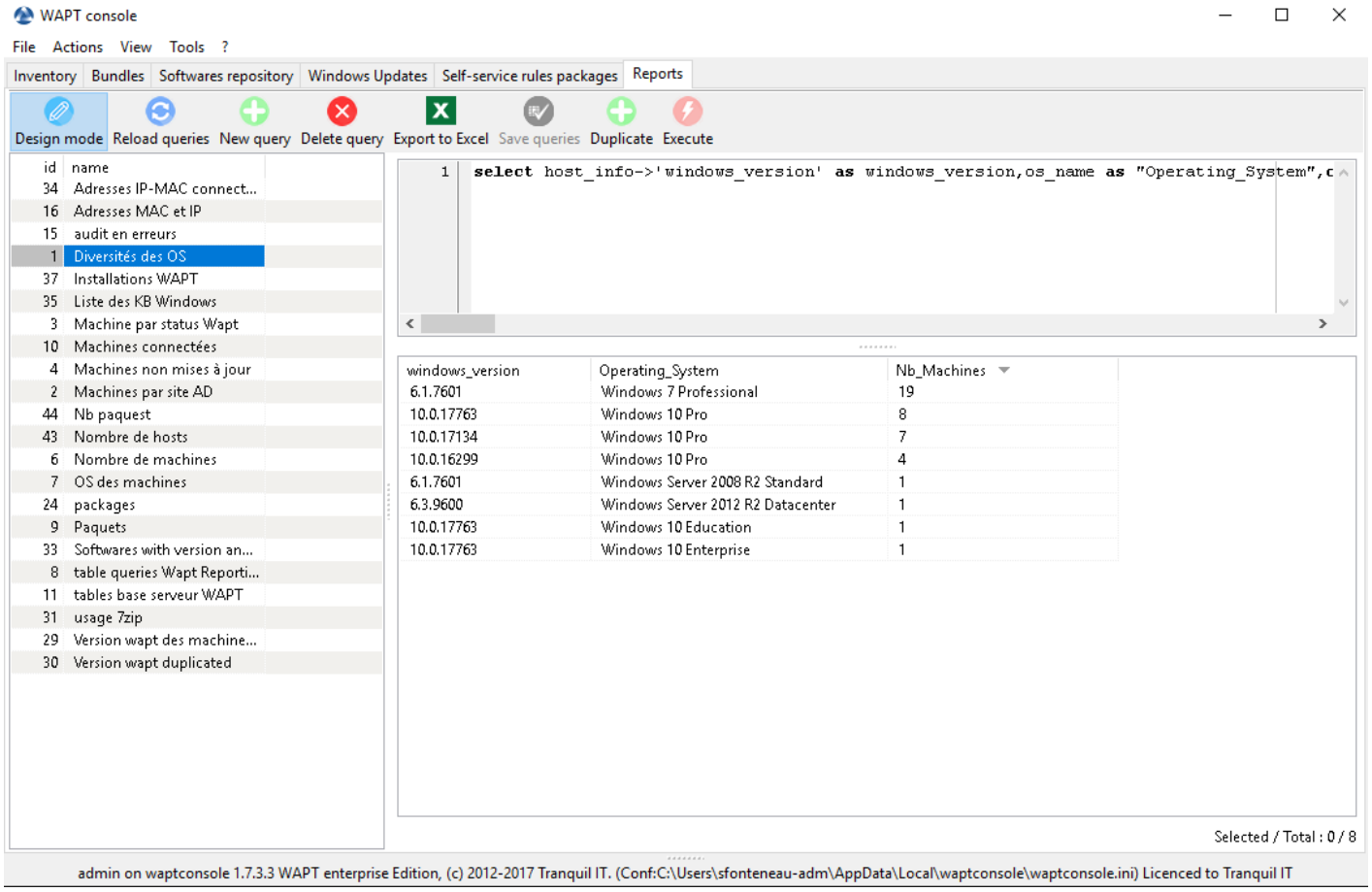


FIG. 1 – Conception d'un rapport de requête SQL dans la Console WAPT

— Le bouton *Recharger* est utilisé pour recharger les requêtes 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 *Exporter vers tableur* va exporter le résultat de votre requête dans un feuille de calcul.
— Le bouton *Enregistrer tout* va sauvegarder votre requête au serveur WAPT.
— Le bouton *Dupliquer* va dupliquer une requête existante pour éviter de repartir d'une requête vide.
— Le bouton *Exécuter* va exécuter la requête sélectionnée.

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.

30.1.2 Exemple de requêtes

Requêtes Ordinateur

Counting hosts

```
select count(*) as "number_of_hosts" from hosts
```

Listing computers

```
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
```

(suite sur la page suivante)

(suite de la page précédente)

```
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.listening_timestamp::timestamp,
h.connected_users from hostsoftwares s
left join hosts h on h.uuid=s.host_id
where s.key='WAPT_is1'
and h.listening_timestamp<'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
```

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
and host_status!='OK')
group by 1,2,3,4
```

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
```

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
```

(suite sur la page suivante)

(suite de la page précédente)

```
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.

30.1.3 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, nous proposons de standradiser le nom des logiciels avec WAPT.

— 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 combinaisonsde touches `shift-up/down`.
— Vous pouvez aussi marquer un logiciel comme *Mise à jour Windows* ou *Banni* (Appuyez sur `la barre espace` dans la colonne correspondante).

— Appuyez sur *Importer* pour charger les changements du serveur.
— Appuyez sur *Enregistrer* pour sauvegarder vos changements.

Vous pouvez maintenant lancer vos requêtes avec ce nom standardisé.

30.1.4 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.

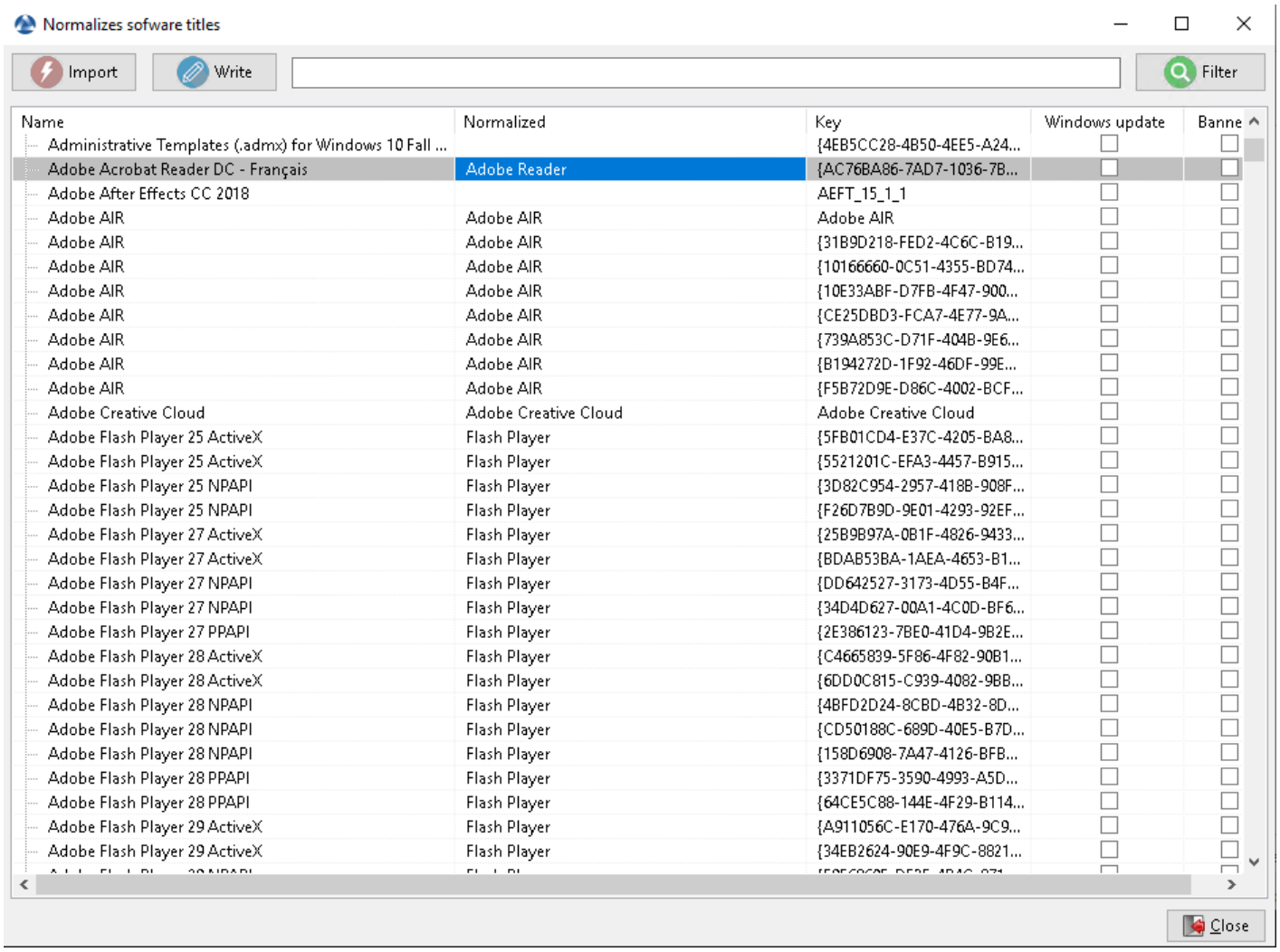


FIG. 2 – Normaliser le nom du logiciel

```
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';
```

CHAPITRE 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

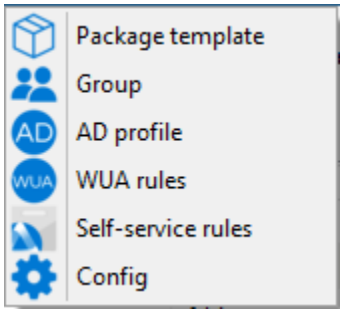
Discovery

Sur Discovery, créez un groupe *self-service* dans votre Active Directory et ajoutez vos utilisateurs. Ces utilisateurs et tous les *Local Administrator* ont accès à tous les paquets de votre dépôt.

Note : Il n'est pas possible de filtrer les paquets rendus accessibles à l'utilisateur.

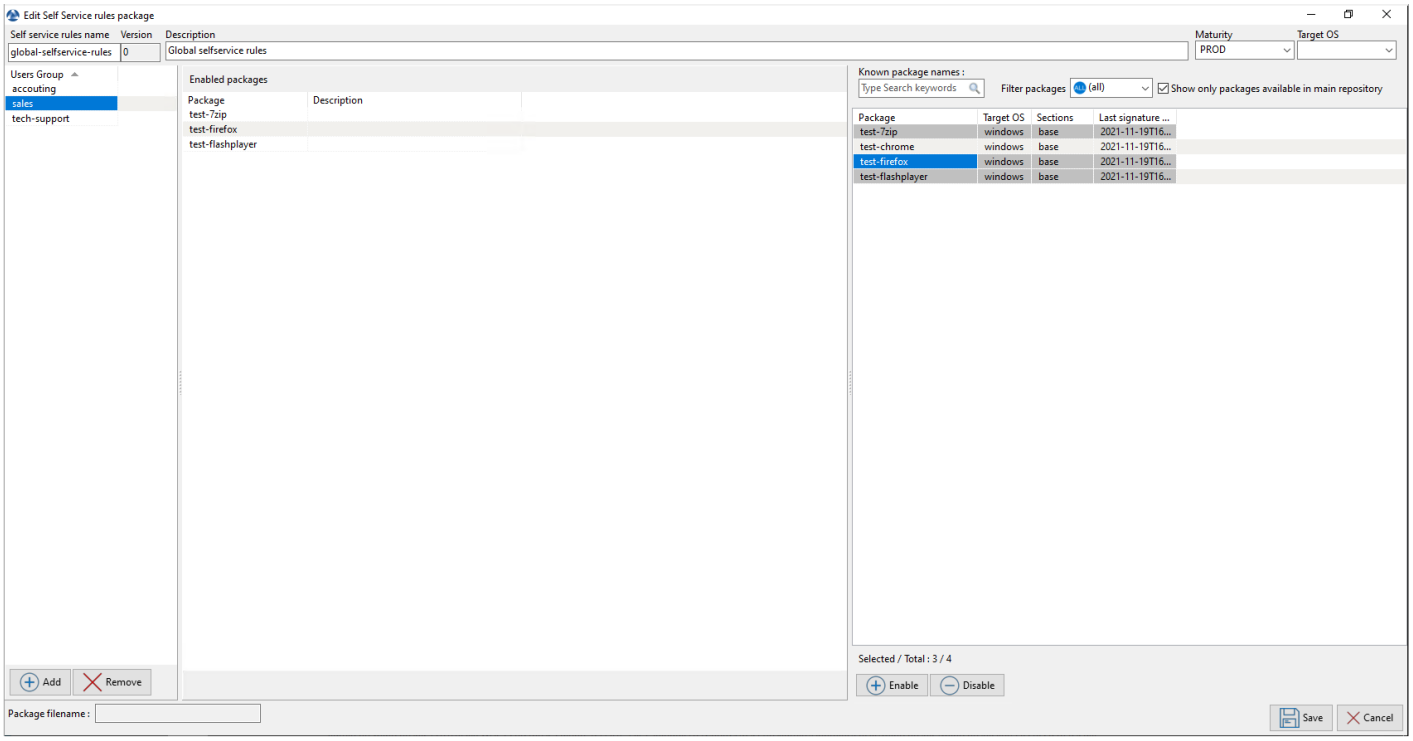
Enterprise

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). Le groupe doit avoir le même nom que le **groupe de sécurité des utilisateurs de l'Active Directory** ;
5. Faites glisser et déposez dans la colonne centrale les logiciels et les paquets de configuration autorisés pour ce groupe *de libre-service* ;
6. Ajoutez autant de groupes que vous le souhaitez dans le paquet ;
7. Sauvegardez le paquet et déployez 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.

- L'utilisateur peur avoir plus de détails sur chaque paquet avec l'icône + ;

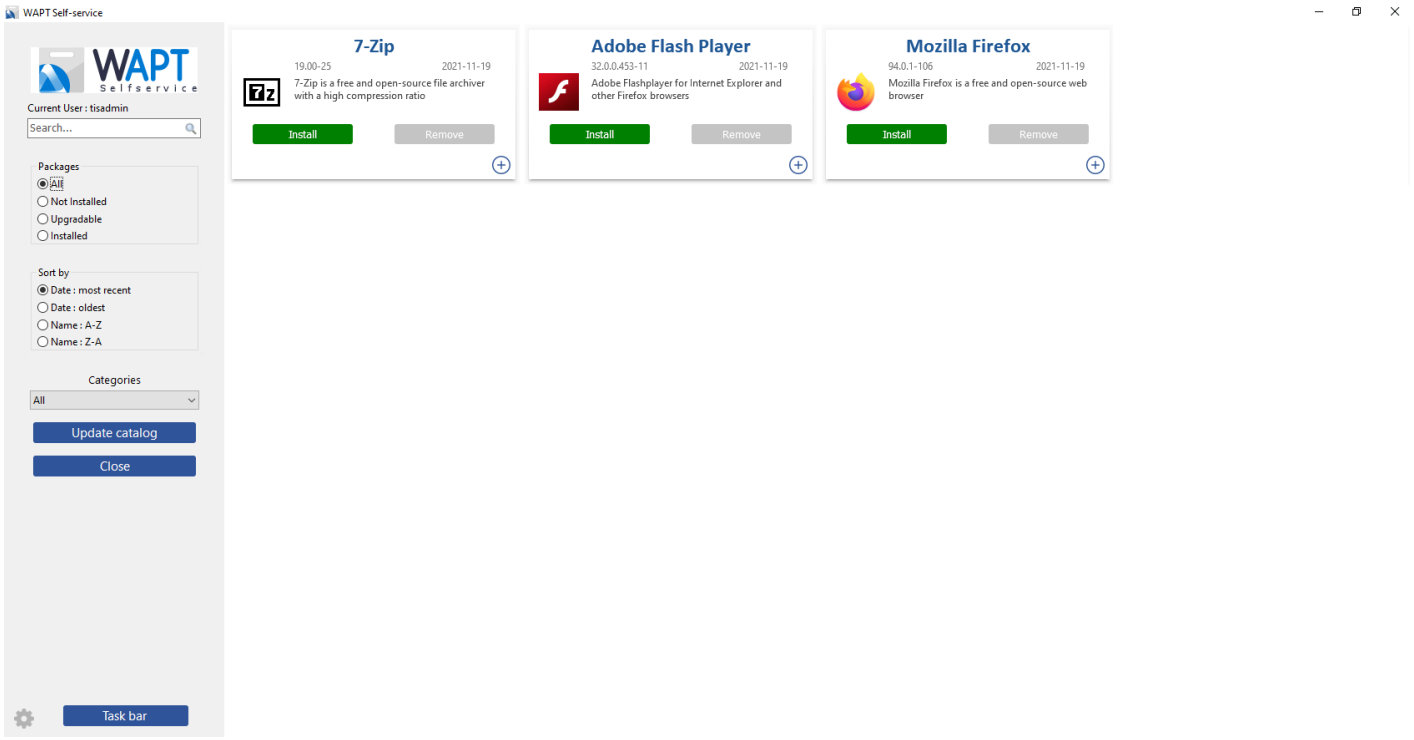
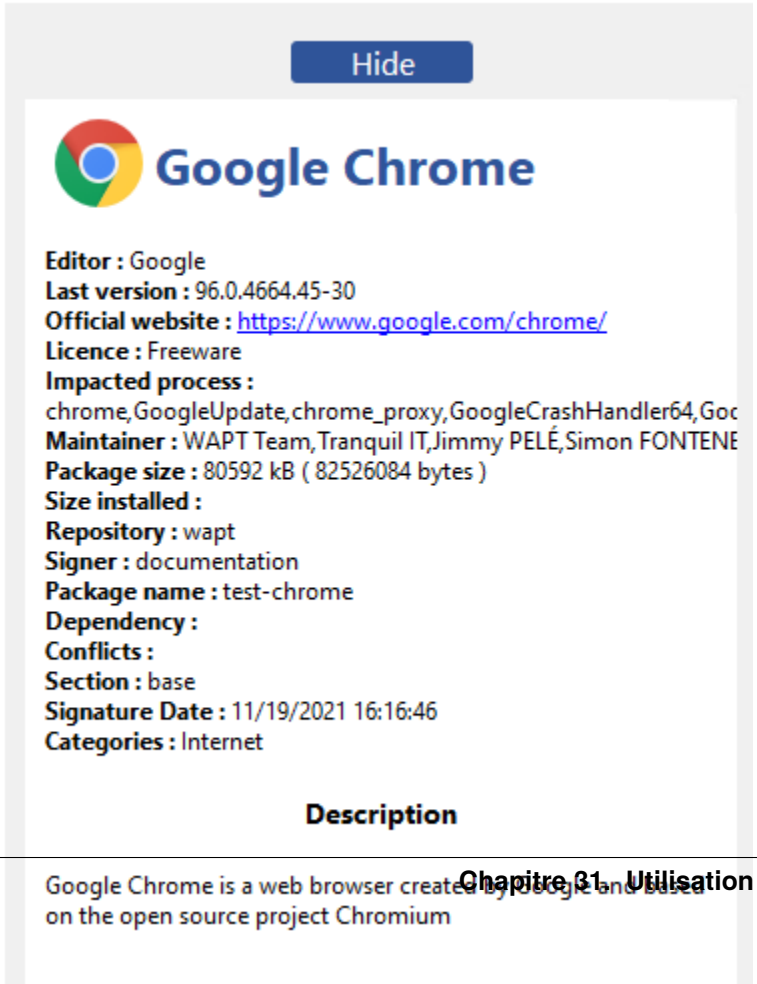
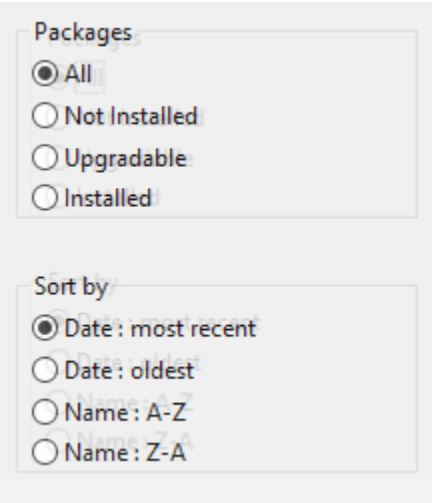


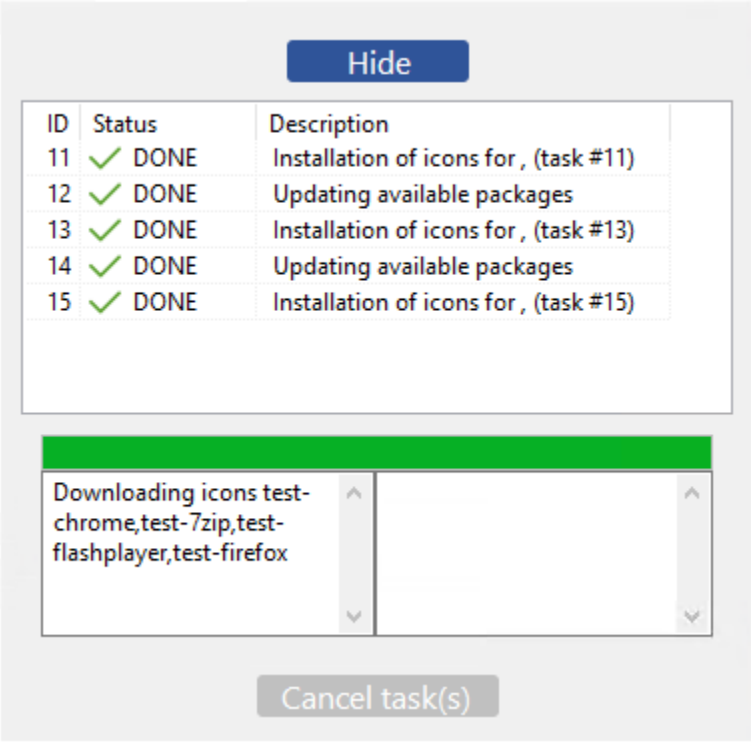
FIG. 1 – Fenêtre principale du libre-service WAPT



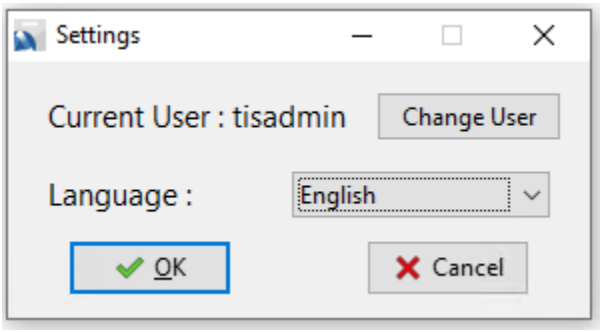
— Différents filtres sont disponibles pour l'utilisateur sur le panneau de gauche ;



— 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 ;

Il est possible de *ajouter d'autres catégories* aux paquets que vous développez.

31.5 Les configurations 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* peut 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 visible 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

CHAPITRE 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

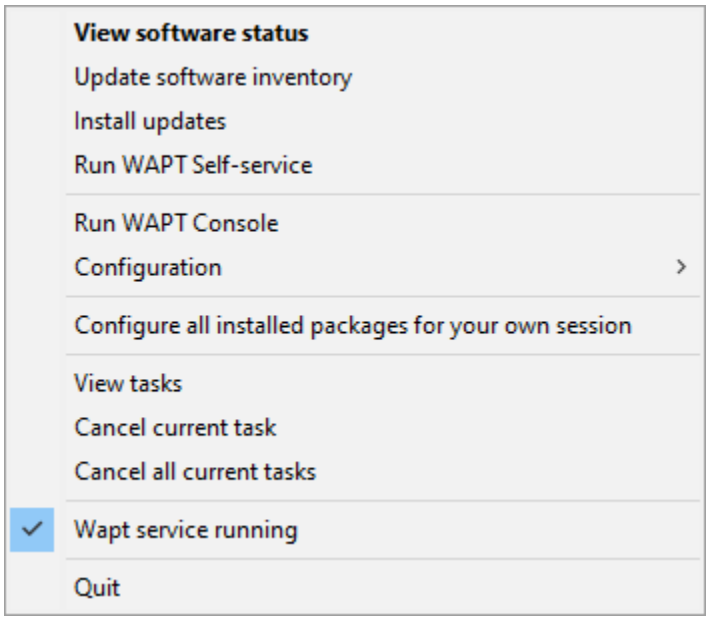


Tableau 1 – Liste des fonctionnalités de WAPTtray

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 *WAPTservice*
True	Ferme l'icone sans stopper le *WAPTservice* local

Fonctions de configuration

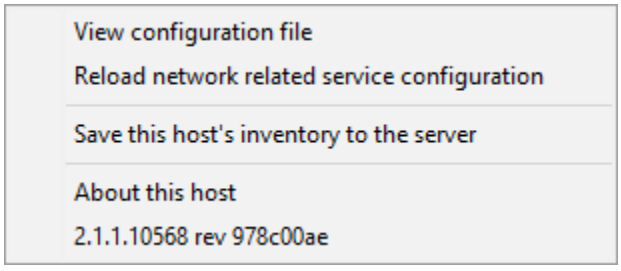


TABLEAU 2 – Liste des options de configuration de l'utilitaire WAPT System Tray

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 hôtes

32.2 Vidéo de démonstration

https://youtu.be/9iG36IeHuVc

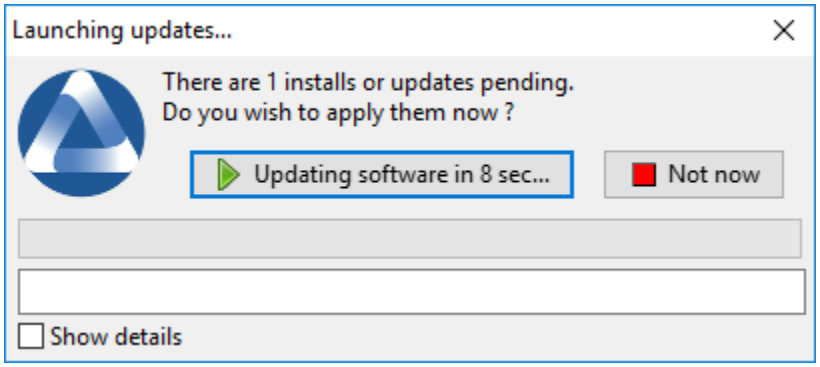
CHAPITRE 33

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 qu'un paquet spécifique priority, vous pouvez exécuter **waptexit** avec l'argument :

```
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

CHAPITRE 34

Personnaliser WAPT

Il est possible de personnaliser WAPT aux couleurs de votre société.

3 programmes sont personnalisables :
- l'utilitaire WAPT Exit ;
- Accès au self-service
- 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

Note : 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

Note : La taille recommandée pour le logo est 200X55 et le format est .png

Avertissement : 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

Note : La taille recommandée pour le logo est 200X55 et le format est .png

Avertissement : 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

Note : La taille recommandée pour le logo est 200X55 et le format est .png

Avertissement : S'il n'est pas défini, WAPT utilise dans l'ordre waptexit-logo.png, waptself-logo.png et enfin le logo WAPT par défaut.

CHAPITRE 35

Utiliser WAPT en ligne de commande

L'agent WAPT fournit un utilitaire d'interface de ligne de commande **`wapt-get`**.

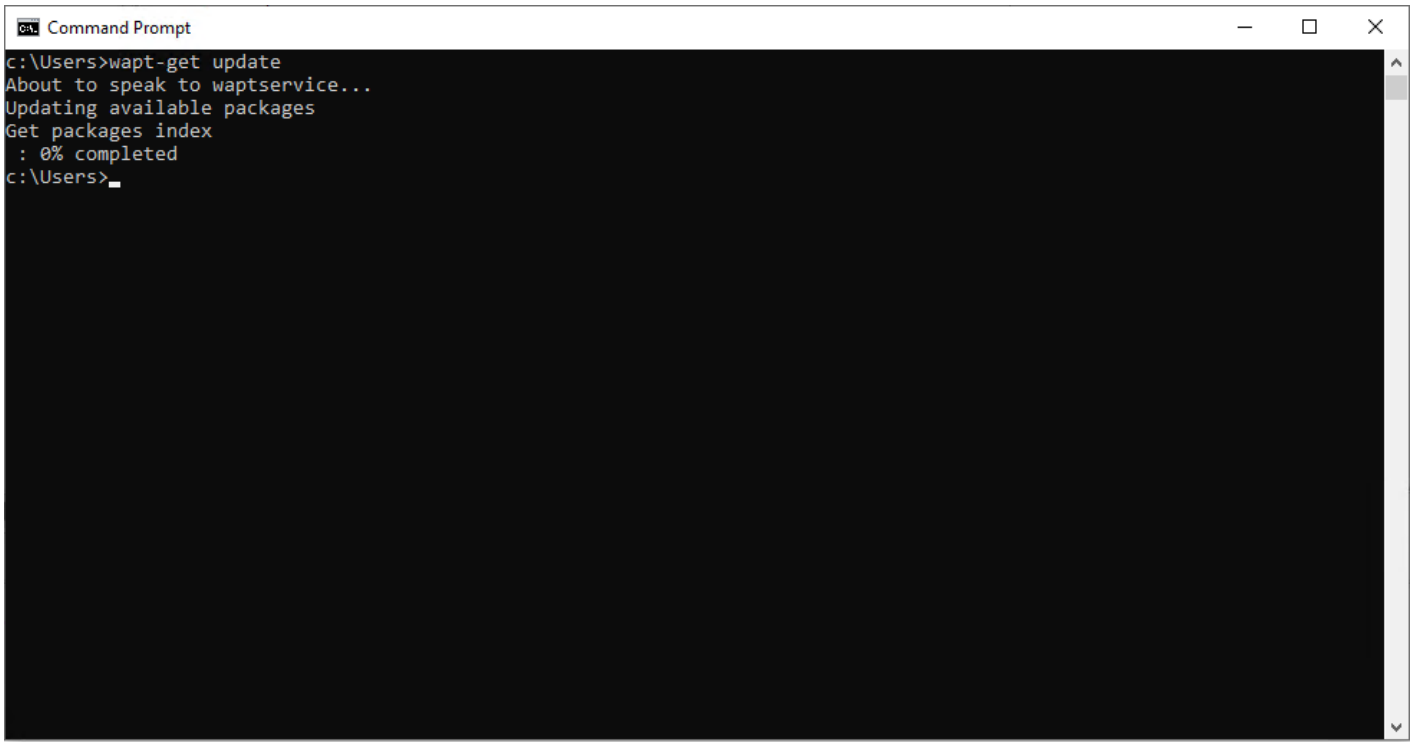


FIG. 1 – L'invite de commande Windows

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 : Toutes les commandes qui prennent un nom de package en paramètre peuvent également prendre le package_uuid unique du package en paramètre (**wapt-get install**, **wapt-get forget**, etc.) L'utilisation d'un GUID permet de spécifier un packaging 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.

La commande `wapt-get install tis-firefox` renvoie :

Note : Si le paquet n'a pas été téléchargé dans le cache, `install` va d'abord télécharger le paquet dans le cache, puis l'installer.

Attention : L'installation d'un paquet WAPT avec `install` n'ajoute pas le paquet comme dépendance à l'hôte.

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)
Installing tis-firefox(=94.0.1-106)
Installing: Firefox_Setup_94.0.1.exe
```

(suite sur la page suivante)

(suite de la page précédente)

```
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 téléchargeables pas 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` permet de lancer l'installation des paquets en attente de mise à jour ou en attente d'installation.

L'agent WAPT local télécharge si nécessaire les paquets WAPT dans son cache local puis les installe.

Indication : Il est recommandé de lancer la commande `wapt-get update` avant de lancer une commande `wapt-get upgrade` ;

Sans 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 un paquet.

La commande prend un argument. Cet argument est le nom paquet ou une liste de noms de paquets avec le préfixe du référentiel.

La commande `wapt-get remove tis-firefox` renvoie :

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 install <package name>` lance l'installation d'un paquet.

La commande prend un argument. Cet argument est le nom paquet ou une liste de noms de paquets avec le préfixe du référentiel.

Attention : La suppression d'un paquet WAPT avec `remove` ne supprime pas la dépendance du paquet sur l'hôte.

La commande `wapt-get install tis-firefox` 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 edit <nom du paquet>` télécharge et édite un paquet WAPT depuis le dépôt du serveur.

La commande prend un argument. Cet argument est le nom paquet ou une liste de noms de paquets avec le préfixe du référentiel.

Attention : Oublier le packaging ne désinstalle pas le packaging.

La commande `wapt-get clean` 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 edit <nom du paquet>` télécharge et édite un paquet WAPT depuis le dépôt du serveur.

La commande prend un argument. Cet argument est le nom paquet ou une liste de noms de paquets avec le préfixe du référentiel.

Si l'argument est défini, la commande `wapt-get audit [<nom du package>]` exécute la fonction d'audit du packaging.

La commande `wapt-get list` liste les paquets WAPT qui sont installés sur l'ordinateur.

La commande `wapt-get install 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 sont disponibles sur le référentiel, 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 : Il est possible qu'un message d'avertissement soit affiché avec cette commande, par exemple :

```
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>]
```

Ceci est normal si votre certificat n'est pas fiable. Seul le fichier `control` est disponible pour `wapt-get show` et non pour tous les paquets.

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 des listes de paramètres qui seraient passés 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 référentiel.

La commande `wapt-get show 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 <nom du paquet>` affiche les informations stockées dans le fichier d'index `Packages`.

La commande prend un argument. Cet argument est le nom paquet ou une liste de noms de paquets avec le préfixe du référentiel.

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 disponibles pour l'hôte qui exécute la commande.
>
> 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 pour être recherchée dans le nom et la description du paquet.

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 situé dans `C:\Program Files\wapt\cache`.

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 `C:\Program Files (x86)\wapt\cache`.

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 /␣
```

(suite sur la page suivante)

(suite de la page précédente)

```
↪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)
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` liste les paquets WAPT qui sont installés sur l'ordinateur.

La commande `wapt-get 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 list` 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 clean

La commande `wapt-get clean` supprime les paquets du dossier `C:\Program Files (x86)\wapt\cache`.

La commande `wapt-get clean` est lancée après chaque mise à niveau 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
```

35.2.2 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
```

35.2.3 wapt-get add-upgrade-shutdown - wapt-get remove-upgrade-shutdown

Ces 2 commandes modifient ce fichier : `C:WindowsSystem32GroupPolicyMachineScriptsscripts.ini`.

— 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]
```

35.2.4 wapt-get register

La commande `wapt-get register [<description>]` rapporte l'inventaire matériel et logiciel de l'ordinateur au serveur d'inventaire WAPT.

Indication : Vous pouvez passer une description comme argument à la commande `register`, cette description sera affichée dans la console WAPT dans la colonne *description*.

Vous pouvez profiter de WAPT pour améliorer votre gestion informatique en affectant un nom d'utilisateur ou un numéro de série d'ordinateur comme descriptions pour vos hôtes.

Note : Si l'hôte est déjà enregistré, la nouvelle exécution avec une description met à jour les informations enregistrées.

Il n'est pas nécessaire de définir une description pour enregistrer l'hôte avec la ligne de commande.

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.wapt-get inventory
```

35.2.5 wapt-get register

La commande `wapt-get register [<description>]` rapporte l'inventaire matériel et logiciel de l'ordinateur au serveur d'inventaire 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
```

(suite sur la page suivante)

(suite de la page précédente)

```
Password:
Host correctly unregistered against server https://srvwapt.mydomain.lan.
```

35.2.6 wapt-get inventory

La commande `wapt-get inventory` affiche toutes les informations relatives à l'inventaire local 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",
    "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"
    ],
```

(suite sur la page suivante)

(suite de la page précédente)

```
    "connected_ips": [
    "192.168.0.1",
    "fe80::2437:567f:79c8:f964"
    ],
    "mac": [
    "a2:fc:1d:6e:8d:e6"
    ],
...
```

35.2.7 wapt-get update-status

La commande `wapt-get update-status` renvoie le statut local au serveur d'inventaire 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}
```

35.2.8 wapt-get setlocalpassword

La commande `wapt-get setlocalpassword` permet de définir un mot de passe local pour les installations de paquets WAPT.

La commande `wapt-get setlocalpassword` renvoie :

```
Local password:
Confirm password:
Local auth password set successfully
```

35.2.9 wapt-get reset-uuid

La commande `wapt-get reset-uuid` récupère l'hôte *UUID* du BIOS et le renvoie au serveur d'inventaire 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
```

35.2.10 wapt-get generate-uuid

La commande `wapt-get generate-uuid` crée un nouvel hôte *UUID* et le renvoie au serveur d'inventaire WAPT.

À utiliser si vous avez un *bios UUID bug*.

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 mettent un RND- devant.

35.2.11 wapt-get get-server-certificate

La commande `wapt-get get-server-certificate` télécharge le certificat SSL du serveur WAPT pour utiliser HTTPS pour communiquer avec le serveur WAPT.

Le certificat téléchargé est stocké dans `C:\Program Files(x86)\wapt\server`.

La commande `wapt-get get-server-certificate` renvoie :

```
Server certificate written to C:\Program Files (x86)\wapt\ssl\server\srvwapt.mydomain.lan.crt
```

35.2.12 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.

Il est utilisé 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
```

35.2.13 wapt-get upgrade

La commande `wapt-get check-upgrades` montre l'état des dernières mises à jour/mises à niveau sur l'hôte.

La commande `wapt-get upgrade` 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": []
  }
}
```

35.2.14 wapt-get duplicate

La commande `wapt-get list` liste les paquets WAPT qui sont installés sur l'ordinateur.

La commande `wapt-get clean` 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
```

(suite sur la page suivante)

(suite de la page précédente)

```
↪"],"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+dlOok9E8i405UTNE7oSVYsF\n90/
↪6S3C4twIDAQABo2QwYjAPBgNVHRMBAf8EBTADAQH/MCAGA1UdJQEB/wQWMBQG\
↪nCCsGAQUFBwMDBggrBgEFBQcDAjAdBgNVHQ4EFgQUpRT6Co2uoWZMCwP7FKiF73+j\nfAEwDgYDVR0PAQH/
↪BAQDAgHWMA0GCSqGSIb3DQEBCwUAA4IBAQAdXX5IkpuH/Gek\nPPHC4KvE/
↪6GsU0kqLI1w5ML5pbF1zyCCL0nm4f8w2JJIJ2Ycdb4QVD27kJqgZcH1\nniYQ3RCIh6aasS8qpCOf90KkpvKMJiyk/
↪ra7I6NSgPut4ErkoxUWocgF6SNFEjwB\naqUZY//Hkoqk2dXqdujLVGJfBpX95ZJ9PmFNLfsyUsvu1WcFMb0En0EUO74Mq4M3\
↪nKo2S86G9pEDKooaN5Vq19biReOwQYpX1YlSLtrxFx8AM87auQgaD8EWSdA1q2ycN\
↪n8ZnmXGxAhDv8hmE2Fv0x0t3hzYXxxcv1ZjYWRHlMUl/buWQQ35u9MFkjh7YZlTlM\nb9wjtN+W\n-----END CERTIFICATE-
↪----\n","signature":
↪"J7DZ+mja7zGghYFCDKh8WIxzzdhKPeoNswWjnKZziT+ddpoRdg45kZz4E8PxMIUzhTI8WIxzzdhKPeoNswrICpQ8t5kepzovZpoONwjgOQ
↪de18bEgSSlgjXgE/wr2ZfclsRsRRfsRbGSterRKQcthNDrFlf8RjH5cpDnDvMJ+qJtTsqxA13/WT2NS2uNWZI93si/
↪9mowWY8MdT/PZjosciCqijbq4oa+/FrPsALhUOtcGE9JylwknszUD5Ayfh+9sNLLxsG6eT0JlnNgf4nx9mXAu4GBg==",
↪"server_uuid":"82295c4d-4944-11ec-bac6-a25b5d7da3d5"}
Login to server api/v3/licences
Waptserver https://srvwapt.mydomain.lan Admin User () :admin
Waptserver Password: ************
Licence properly activated on server
```

35.2.15 wapt-get clean

La commande `wapt-get list` liste les paquets WAPT qui sont installés sur l'ordinateur.

La commande `wapt-get clean` 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
```

(suite sur la page suivante)

(suite de la page précédente)

```
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
```

35.2.16 wapt-get download

La commande `wapt-get inventory` affiche toutes les informations relatives à l'inventaire local au format JSON.

La commande `wapt-get list` 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.

Elle est définie dans la fonction *session-setup* du fichier `setup.py`.

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, non sensible à la casse, pour rechercher un mot spécifique : `wapt-get list-registry <keyword>`.

Les informations retournées sont :

Information	Définition	Disponible sur Windows	Disponible sur Linux	Disponible sur Mac OS
Désinstaller la clé	Désinstaller l'identifiant de clé dans le registre	✅	❌	❌
Les logiciels	Nom du logiciel dans le registre	✅	✅	✅
Version	Version du logiciel dans le registre	✅	✅	✅
Désinstaller la chaîne	Chaîne de désinstallation du logiciel dans le registre	✅	❌	❌

Note :

- Sous Windows, ces informations sont collectées par WAPT à partir du registre en deux localisations :
  - `OrdinateurHKEY_LOCAL_MACHINESOFTWAREMicrosoftWindowsCurrentVersionUninstall`
  - `MachineHKEY_LOCAL_MACHINESOFTWAREWOW6432NodeMicrosoftWindowsCurrentVersionUninstall`
- Sous Linux, ces informations sont collectées par WAPT dans `Applications`
- Sous Mac OS, ces informations sont collectées par WAPT dans `/var/lib/dpkg/info/`

La commande `wapt-get list-registry firefox` renvoie (sous Windows) :

La sortie de **`wapt-get list-registry`** est un tableau listant les *clés de désinstallation* pour chaque logiciel correspondant au terme recherché.

```
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 une plateforme 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 <chemin-installateur> [<nompaquet> [<nom-dossier source>]]` permet de créer un modèle de paquet à partir d'un binaire.

Utilisez ces arguments :

Arguments	Définition	Valeur par défaut
chemin de l'installateur	Chemin d'installation du binaire	Pas de valeur par défaut, c'est nécessaire
nompaquet	Nom du paquet	S'il n'est pas défini, le nom du paquet est créé comme ceci `prefix-binaire-nom_paquet-version-wapt`
nom du répertoire source	Répertoire pour enregistrer les fichiers sources	Si elle n'est pas définie, c'est la valeur de `default_sources_root` dans `waptconsole.ini`

La commande 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 le packaging de l'hôte est automatiquement créé 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.

Utilisez ces arguments :

Arguments	Définition	Valeur par défaut
nom d'hôte	Nom de la machine utilisée pour le nom du paquet	Si aucun n'est donné, utiliser le FQDN
package	Liste des paquets nécessaires sur l'hôte.	S'il n'est pas défini, aucun paquet n'est ajouté comme dépendance
répertoire	Répertoire pour enregistrer les fichiers sources	S'il n'est pas défini, c'est `C:\waptdev`

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 pour créer un packaging.

La commande `wapt-get make-group-template <nom du groupe>` crée un paquet de groupe WAPT 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 upload-package C:waptdev is-tightvnc.wapt` 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
```

(suite sur la page suivante)

(suite de la page précédente)

```
1 packages successfully built
0 packages failed

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 et le préfixe du packaging, il est défini 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 : `sign-package` ne renomme pas le paquet WAPT avec le préfixe choisi de l'*Organisation*.

La commande `wapt-get sign-package c:waptdevtis-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 `build-upload`, le numéro de version du packaging WAPT est incrémenté avant le téléchargement, afin d'éviter de devoir modifier manuellement le fichier `control`.

La commande `wapt-get -i build-upload 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
```

(suite sur la page suivante)

(suite de la page précédente)

```
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 : 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 <package source> <package new_duplicate>** duplique un package téléchargé depuis le référentiel et l'ouvre en tant que projet **PyScripter**.

Avertissement : N'utilisez pas cette commande pour dupliquer un packaging d'hôte.

Utilisez ces arguments :

Arguments	Définition	Requis
Version du paquet	Chemin du répertoire du packaging ou du package compilé dans `.wapt`	✅
nompaquet	Nom du paquet	✅
version	Change la version du packaging dans le fichier `control`. S'il n'est pas défini, la même version est dupliquée.	❌
répertoire	Chemin vers le répertoire cible du packaging dupliqué. S'il n'est pas défini, utilisez le même que celui du packaging 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
```

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.

35.4.10 wapt-get edit

Avertissement : Cette méthode ne doit être utilisée que si vous ne pouvez pas utiliser la console pour créer un packaging.

La commande `wapt-get edit <nom du paquet>` télécharge et édite un paquet WAPT depuis le dépôt du serveur.

La commande prend un argument. Cet argument est le nom paquet ou une liste de noms de paquets avec le préfixe du référentiel.

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.

35.4.11 wapt-get edit-host

Avertissement : Cette méthode ne doit être utilisée que si vous ne pouvez pas utiliser la console pour créer un packaging.

La commande `wapt-get edit-host <host FQDN>` édite un paquet WAPT *host*.

La commande `wapt-get edit tis-firefox` 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
```

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.

35.4.12 wapt-get update-package-sources

La commande `wapt-get upload-package <chemin vers le paquet>` télécharge un paquet sur le dépôt principal de WAPT.

La commande `wapt-get remove 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
```

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.

35.5 Utilisation des lignes de commande pour la gestion de WaptWUA

35.5.1 wapt-get clean

Le `wapt-get waptwua-scan` analyse l'état de Windows par rapport aux règles actuelles et envoie le résultat au serveur.

La commande `wapt-get clean` 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 Windows par rapport aux règles actuelles, télécharge les kb manquants et envoie le résultat au serveur.

La commande `wapt-get download-upgrade` 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:
{
```

(suite sur la page suivante)

(suite de la page précédente)

```
"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:
{
"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

Le `wapt-get waptwua-install` installe les mises à jour en attente.

La commande `wapt-get list` 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,
```

(suite sur la page suivante)

(suite de la page précédente)

```
"allowed_classifications": null,
"allowed_severities": null,
"allowed_updates": null,
"forbidden_updates": null,
"allowed_kbs": null,
"forbidden_kbs": null,
"default_allow": false
}
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 update-status` renvoie :

La commande `wapt-get update-status` renvoie :

```
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
```

35.5.5 wapt-get restart-waptservice

The `wapt-get restart-waptservice` restart waptservice on Windows, Linux and MacOS.

35.6 Utilisation des lignes de commande pour l'interaction avec les utilisateurs

35.6.1 wapt-get upgrade

La commande `wapt-get propose-upgrade` lance une proposition de mise à niveau en lançant waptexit dans les sessions ouvertes.

La commande `wapt-get 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.

Tableau 1 – Liste des options disponibles

Option	Description	Valeur par défaut
`--CommonName`	Nom d'affichage du certificat	/
`--CommonName64`	Nom d'affichage du certificat, encodé en base64 (si accents, espaces etc.)	/
`--CodeSigning` **|enterprise_feature||**	Si le certificat / la paire de clés sera autorisé(e) à signer des paquets de logiciels	`0`
`--CA` **|enterprise_feature||**	Si ce certificat / paire de clés peut être utilisé pour signer d'autres certificats (Autorité de certification principale ou intermédiaire)	`0`
`--ClientAuth`	Correspond à une propriété (utilisation) du certificat	- pour non installé
`--PrivateKeyPassword`	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	Mot de passe pour déverrouiller la clé, codé en base64 (si accents, espaces etc.)	Mot de passe généré aléatoirement
`--NoPrivateKeyPassword`	Si la clé privée n'est pas chiffrée si `--PrivateKeyPassword` ou `--PrivateKeyPassword64` ne sont pas utilisés	Vide
`-F`	Forcer l'écrasement du certificat existant	/
`--Pays`	Nom du pays du titulaire du certificat à enregistrer dans le certificat.	/
`--Localité`	Nom de la ville du titulaire du certificat à inscrire dans le certificat.	/
`--Organisation`	Nom de l'organisation du titulaire du certificat à enregistrer dans le certificat.	/
`--OrgUnit`	Nom de l'unité d'organisation (service) du titulaire du certificat à enregistrer dans le certificat.	/
`--Email`	Adresse e-mail du détenteur du certificat à enregistrer dans le certificat	/
`--CAKeyFilename`	Chemin vers la clé (`.pem`) d'une autorité de certification	Paramètre `default_ca_key_path` dans `waptconsole.ini`
`--CACertFilename`	Chemin vers le certificat (`.crt`) d'une autorité de certification	Paramètre `default_ca_cert_path` dans `waptconsole.ini`
`--CAKeyPassword`	Mot de passe pour déverrouiller la clé de l'Autorité de Certification	/
`-NoCAKeyPassword`	Si la clé de l'autorité de certification n'est pas chiffrée	/
`--BaseDir`	Dossier où la clé privée et le certificat public seront stockés.	Répertoire `personal_certificate_path` dans `waptconsole.ini`
`-EnrollNewCert`	Copiez le certificat dans `waptssl`	/
`-SetAsDefaultPersonalCert`	Le chemin du certificat est attribué à `personal_certificate_path` dans `waptconsole.ini`	/

La commande `wapt-get clean` renvoie :

```
Using config file C:\Users\Administrator\AppData\Local\waptconsole\waptconsole.ini
```

(suite sur la page suivante)

(suite de la page précédente)

```
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 `C:UsersAdministratorAppDataLocalwaptconsolewaptconsole.ini`, 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

Le `wapt-get build-waptagent [</ConfigFilename>]` compile et télécharge un package **`waptagent.exe`** et **`waptupgrade.exe`** en utilisant le paramètre `/ConfigFilename` du fichier ini. Pour le fichier ini, utilisez la syntaxe *wapt-get.ini*.

Note : Par défaut, utilisez la configuration `waptconsole.ini`.

La commande `wapt-get update` 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 is-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 ne concerne que les dépôts Windows

La commande `wapt-get scan-packages <directory>` reconstruit un fichier `Packages` pour le dépôt de packages http.

La commande `wapt-get update-status` 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 est seulement pour les dépôts Linux

La commande `wapt-scanpackages <directory>` reconstruit un fichier `Packages` pour le dépôt de packages http.

La commande `wapt-get sources tis-firefox` ne renvoie rien ;

35.8.4 Re-signature des paquets sur le serveur à l'aide d'une ligne de commande

Indication : Ces commandes ne sont disponibles que sur les serveurs Linux

Danger : 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/
```

Avertissement : Si l'erreur **Access violation** apparaît, c'est que le paquet est trop gros.

Éditez le paquet et suivez *cette procédure*.

Indication : Utilisez cette méthode si la resignature à partir de la méthode de la console WAPT n'aboutit pas.

Attention : Supprimez vos fichiers `.crt` et `.pem` de `/tmp/` sur le serveur WAPT.

35.9 Utilisation de lignes de commande spéciales avec WAPT

Option	Définition
`--version`	Afficher le numéro de version du programme et c
`-h | --help`	Afficher le message d'aide et quitter
`-c CONFIG | --config=CONFIG`	Chemin vers un autre fichier comme `wapt-get.`
`-l LOGLEVEL | --loglevel=LOGLEVEL`	Niveau des fichiers journaux suivant cette liste :
`-D | --direct`	N'utilisez pas le service http pour les mises à jou
`-S | --services`	Demander un utilisateur de Waptservice
`-u | --update-packages`	Exécuter `wapt-get update` avant la ligne de co
`-f | --force`	Forcez la ligne de commande
`-p PARAMS | --params=PARAMS`	Configurer les paramètres comme un objet JSon
`-r WAPT_URL | -repo WAPT_URL | --repository=WAPT_URL`	Remplace l'URL du dépôt principal de wapt à pa
`-y | --hide`	Utilisation des fonctions les plus courantes dans
`-F FILTRE_ON_HOST_CAP | --use-host-caps=FILTRE_ON_HOST_CAP`	Filtrer les packages en fonction des capacités act
`-i | --inc-release`	Augmenter le numéro de version lors de la const
`-a UPDATE_SERVER_STATUS | --update-server-status=UPDATE_SERVER_STATUS`	Envoyer l'état mis à jour de l'hôte (soft, package
`keep-signature-date`	Conserve la date de signature du packaging actue

suite sur la page suivante

Tableau 2 – suite de la page précédente

Option	Définition
-s SECTION_FILTER | --sections=SECTION_FILTER	Ajout d'un filtre **section**à la recherche wapt
-o REDIRECT_OUTPUT | --output=REDIRECT_OUTPUT	Rediriger les sorties vers un fichier donné .ini
-j | -json	Passage à la sortie json pour les scripts
-e ENCODING | --encoding=ENCODING	Changer l'encodage des caractères pour la sortie
-x EXCLUDES | --excludes=EXCLUDES	Liste de fichiers ou de répertoires séparés par des
-k PERSONAL_CERTIFICATE_PATH | --certificate=PERSONAL_CERTIFICATE_PATH	Chemin vers le certificat PEM X509 pour signer
-w PRIVATE_KEY_PASSWD | --private-key-passwd=PRIVATE_KEY_PASSWD	Chemin d'accès au fichier contenant le mot de pa
-U USER | --user=USER	Pour définir un utilisateur interactif
-g USERGROUPS | --usergroups=USERGROUPS	Groupes de l'utilisateur final sous forme de table
-t MAX_TTL | --maxttl=MAX_TTL	durée axiale d'exécution en minutes du processu
-L LANGUAGE | --language=LANGUAGE	Langue prioritaire pour l'installation, exemple : f
-m MD | --message-digest=MD	Message digest type pour wapt-get sign-pac
-n | --newest-only	Renvoyer uniquement la version la plus récente d
--locales=LOCALES	Remplace le filtre local des packages pour la recl
--maturité=MATURITÉ	Définir/modifier la maturité du packaging lors de
--pin-server-cert	Lors de l'enregistrement, épinglez le certificat du
--wapt-server-url=SET_WAPTSERVER_URL	Lorsque le paramètre wapt_server n'est pas dé
--wapt-repo-url=SET_WAPTREPO_URL	Lorsque le paramètre repo_url n'est pas défini,
--wapt-server-user=WAPT_SERVER_USER	Définit l'utilisateur chargé de télécharger les pac
--wapt-server-passwd=WAPT_SERVER_PASSWD	Définit le mot de passe pour télécharger les pack
--log-to-windows-events	Consigner les étapes dans le journal des événem
--use-gui	Force l'utilisation de GUI Helper même si elle n
--no-ide	Ne lancez pas d'idéal lors de l'édition d'un pack

CHAPITRE 36

Configuration avancée de l'agent

Le fichier de configuration `wapt-get.ini` définit le comportement de l'agent WAPT.

Tableau 1 – Emplacement du `wapt-get.ini` par le système

Système	Localisation
Windows	`C:\Program Files(x86)\wapt\wapt-get.ini`
Linux	`/opt/wapt/`
Mac OS	`/opt/wapt/`

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

Tableau 2 – Description des sections disponibles pour l'agent WAPT

Section	Description
[global]	Options de l'agent global.
[wapt]	Options du dépôt principal.
[wapt-template]	Options du dépôt externe à distance.
[wapt-host]	Dépôt pour les paquets de l'hôte options.
[waptwua]	Options de l'agent WUA.
[repo-sync]	Options de dépôt multiple.

Toutes les sections sont détaillées ci-dessous.

36.2 Description des options disponibles par section

36.2.1 [global]

Paramètres généraux

Tableau 3 – Description des options disponibles pour l'agent WAPT sur la section globale

Options / Valeur par défaut	Description	Exemple
allow_remote_reboot = False	Permet de redémarrer le ou les hôtes sélectionnés à distance à partir de la console WAPT.	allow_remote_reboot = True
allow_remote_shutdown = False	Permet d'arrêter le ou les hôtes sélectionnés à distance à partir de la console WAPT.	allow_remote_reboot = True
check_certificates_validity = False	Force la vérification de la date et de la CRL du certificat du paquet.	check_certificates_valid = True
dbpath = 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_wapt = True	Définit si le téléchargement des paquets en attente doit être lancé après une mise à jour avec waptupdate_task_period.	download_after_update_wi = False
host_organizational_unit_dn = 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 = non défini	Permet de définir une liste de paquets WAPT que l'agent WAPT doit installer.	host_profiles = tis-firefox,tis-java
language = langue par défaut sur le client	Force la langue par défaut pour l'interface graphique (pas pour le filtrage des paquets)	language = fr
locales = locale par défaut sur le client	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 = False	Envoie les journaux WAPT dans le journal des événements de Windows.	log_to_windows_events = True
loglevel = warning	Niveau de journalisation de l'agent WAPT. Les valeurs possibles sont : debug, info, warning, critical.	loglevel = critical
maturités = (par défaut 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.	échéances = PROD, PREPROD
repo_url = l'adresse de votre dépôt WAPT	Adresse du dépôt principal de WAPT.	repo_url = https://srvwapt.mydomain.lan/wapt
repositories (par 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 = 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 = 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 = True	S'il est possible ou non pour l'utilisateur de désinstaller des applications via le self-service.	uninstall_allowed = False
use_ad_groups = False	Permet d'utiliser les *groupes* (par défaut False).	use_ad_groups = True
use_fqdn_as_uuid = 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 = False	Utilisez *paquets host* (par défaut False).	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.

Tableau 4 – 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` = 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` (par défaut `False`)	Voir la documentation sur l'activation de la *vérification des certificats HTTPS*.	`verify_cert` = True
`wapt_server` (par défaut `None`)	URL du serveur WAPT. Si cet attribut n'est pas présent, aucun serveur WAPT ne sera contacté.	`wapt_server` = https://srvwapt.mydomain.lan
`wapt_server_timeout` = 30	Délai de connexion HTTPS du serveur WAPT en secondes.	`wapt_server_timeout` = 10

Paramètres pour le wapttray

Tableau 5 – Description des options disponibles pour l'agent WAPT sur la section globale pour le wapttray

Options / Valeur par défaut	Description	Exemple
""allow_cancel_upgrade"" = 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` = None	Désactive Hiberboot sur Windows 10 pour **`waptexit`**.	`hiberboot_enabled` = True
`max_gpo_script_wait` = None	Délai d'exécution des GPO à l'arrêt de l'ordinateur.	`max_gpo_script_wait` = 180
`pre_shutdown_timeout` = None	Délai d'attente pour les scripts à l'arrêt de l'ordinateur.	`pre_shutdown_timeout` = 180
""upgrade_only_if_not_process_running"" = False	Empêche la mise à niveau du logiciel si celui-ci est en cours d'exécution sur l'hôte (attribut *"process"* du packaging).	""upgrade_only_if_not_process_running"" = True
`upgrade_priorities` (par défaut `None`)	Ne mettre à niveau que les paquets ayant une priorité spécifique.	`upgrade_priorities` = high
""waptexit_countdown"" = 1	Temps avant le démarrage automatique des installations.	""waptexit_countdown"" = 25

Paramètres pour l'authentification WAPT Self-Service et Waptservice

Tableau 6 – 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` = 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` = False	Utile avec *waptagent-ldap*, définit si la requête LDAP doit être chiffré.	`ldap_auth_ssl_enabled` = True
`ldap_auth_server` = None	Utile avec *waptagent-ldap*, définit le serveur LDAP à contacter.	`ldap_auth_server` = srvads.mydomain.lan
`service_auth_type` = 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` = False	Utile avec *waptagent-ldap*, définit si le certificat doit être vérifié.	`verify_cert_ldap` = True
`waptservice_admin_filter` = False	Appliquer un filtrage d'affichage pour les *paquets self-service* pour les administrateurs locaux.	`waptservice_admin_filter` = True
`waptservice_password` = 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` = None	Force un utilisateur à s'authentifier sur le service WAPT.	`waptservice_user` = admin

Paramètres pour le wapttray

Tableau 7 – Description des options disponibles pour l'agent WAPT sur la section globale pour le wapttray

Options / Valeur par défaut	Description	Exemple
`notify_user` (par défaut `False`)	Empêche `wapttray` d'envoyer des notifications (popup).	`notify_user` = True

Paramètres du Proxy

Tableau 8 – Description des options disponibles pour l'agent WAPT sur la section globale pour le proxy

Options / Valeur par défaut	Description	Exemple
`http_proxy` (par défaut `None`)	Définit l'adresse du proxy HTTP.	`http_proxy` = http://user:pwd@host_fqdn:port
`use_http_proxy_for_repo` = False	Utilisez un proxy pour accéder aux dépôts.	`use_http_proxy_for_repo` = True
`use_http_proxy_for_server` = False	Utilisez un proxy pour accéder au serveur WAPT.	`use_http_proxy_for_server` = True

Paramètres pour la création de paquets WAPT

Tableau 9 – 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` = tis	Définit le préfixe par défaut pour les nouveaux paquets ou ceux importés.	`default_package_prefix` = doc
`default_sources_root` = `C:\waptdev` (Windows) ou `~/waptdev` (Linux)	Définit le répertoire de stockage des paquets en cours de développement.	`default_sources_root` = `C:\waptdev`
`default_sources_suffix` = wapt	Définit le suffixe par défaut pour les paquets nouveaux ou importés.	`default_sources_suffix` = doc
`personal_certificate_path` = None	Définit le chemin d'accès à la clé privée de l'administrateur.	`personal_certificate_path` = 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*.

[sections]

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.

[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*.

CHAPITRE 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

TABLEAU 1 – Description des sections disponibles pour l'agent WAPT

Section	Description
`[global]`	options globales de la console
`[sections]`	options du dépôt externe. `[wapt-template]` est le *dépôt Tranquil IT* par défaut
`[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`.

Tableau 2: Description des options disponibles sur `AppData\Local`

Options / Valeur par défaut	Description	Exemple
`advanced_mode` = False	Lance la console en mode débogage.	
`allow_remote_reboot` = False	Permet de redémarrer le ou les hôtes sélectionnés à distance	
`allow_remote_shutdown` = False	Permet d'arrêter le ou les hôtes sélectionnés à distance à part	
`client_certificate` = None	Définit si le dépôt distant utilise l'authentification SSL côté c	
`client_private_key` = None	Définit si le dépôt distant utilise l'authentification SSL côté c	
`check_certificates_validity` = False	Force la vérification de la date et de la CRL du certificat du p	
`default_maturity` (par défaut `None`)	Définit la maturité de téléchargement par défaut pour les paq	
`default_package_prefix` = tis	Définit le préfixe par défaut pour les nouveaux paquets ou ce	
`default_sources_root` = `C:\waptdev` (Windows) ou `~/waptdev` (Linux)	Définit le répertoire de stockage des paquets en cours de dév	
`grid_hosts_plugins` = W10=	Liste *Les plugins externes* pour la console WAPT. La valeur	
`host_profiles` (par défaut `None`)	Définit une liste de paquets WAPT que l'agent WAPT doit in	
`hiberboot_enabled` = False	Désactive Hiberboot sur Windows 10 pour **`waptexit`**	
`http_proxy` (par défaut `None`)	Définit l'adresse du serveur proxy dans la console WAPT.	
`last_usage_report` (par défaut `None`)	Indique la date à laquelle la console WAPT a été utilisée pou	
`lastwaptserveruser` (par défaut `None`)	Fournit le dernier utilisateur connecté sur cette console WAP	
`max_gpo_script_wait` = 180	Définit le délai d'exécution des GPO à l'arrêt de l'ordinateur.	
`personal_certificate_path` = None	Définit le chemin d'accès au certificat associé à la clé privée	
`pre_shutdown_timeout` = 180	Définit le délai d'attente pour les scripts à l'arrêt de l'ordinat	
`repo_url` = l'adresse de votre dépôt WAPT	Définit l'adresse du dépôt principal de WAPT.	
`send_usage_report` = True	Permet à la console WAPT d'envoyer des statistiques anonyn	
`sign_digests` = sha256	Liste des algorithmes de signature autorisés pour les paquets	
`use_ad_groups` = False	Permet l'utilisation des *paquets unit*.	
`use_fqdn_as_uuid` = False	Permet d'utiliser le FQDN plutôt que l'UUID du BIOS comm	
`use_kerberos` (par défaut `False`)	Permet d'utiliser l'authentification kerberos pour l'enregistre	
`use_hostpackages` = False	Permet d'utiliser *les paquets hôtes*.	
`use_http_proxy_for_repo` = False	Permet d'utiliser un proxy pour se connecter au dépôt princi	
`use_http_proxy_for_server` = False	Permet d'utiliser un proxy pour se connecter au serveur WAP	
`use_repo_rules` = False	Permet d'utiliser *replication pour les dépôts*.	
`verify_cert` (par défaut `False`)	Pour la *vérification des certificats SSL / TLS*.	
`wapt_server` (par défaut `None`)	Définit le port du serveur PostgreSQL.	

Tableau 3 – Description des options disponibles sur AppData\Roaming

Options / Valeur par défaut	Description	Exemple
advanced_mode = False	Lance la console en mode débogage.	advanced_mode = True
enable_external_tools = False	Affiche les actions qui appellent des applications externes (RDP, outils Windows etc…).	enable_external_tools = True
enable_management_features = 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 = False	Masque les actions qui ne sont pas disponibles pour l'agent WAPT	hide_unavailable_actions = True
HostsLimit (par défaut 2000)	Limite des hôtes affichés dans la console WAPT.	HostsLimit = 300
language = langue par défaut sur le client	Force la langue par défaut pour l'interface graphique (pas pour le filtrage des paquets)	language = fr
lastappinifilename (par défaut None)	Définit le fichier *.ini* utilisé pour stocker la configuration de la console WAPT.	lastappinifilename = C:\Users\%username%\AppData\Roaming\waptconsole\waptconsole.ini
show_host_audit_data_tab = False	Affiche l'onglet *Données d'audit* sur l'inventaire des machines.	show_host_audit_data_tab = True
use_ad_groups = False	Permet l'utilisation des *paquets unit*.	use_ad_groups = True
use_fqdn_as_uuid = 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 (par défaut None)	Version de la console	waptconsole.version = 2.0.0.9424
waptwua_enabled = False	Pour afficher l'onglet Mises à jour Windows sur la console	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*.

CHAPITRE 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]
```

Tableau 1: Paramètres disponibles pour la section [option] de waptserver.ini

Options / Valeur par défaut	Description	Exemple
`allow_unauthenticated_connect` = None	Définit si les connex	
`allow_unauthenticated_registration` = False	Permet l'enregistrer	
`allow_unsigned_status_data` = False	Débogage uniquem	
`application_root` (par défaut `None`)	Définit un chemin r	
`client_certificate_lifetime` = 3650	Définit la durée de	
`cleanup_kbs` (par défaut `True`)	Définit si les KB W	
`clients_read_timeout` = 5	Définit le délai d'at	
`clients_signing_certificate` = None	Définit le chemin d	
`clients_signing_crl_days` = 30	Définit la durée de	

suite sur la page suivante

Tableau 1 – suite de la page précédente

Options / Valeur par défaut	Description	Exemple
clients_signing_crl_url = None	Définit le chemin d
clients_signing_crl_url = None	Définit l'URL de la
clients_signing_key = None	Définit le chemin d
client_tasks_timeout = 5	Définit le délai max
db_connect_timeout = 3	Définit le délai max
db_host (par défaut None)	Définit l'url du serv
db_max_connections = 90	Définit l'url du serv
nom_bd (par défaut wapt)	Définit la base de d
db_password (par défaut None)	Définit le mot de pa
db_port (par défaut 5432)	Définit le port du se
db_stale_timeout = 300	Définit le délai du t
db_user (par défaut wapt)	Définit l'utilisateur
enable_store (par défaut False)	Active le Webui du
encrypt_host_packages = False	Chiffre le paquet ma
htpasswd_path (par défaut None)	Ajoute l'authentific
http_proxy (par défaut None)	Définit le serveur pr
known_certificates_folder = dossier par défaut de WAPT /ssl/ folder	Ajoute une CA supp
ldap_auth_base_dn = None	Définit le DN de ba
ldap_auth_server = None	Définit le serveur d'
ldap_auth_ssl_enabled = True	Définit l'authentific
loglevel (par défaut warning)	Définit le niveau du
max_clients = 4096	Définit la connexion
min_password_length = 10	Définit la longueur
nginx_http (par défaut 80)	Définit le port **HTT**
nginx_https (par défaut 443)	Définit le port **HTT**
remote_repo_support = False	Active la fonctionna
remote_repo_websockets = True	Permet la communi
secret_key (par défaut None)	Définit la chaîne alé
server_uuid (par défaut None)	Définit le serveur W
signature_clockskew = 300	Définit la différence
token_lifetime = 43200	Définit la durée de v
trusted_signers_certificates_folder = None	Définit le chemin d'
trusted_users_certificates_folder = None	Définit le chemin d'
use_kerberos (par défaut False)	Permet à un agent W
use_ssl_client_auth = False	Active *l'authentifica*
wapt_admin_group_dn (par défaut None)	DN LDAP du group
wapt_folder = /var/www/wapt ou /var/www/html/wapt ou WAPT root_dir/waptserver/repository/wapt	Définit le chemin du
wapt_huey_db (par défaut None)	Définit le chemin d'
wapt_password (par défaut None)	Définit le mot de pa
waptserver_port = 8080	Définit le port du se
wapt_user (par défaut admin)	Définit le nom d'uti
waptwua_folder = dossier_wapt + "wua"	Définit l'emplacem
wol_port (par défaut 9)	Définit la liste des p
wapt_bind_interface = 127.0.0.1	Définit comment éc
ipxe_script_jinja_path (par défaut ``/opt/wapt/waptserver/templates/ipxe-default.j2`)	Définit l'emplacem

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;
    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;
    }
```

(suite sur la page suivante)

(suite de la page précédente)

```
  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

Tableau 2 – `nginx.conf` emplacement du fichier de configuration

Type d'OS	Localisation
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

Tableau 3 – `postgresql.conf` emplacement du fichier de configuration

Type d'OS	Localisation
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
```

CHAPITRE 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.

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.

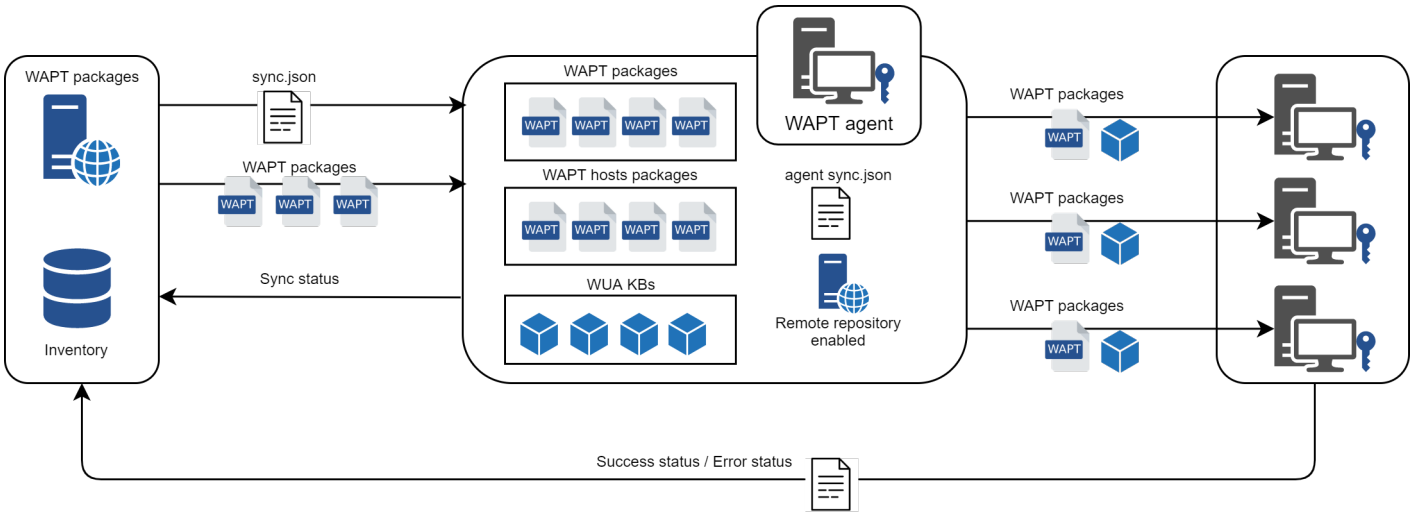


FIG. 1 – Diagramme de flux du comportement de réplication de l'agent WAPT

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`.

Tableau 1 – Configuration de la réplication de l'agent WAPT

Options / Valeur par défaut	Définition	Exemple
`enable_remote_repo` = False	Permet au dépôt secondaire de se synchroniser avec le dépôt principal.	`enable_remote_repo` = True
`local_repo_path` = Dossier de WAPT /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` = 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` = 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` = None	Définit la périodicité de la synchronisation (minutes)	`local_repo_sync_task_period` = 25
`local_repo_limit_bandwidth` = None	Définit la largeur de bande autorisée pour la synchronisation (Mbits/s)	`local_repo_limit_bandwidth` = 2
`remote_repo_dirs` = wapt,waptwua	Définit les dossiers à synchroniser	`remote_repo_dirs` = wapt,waptwua,wapt-host
`use_repo_rules` = False	Activer pour l'utilisation des *règles du dépôt*	`use_repo_rules` = True

Avertissement : Si vous modifiez manuellement le fichier `wapt-get.ini` sur le dépôt secondaire, vous devez redémarrer **wapt-service**.

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
```

(suite sur la page suivante)

(suite de la page précédente)

```
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

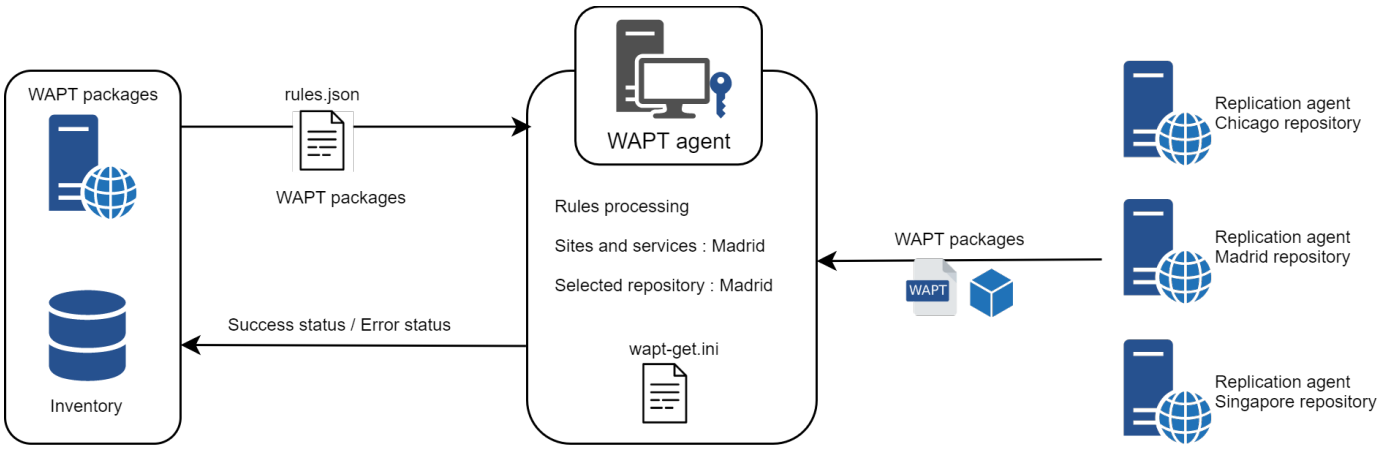
Par défaut, le serveur sait quels agents WAPT sont configurés en tant que dépôt secondaire et il les répertorie dans la console WAPT.

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` = 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 :

Tableau 2 – Paramètres disponibles pour les règles du dépôt

Options	Exemple de valeur	Description
Name	`192.168.85.0/24`	Règle basée sur le sous-réseau IP de l'agent.
Condition	`ad.mydomain.lan`	Règle basée sur le nom de domaine Active Directory.
Name	`desktop-04feb1`	Règle basée sur le nom d'hôte de l'agent WAPT.
Valeur	`256.89.299.22/32`	Règle basée sur l'adresse IP publique (hôtes NATés).
Autre	`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 *Repositories* de la console WAPT et cliquez sur le bouton *Add rule*.

Tableau 3 – Création d'une nouvelle règle de dépôt

Options	Exemple de valeur	Description
Name	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.

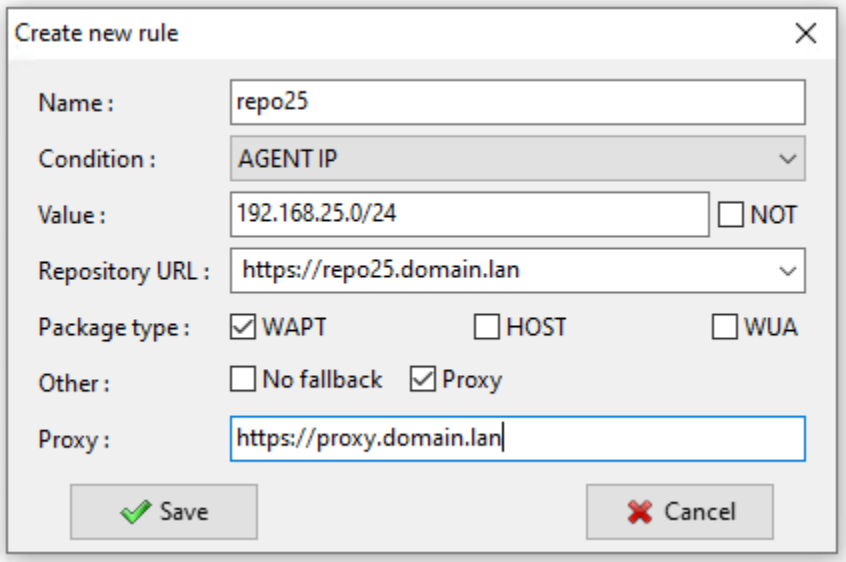


Fig. 2 – Fenêtre pour définir les règles du référentiel dans la console WAPT

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

Tableau 4 – Description des options disponibles pour l'utilisation de dépôts multiples

Options	Exemple de valeur	Description
[global]	dépôts = 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 = 1 `repo_url = https://srvwapt.mydomain.lan/wapt`	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.

Tableau 5 – Options pour les propriétés du référentiel

Options / Valeur par défaut	Description	Exemple
use_repo_rules = False	Définit l'adresse du proxy HTTP.	http_proxy = http://user:pwd@host_fqdn:port
use_repo_rules = False	Définit l'adresse du dépôt WAPT principal.	repo_url = https://srvwapt.mydomain.lan/wapt
local_repo_time_for_sync = None	Définit le délai d'attente lors de la connexion à des dépôts distants. Valeur en millisecondes.	timeout = 5000
use_repo_rules = False	Définit si un proxy doit être défini pour accéder aux dépôts.	use_http_proxy_for_repo = 1
use_repo_rules = False	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 = None

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
```

Tableau 6 – Options pour les référentiels externes dans la Console WAPT

Options / Valeur par défaut	Description	Exemple
client_certificate = 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 (sur 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 (sur Windows)
use_repo_rules = False	Définit l'adresse du proxy HTTP.	http_proxy = http://proxy.mydomain.lan:8080
public_certs_dir =	Définit le dossier qui contient les certificats utilisés pour authentifier les paquets externes téléchargés.	public_certs_dir = C:\private
use_repo_rules = False	Définit l'adresse du dépôt WAPT principal.	repo_url = https://srvwapt.mydomain.lan/wapt
local_repo_time_for_sync_start = None	Définit le délai d'attente lors de la connexion à des dépôts distants. Valeur en millisecondes.	timeout = 5000
use_repo_rules = False	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 = None

CHAPITRE 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 : **srv-ads.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`.

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
```

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`.

Debian / Ubuntu

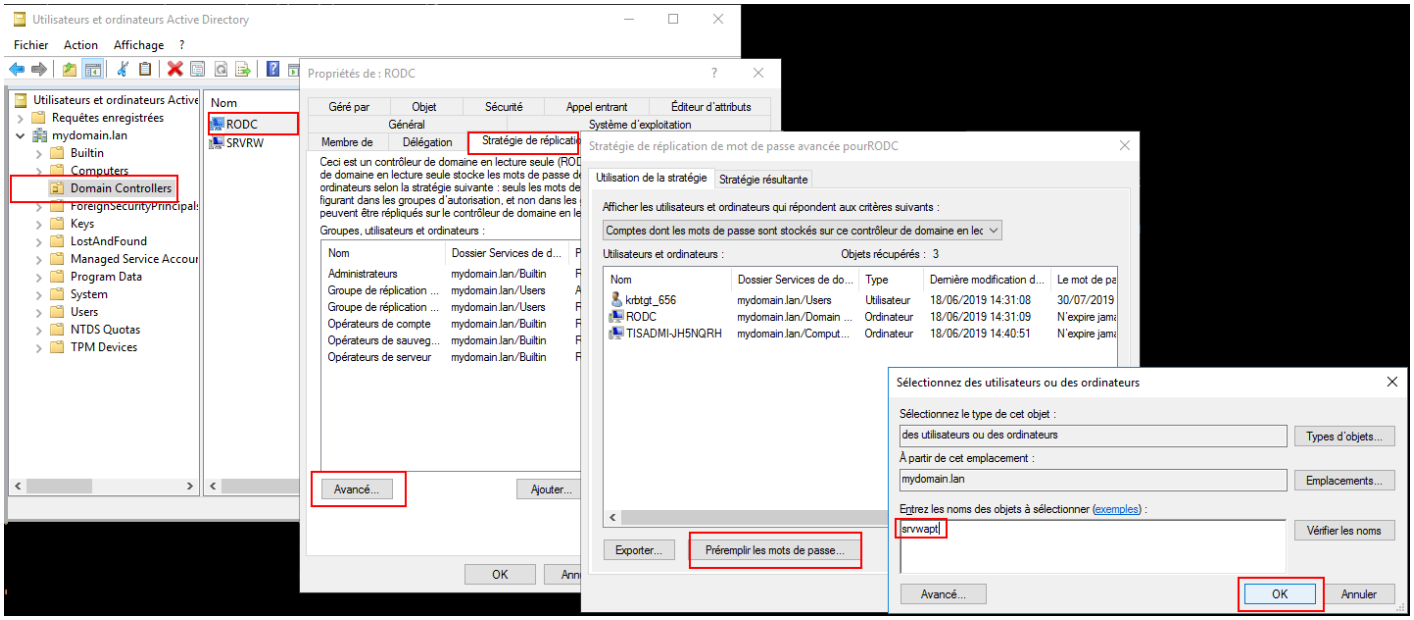
```
sudo chmod 640 /etc/nginx/http-krb5.keytab
sudo chown root:www-data /etc/nginx/http-krb5.keytab
```

CentOS / RedHat

```
sudo chown root:nginx /etc/nginx/http-krb5.keytab
sudo chmod 640 /etc/nginx/http-krb5.keytab
```

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=1
```

- 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=1
```

- 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;
```

(suite sur la page suivante)

(suite de la page précédente)

```
      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,
```

(suite sur la page suivante)

(suite de la page précédente)

```
↪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
*  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
```

(suite sur la page suivante)

(suite de la page précédente)

```
* 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
> 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
  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`.

Tableau 1 – 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
net stop waptservice
net start 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 :

Debian / Ubuntu

```
chown root:www-data /opt/wapt/waptserver/ssl/*.pem
```

CentOS / RedHat

```
chown root:nginx /opt/wapt/waptserver/ssl/*.pem
```

— 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_dn=CN=waptadmins,OU=groupes,OU=tranquilit,DC=mydomain,DC=lan
ldap_auth_server=srvads.mydomain.lan
ldap_auth_base_dn=DC=mydomain,DC=lan
ldap_auth_ssl_enabled=False
```

Tableau 2 – Options d'authentification disponibles

Options / Valeur par défaut	Description	Exemple
`wapt_admin_group_dn` = []	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,DC=mydomain,DC=lan
`ldap_auth_server` = None	Définit le serveur d'authentification LDAP	`ldap_auth_server` = srvads.mydomain.lan
`ldap_auth_base_dn` = None	Définit le DN de base de l'authentification LDAP	`ldap_auth_base_dn` = dc=domain,dc=lan
`ldap_auth_ssl_enabled` = True	Définit l'authentification SSL sur les connexions LDAP	`ldap_auth_ssl_enabled` = 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.

Vous pouvez utiliser Kerberos pour vous authentifier sur la **console WAPT** et le **selfservice**. De cette façon, les utilisateurs n'ont plus besoin de saisir leur mot de passe.

Indication : Il n'est pas nécessaire d'utiliser Kerberos pour l'enregistrement des agents afin d'utiliser le SSO.

Préparer le serveur pour Kerberos Single Sign On

Attention : Pour activer Kerberos sur le serveur WAPT avec l'option `use_kerberos = True`, veuillez vous référer à *la configuration de l'authentification Kerberos* au préalable.

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 .

Avec ces options, vous ne vérifiez pas la certification ldap et n'utilisez pas de port sécurisé pour requêter le serveur :

```
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 :

```
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.

Il s'agit d'activer l'authentification ssl mais sans vérification de la certification :

```
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
```

— Puis redémarrez les services sur le serveur :

```
systemctl restart waptserver wapttasks
```

Note : 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.

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.

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 2024-01-09, 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.

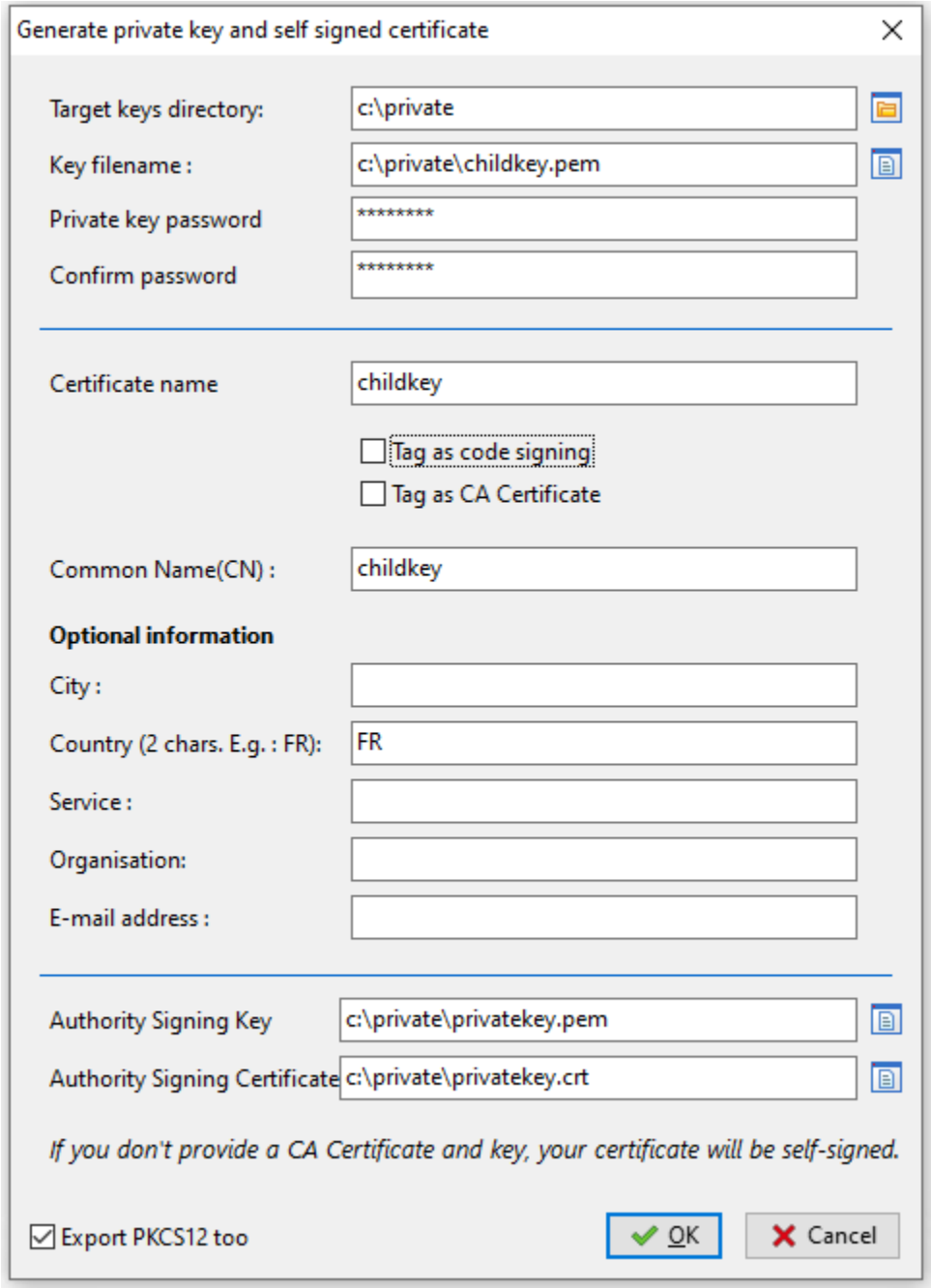


FIG. 1 – Génération d'un certificat sans l'attribut *Pour Signature de code*

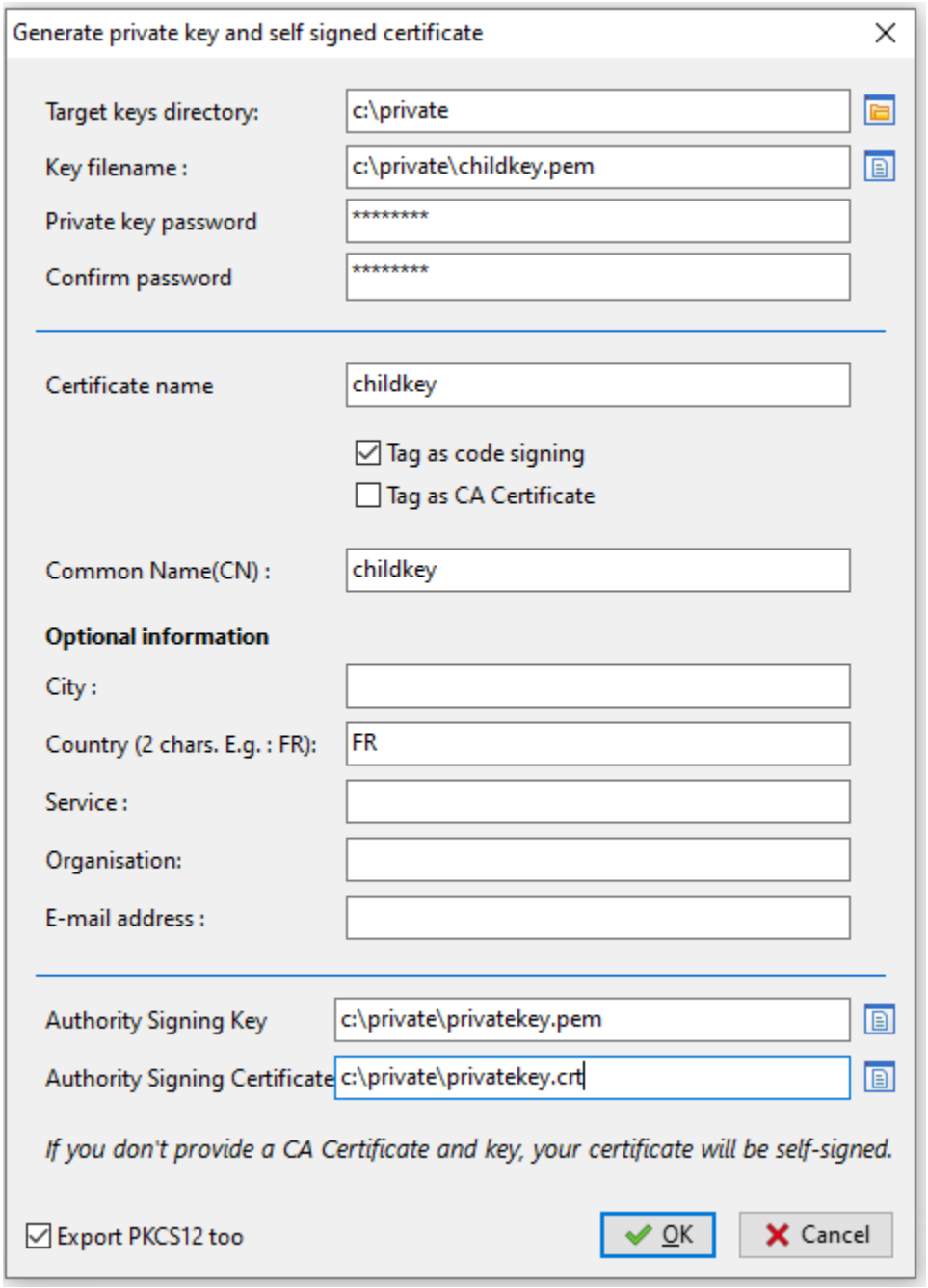


FIG. 2 – Génération d'un certificat avec l'attribut *Pour Signature de code*

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 : N'oubliez pas de redémarrer le service WAPT sur les clients pour qu'ils utilisent leur nouveau certificat. Ouvrez une ligne de commande **`cmd.exe`** puis :

```
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`.

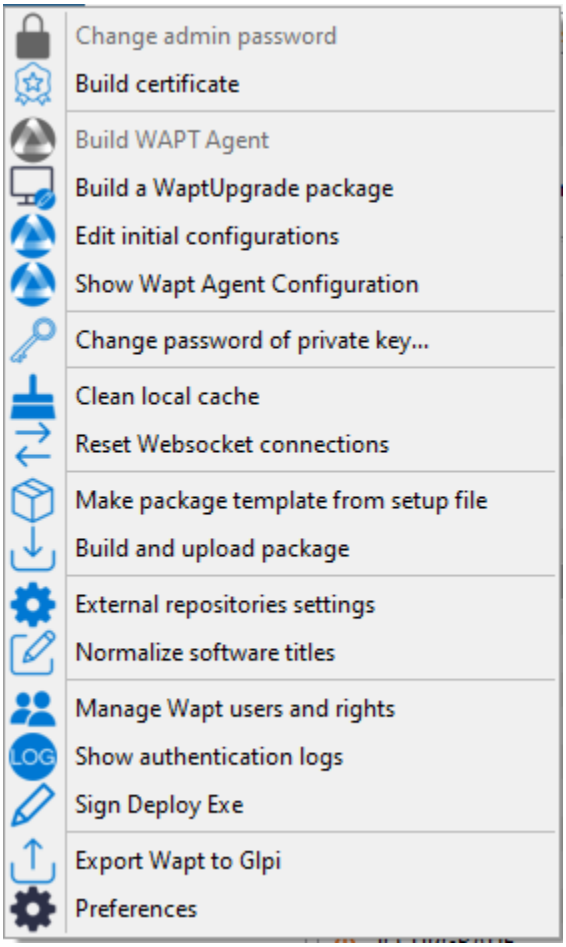
L'ACL par défaut pour un nouvel utilisateur est `vue`.

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
```

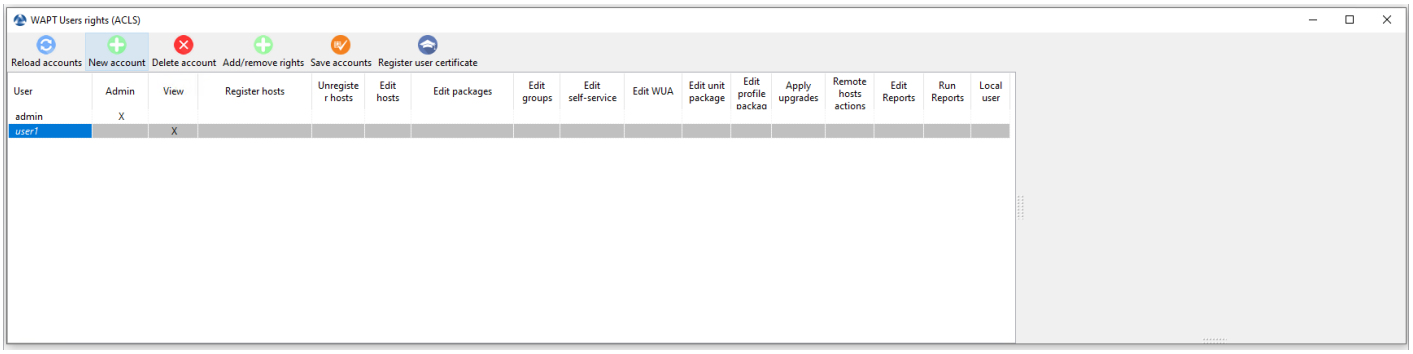
— Sur `waptserver.ini` ajoutez les paramètres `htpasswd_path`.

```
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*.

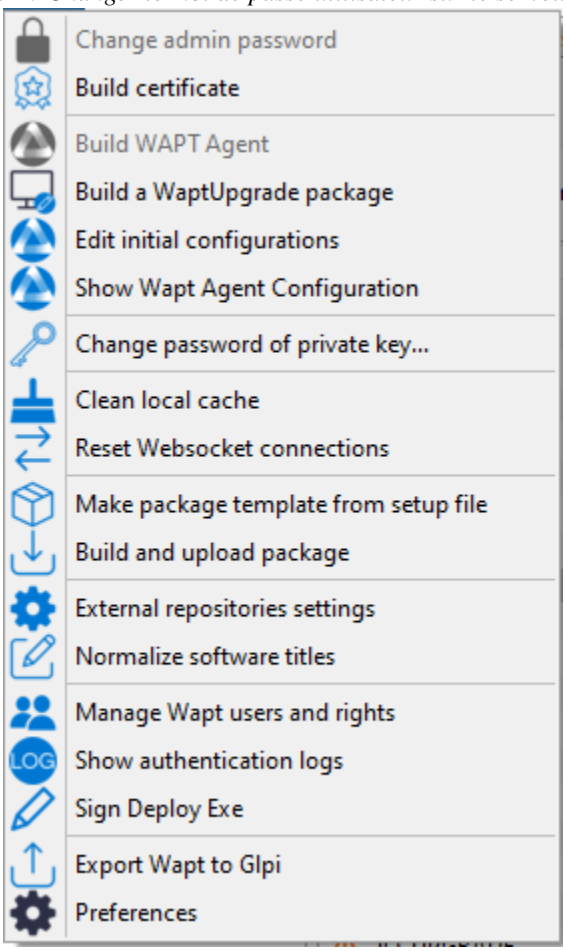
— Sauvegardez en cliquant sur *Enregistrer les comptes*.
— Pour définir un mot de passe, voir le point **Changez le mot de passe**.
— Pour définir les droits, consultez la section *gérer les droits ACL*.

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.

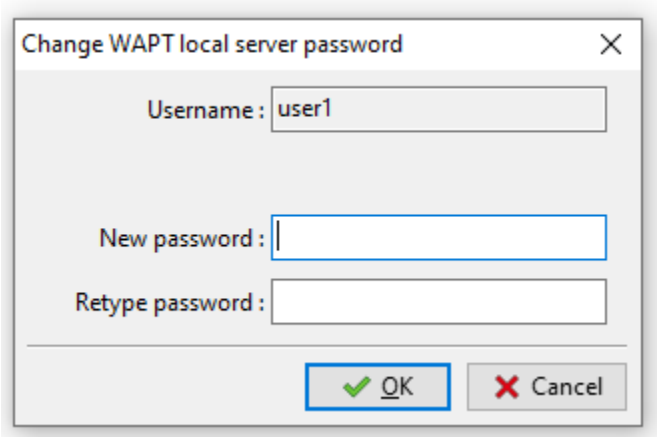


FIG. 3 – Boîte de dialogue permettant de modifier le mot de passe de l'utilisateur dans le fichier htaccess

L'utilisateur local apparaît en *gras* et la case *Mots de passe* est cochée.

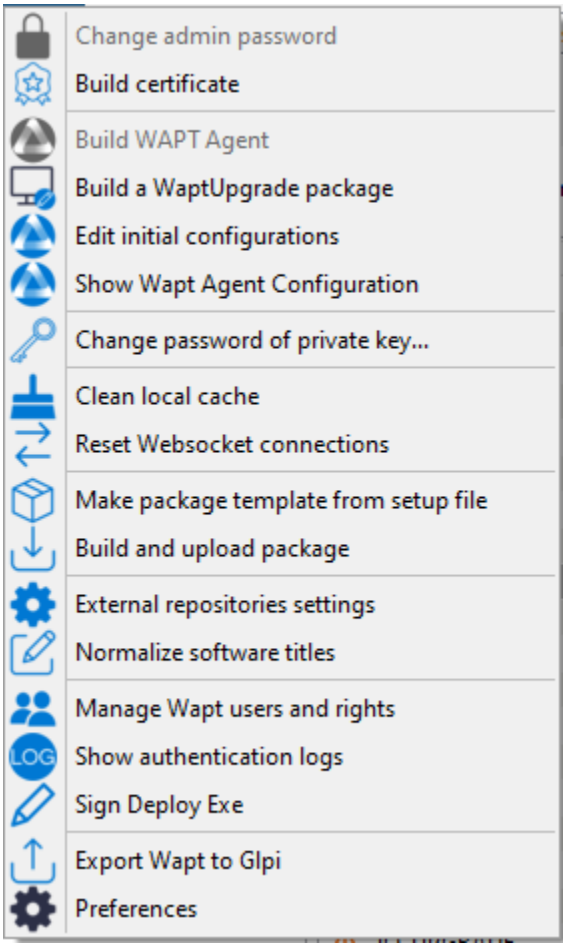
40.8.2 Utilisateurs WAPT définis comme utilisateurs Active Directory

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.

Tableau 3 – 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 Admin pour *enregistrer manuellement un hôte* 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.

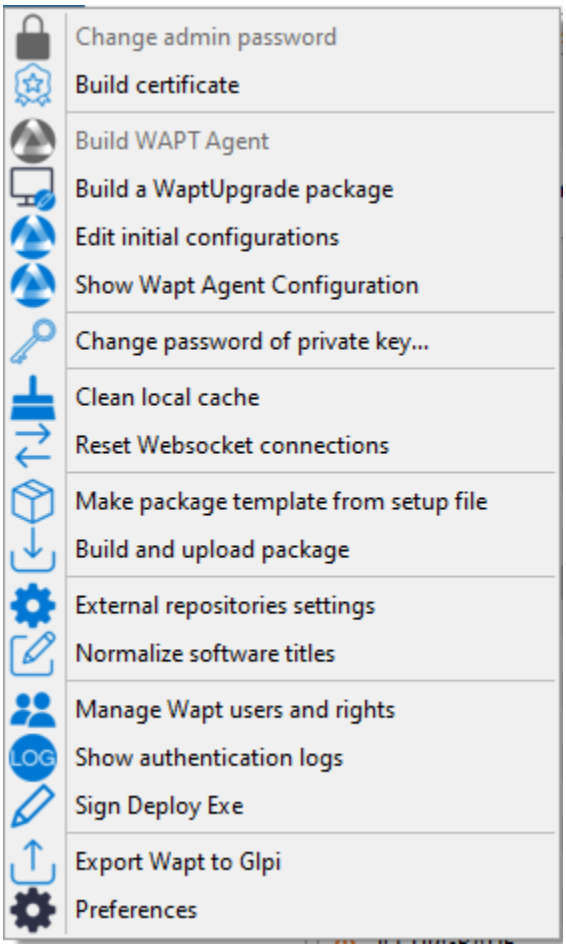
Attention : Si les certificats ne sont pas émis par l'autorité de certification :
- Les paquets mis à jour sont disponibles uniquement sur les ordinateurs où les certificats sont déployés.
- Les ACL sont valides uniquement sur le périmètre des hôtes où le certificat de l'administrateur est déployé.

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

Tableau 4 – 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.

Tableau 5 – 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.

CHAPITRE 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).

CHAPITRE 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.

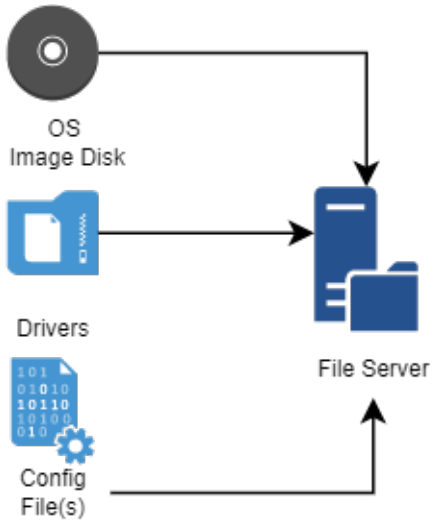


FIG. 1 – Diagramme de flux pour l'importation des fichiers requis pour le déploiement de WADS

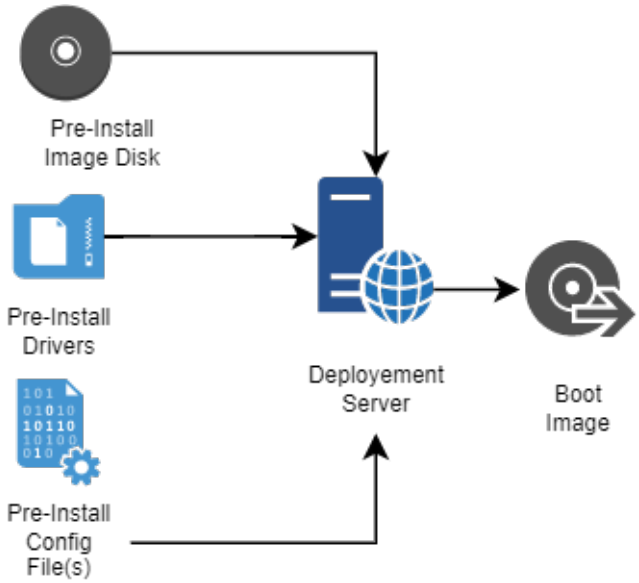


FIG. 2 – Diagramme de flux pour la création du support d'amorçage pour le déploiement WADS

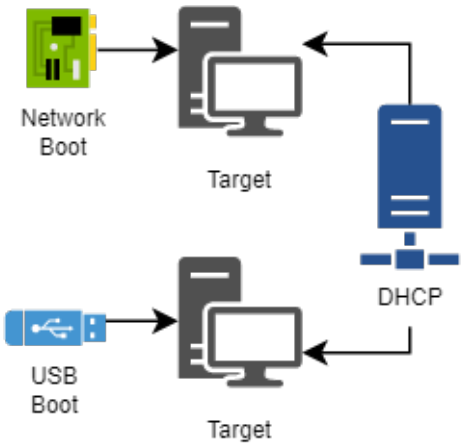


FIG. 3 – Diagramme de flux pour l'utilisation du support de démarrage dans le déploiement WADS

Différences entre les logiciels

TABLEAU 1 – 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

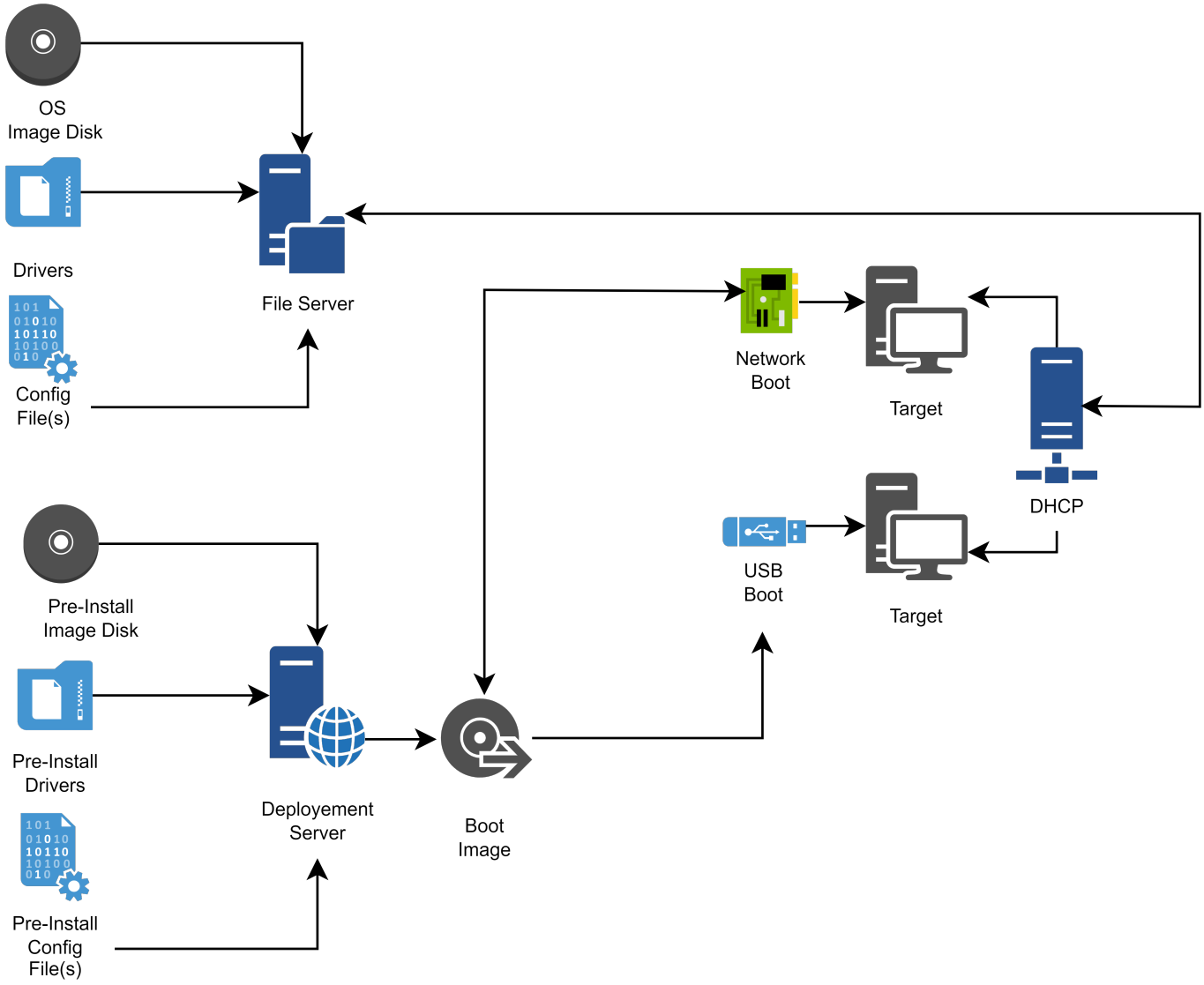


FIG. 4 – Diagramme de flux pour le déploiement d'un OS classique

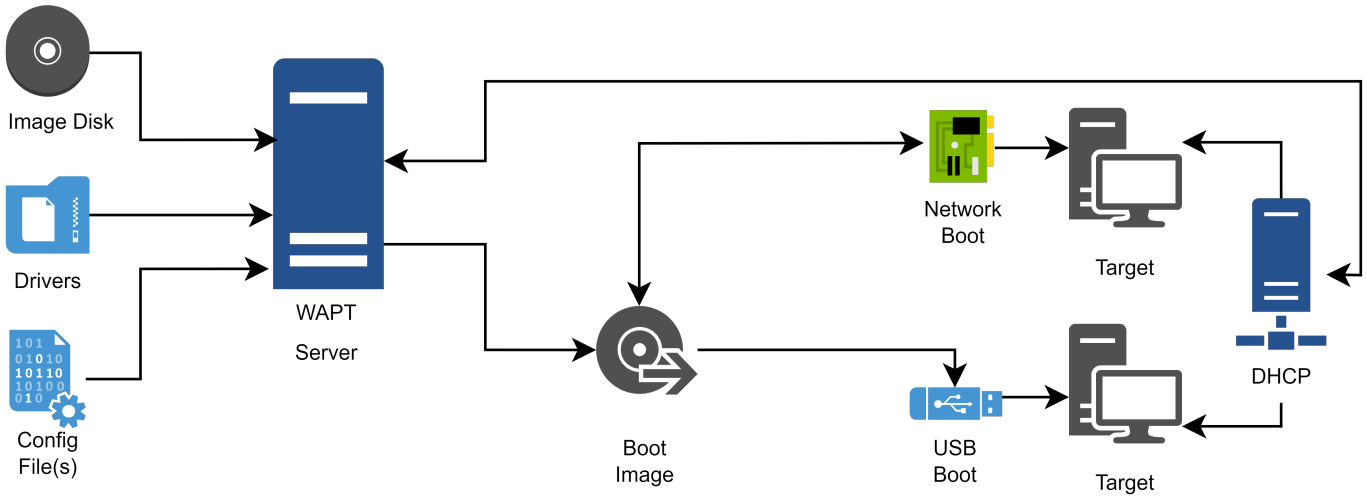


FIG. 5 – Diagramme de flux pour un déploiement WADS

CHAPITRE 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";
  }
  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";
  }
  else{
    filename "ipxe.efi";
  }
}
```

Pour plus d'informations, vous pouvez consulter le site https://ipxe.org/howto/dhcpd

Windows

Vous pouvez utiliser la ligne de commande PowerShell suivante pour configurer le démarrage iPXE sur votre réseau. Veuillez adapter les *$url_waptserver* et *$waptserver_ipaddress_tftp* en fonction de votre installation actuelle.

```
$waptserver_ipaddress_tftp = "192.168.154.13"
$url_waptserver = "http://srvwapt.mydomain.lan"
```

(suite sur la page suivante)

(suite de la page précédente)

```
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"

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"

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"
```

Pour plus d'informations, vous pouvez consulter le site https://ipxe.org/howto/msdhcp

CHAPITRE 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.

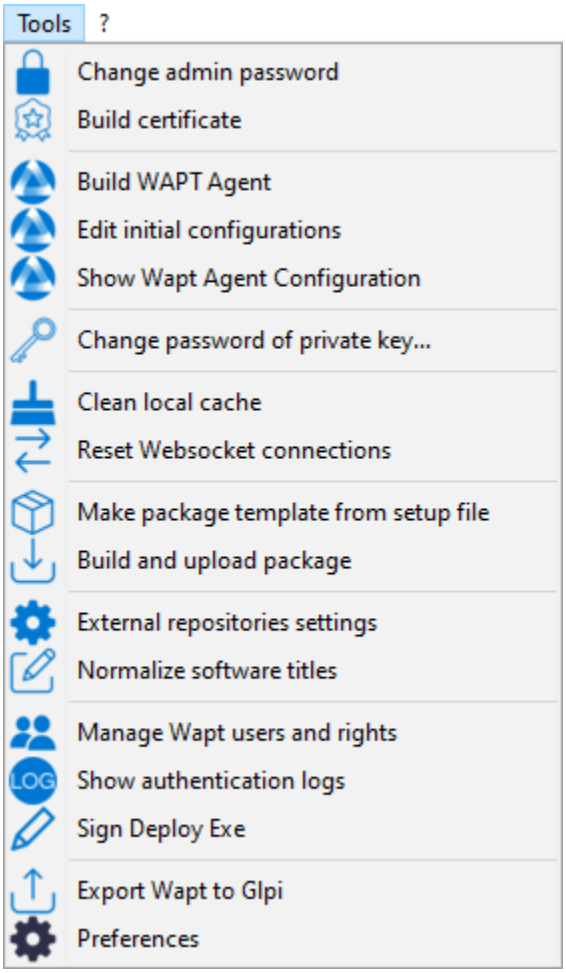
44.2 Exigences avant de commencer

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 2024-01-09, 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* :

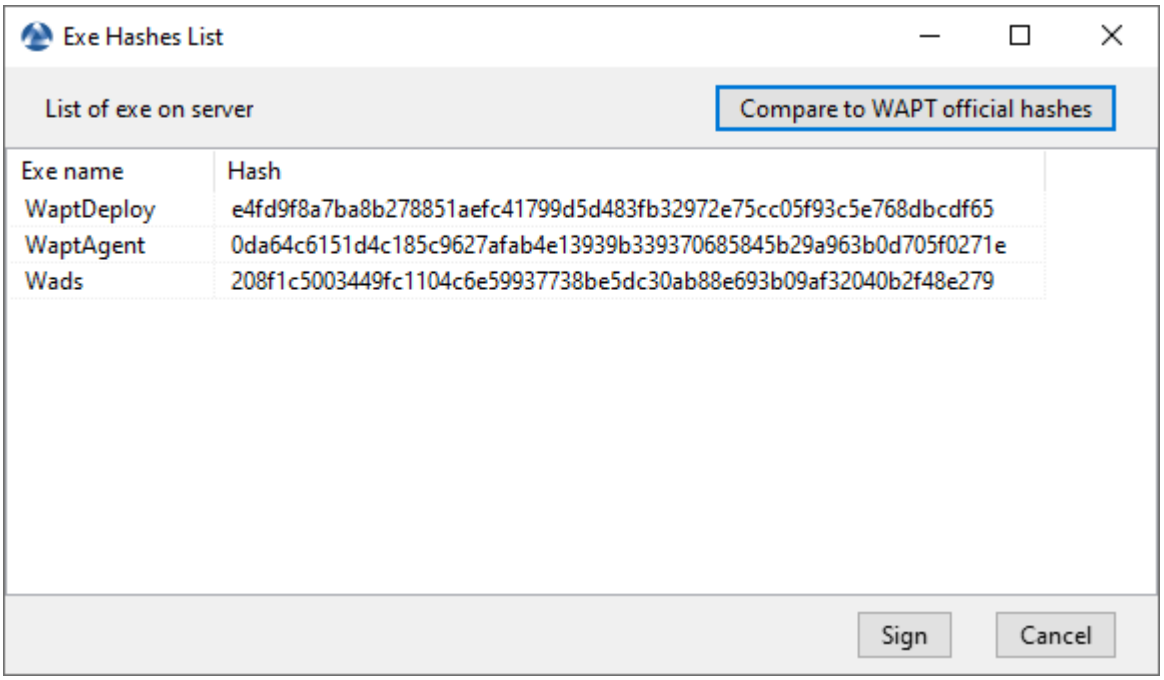


FIG. 1 – Fenêtre de signature des binaires dans la console WAPT

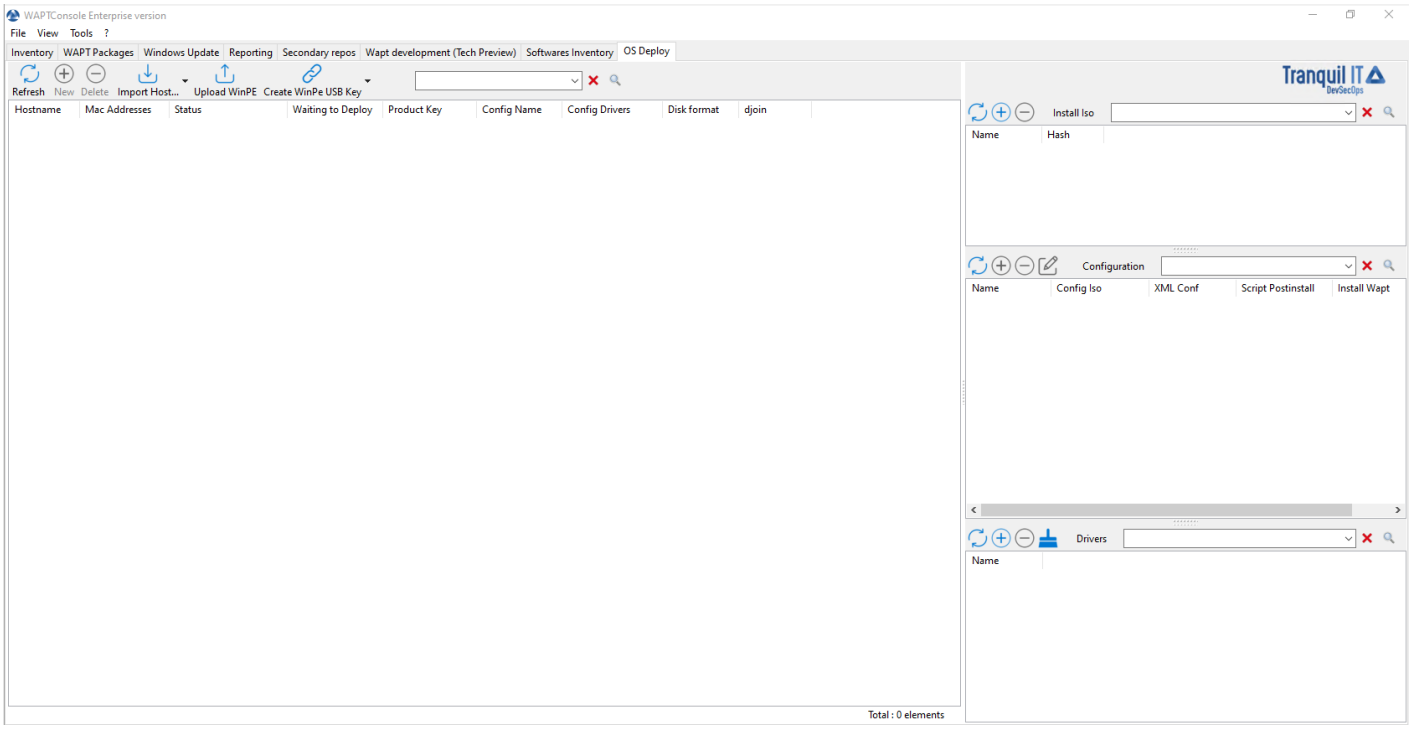


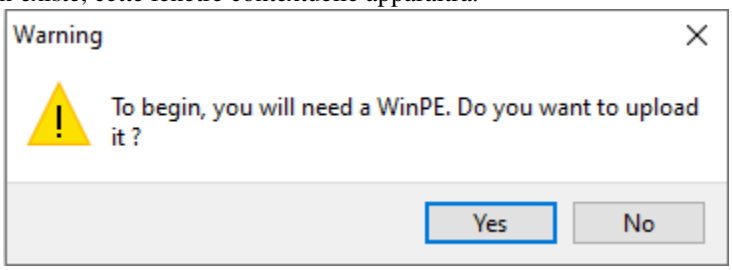
FIG. 2 – Fenêtre principale de la console WADS

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**.

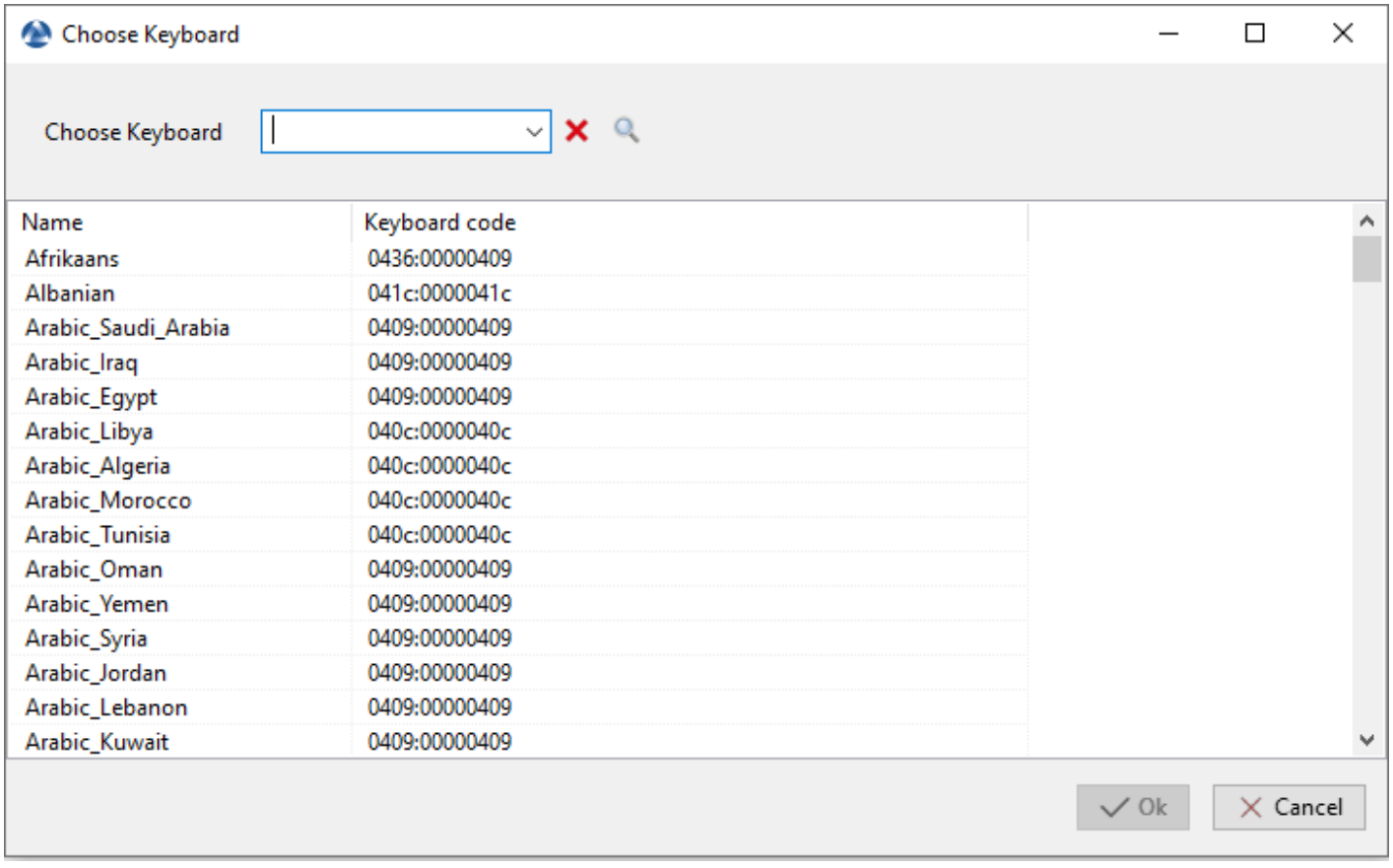


FIG. 3 – Boîte de dialogue pour sélectionner le clavier dans la console WADS

— Sélectionnez 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 :

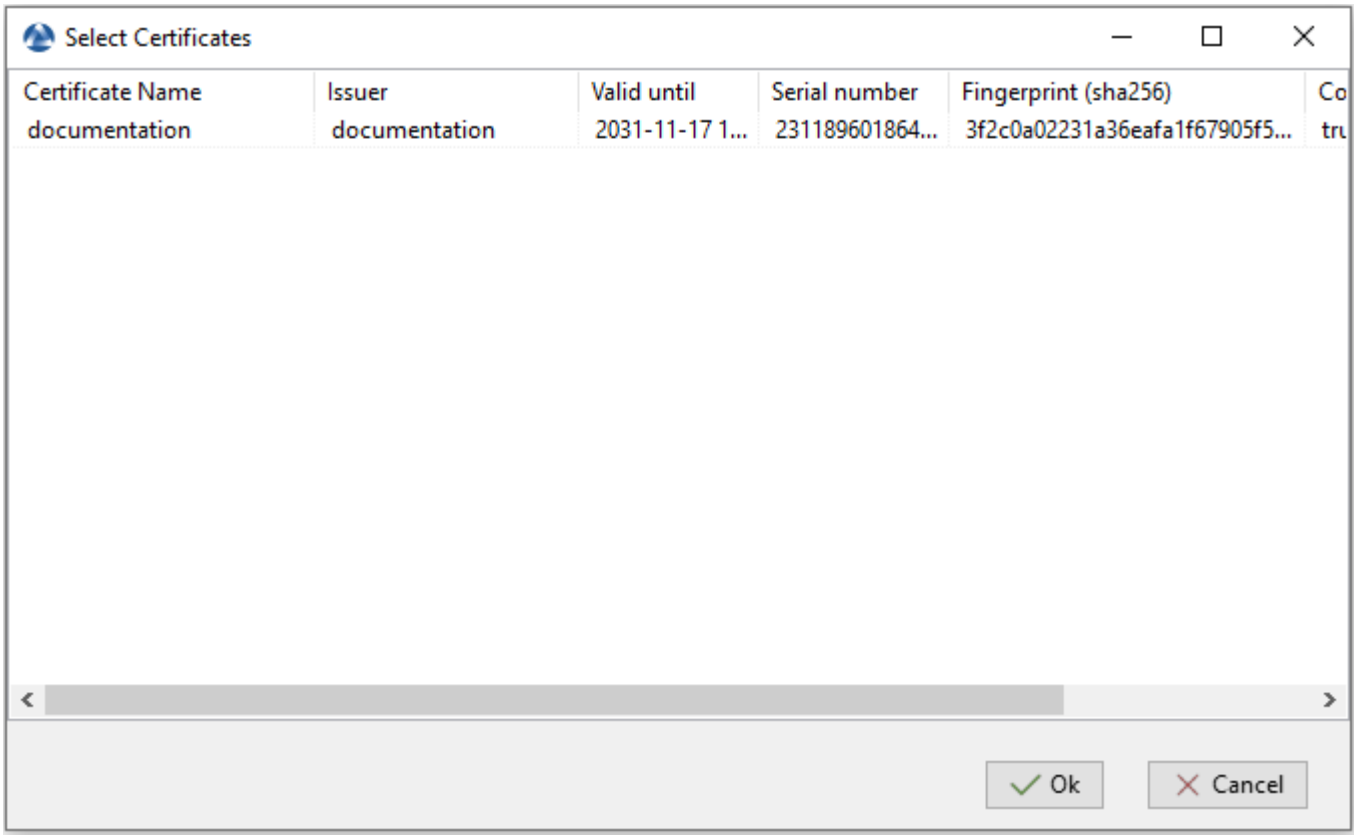
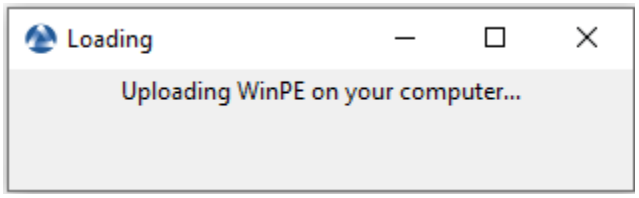


FIG. 4 – Boîte de dialogue pour la sélection du certificat dans la Console WADS



— Attendez que le fichier `WinPE` soit téléchargé sur le serveur WADS :

Note : 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*.

FIG. 5 – Section ISO de la console WADS

— 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.

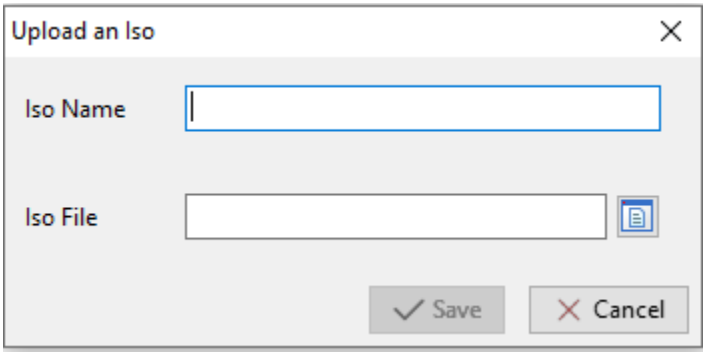


FIG. 6 – Boîte de dialogue permettant de sélectionner le fichier ISO à télécharger sur le serveur WADS

— 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 :

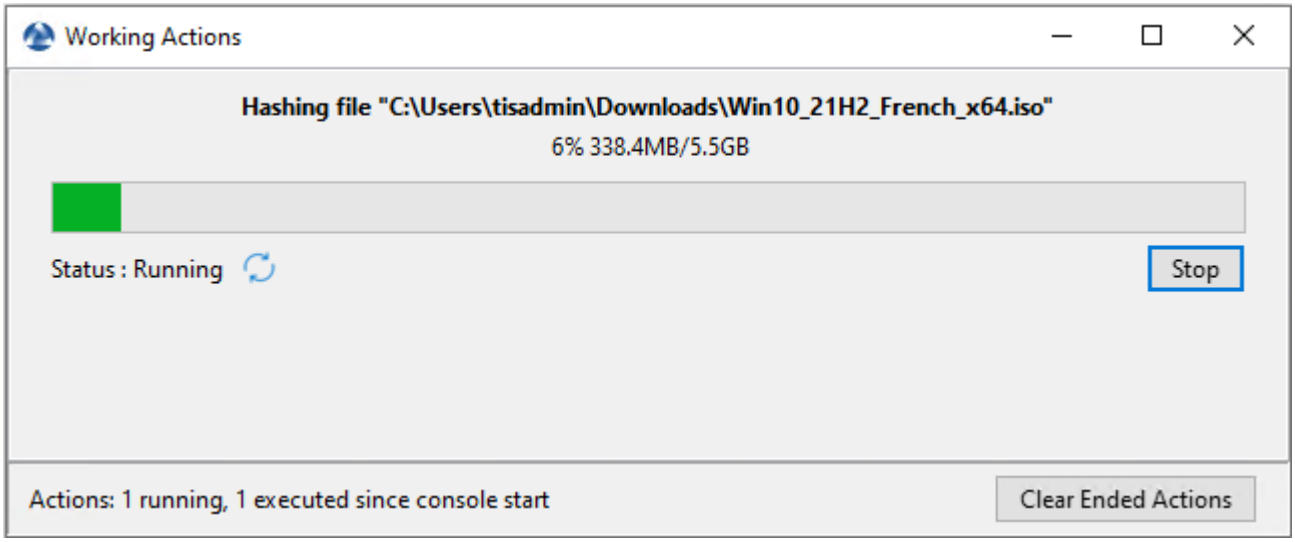


FIG. 7 – Boîte de dialogue informant de la progression de la signature du fichier ISO dans la console WADS

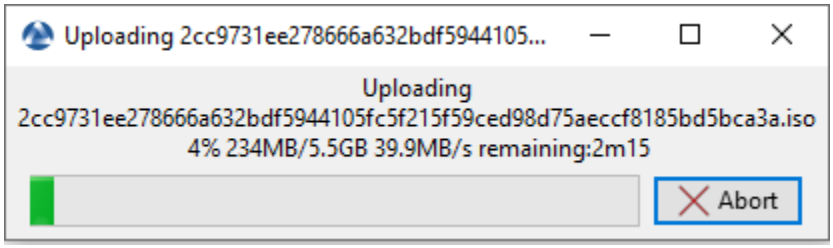
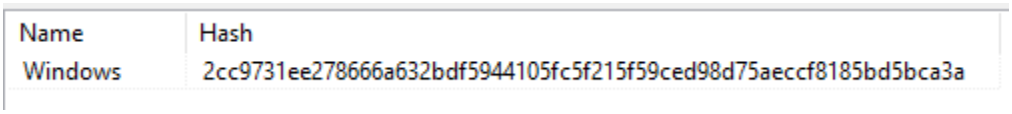


FIG. 8 – Boîte de dialogue informant de la progression du téléchargement du fichier ISO dans la console WADS

— Une fois l'étape de téléchargement terminée avec succès, le fichier `.iso` apparaît dans la section *Installation iso* de la console principale de WADS :



Indication : Il est possible de télécharger plusieurs versions `.iso` de Windows pour différents cas d'utilisation.

44.5 Ajout du fichier de réponse de la configuration XML

L'étape suivante consiste à ajouter le fichier de réponse XML qui sera utilisé pour configurer le déploiement du système d'exploitation Windows.

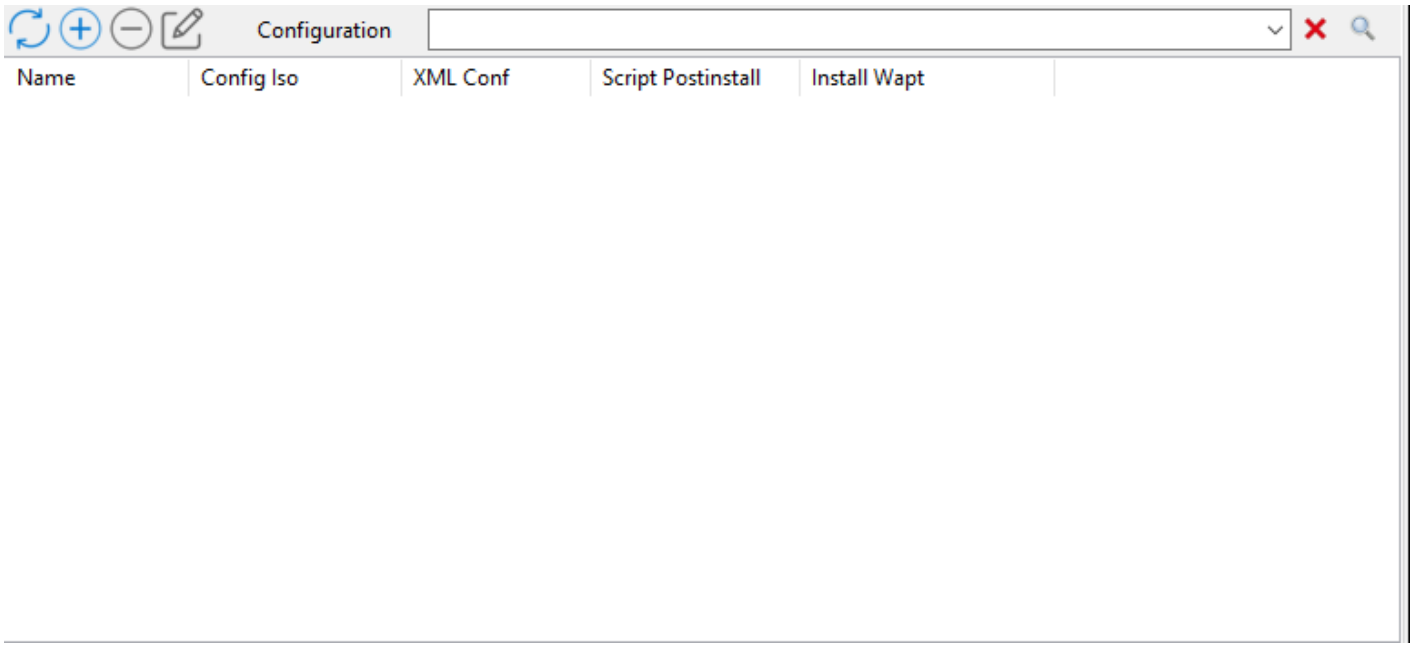


FIG. 9 – Section du fichier de réponse XML de la console WADS

— Dans la section *Configuration*, cliquez sur le bouton + pour configurer le fichier de réponse XML.

TABLEAU 1 – Options pour le fichier de réponse XML 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.
Installer Wapt	Définit s'il faut installer l'agent **WAPT** après l'installation du système d'exploitation.
Choisir le fichier XML	Définit le Fichiers de réponses XML modèle à utiliser.
Choisir un script	Définit un script de post-installation `.bat` à exécuter après l'installation du système d'exploitation.

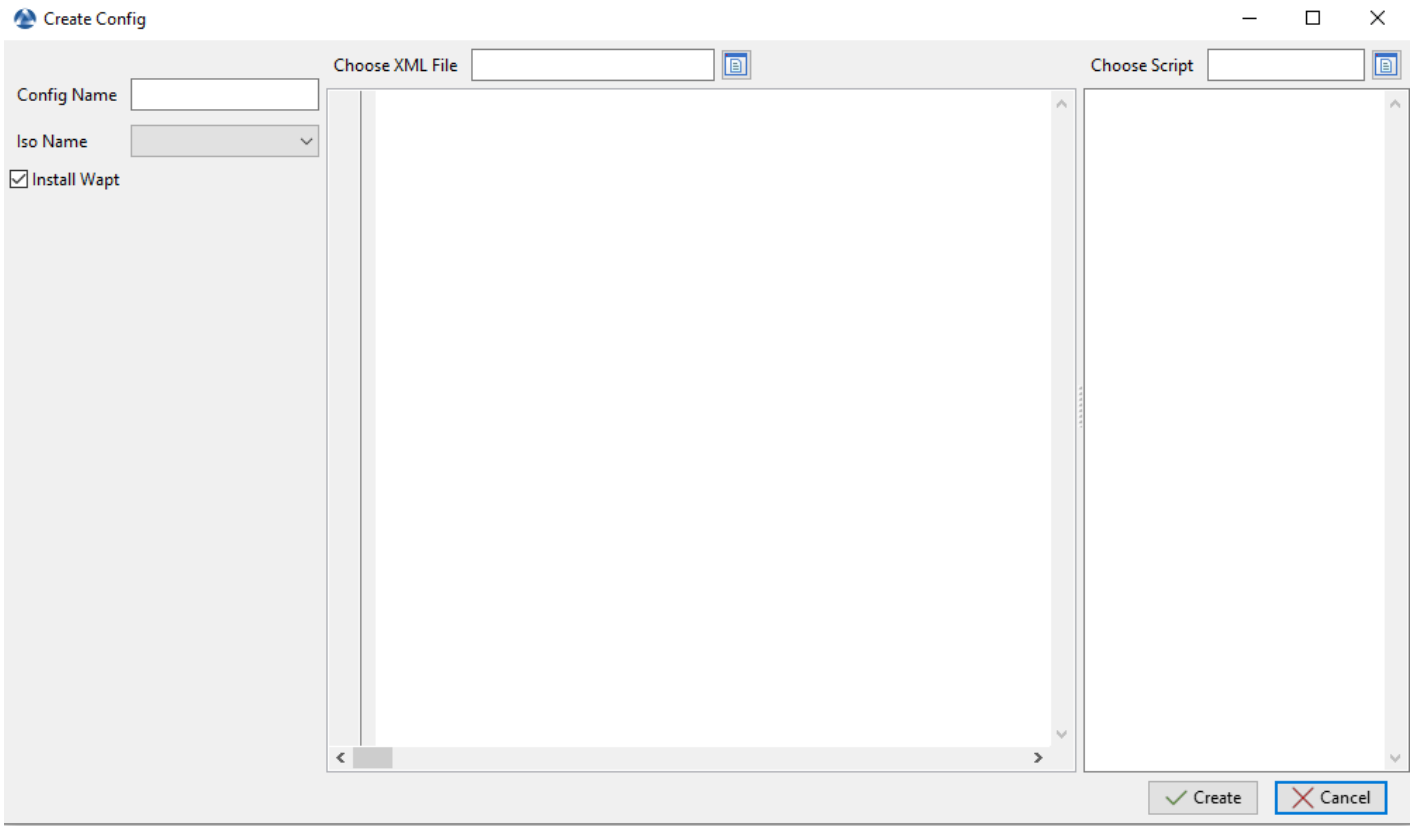


FIG. 10 – Fenêtre de création du fichier de configuration de réponse XML dans la console WADS

— Insérez dans le champ *Config Name* le nom du fichier de réponse XML.
— 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.
— Sélectionnez le modèle de fichier de réponse XML à associer à la configuration de déploiement avec le champ *Choisir un fichier XML*.

Note : Par défaut, WADS intègre *2* types de fichiers de réponses :
— **Offline**pour joindre un ordinateur avec la méthode DirectAccess Offline Domain Join (Djoin)
— **Online**pour joindre un ordinateur sur l'AD
  — Mettez à jour cette partie avec votre **compte de service d'adhésion** :

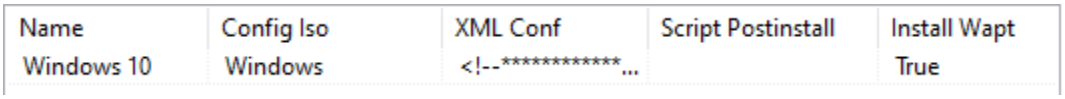
```
<Identification>
  <Credentials>
    <Domain>mydomain.lan</Domain>
    <Password>password</Password>
    <Username>wadsjoin</Username>
  </Credentials>
  <JoinDomain>mydomain.lan</JoinDomain>
</Identification>
```

Indication : Vous pouvez utiliser votre propre fichier de réponses avec WADS.

— Si nécessaire, définissez le script de post-installation dans *Choose Script*, par exemple :

```
"C:\Program Files (x86)\wapt\wapt-get.exe" install tis-firefox-esr
```

— Cliquez sur le bouton *Create* pour créer le fichier de réponse XML.
— Lorsque cela est fait, la configuration apparaît dans la section *Configuration*.



Indication : Il est possible de créer plusieurs configurations de fichiers de réponses XML pour différentes versions de Windows et pour différents cas d'utilisation.

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.

Tableau 2 – 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.

FIG. 11 – Section des pilotes de la console WADS

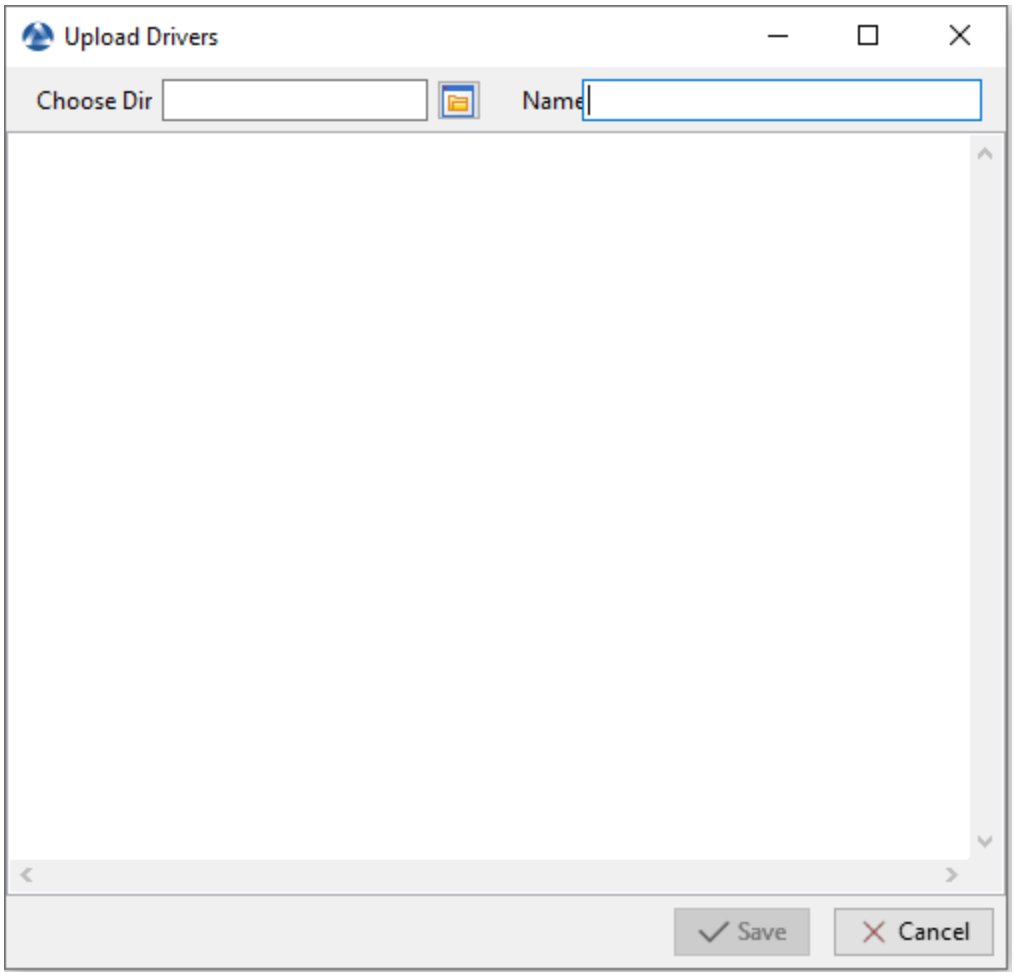


FIG. 12 – Fenêtre pour la création des paquets de pilotes dans la console WADS

— Cliquez sur le bouton *Save*, le téléchargement des bundles de pilotes commence.

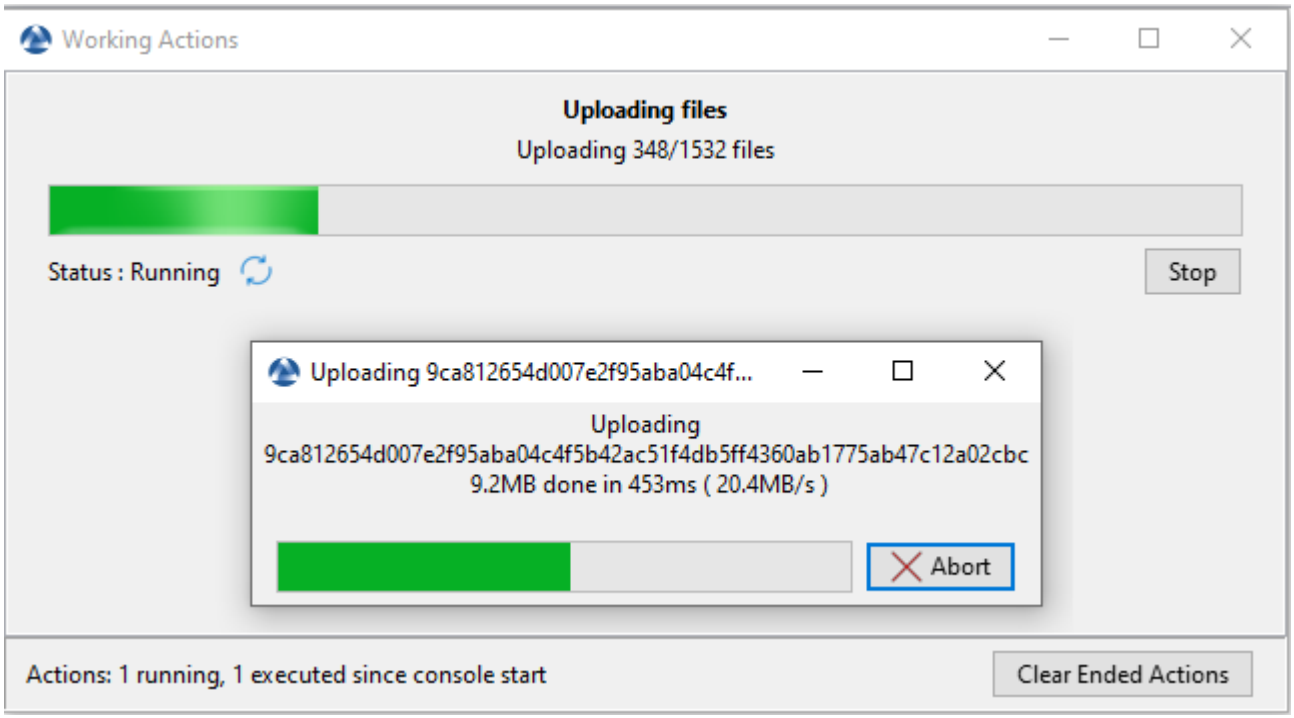


FIG. 13 – Boîte de dialogue informant de la progression du téléchargement des paquets de pilotes dans la console WAPT

— Une fois téléchargé, le pack de pilotes apparaît dans la section *Drivers* de la console WADS.

Indication : Il est possible de créer plusieurs packs de pilotes pour différentes versions de Windows et pour différents cas d'utilisation.

Note :

— 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**.

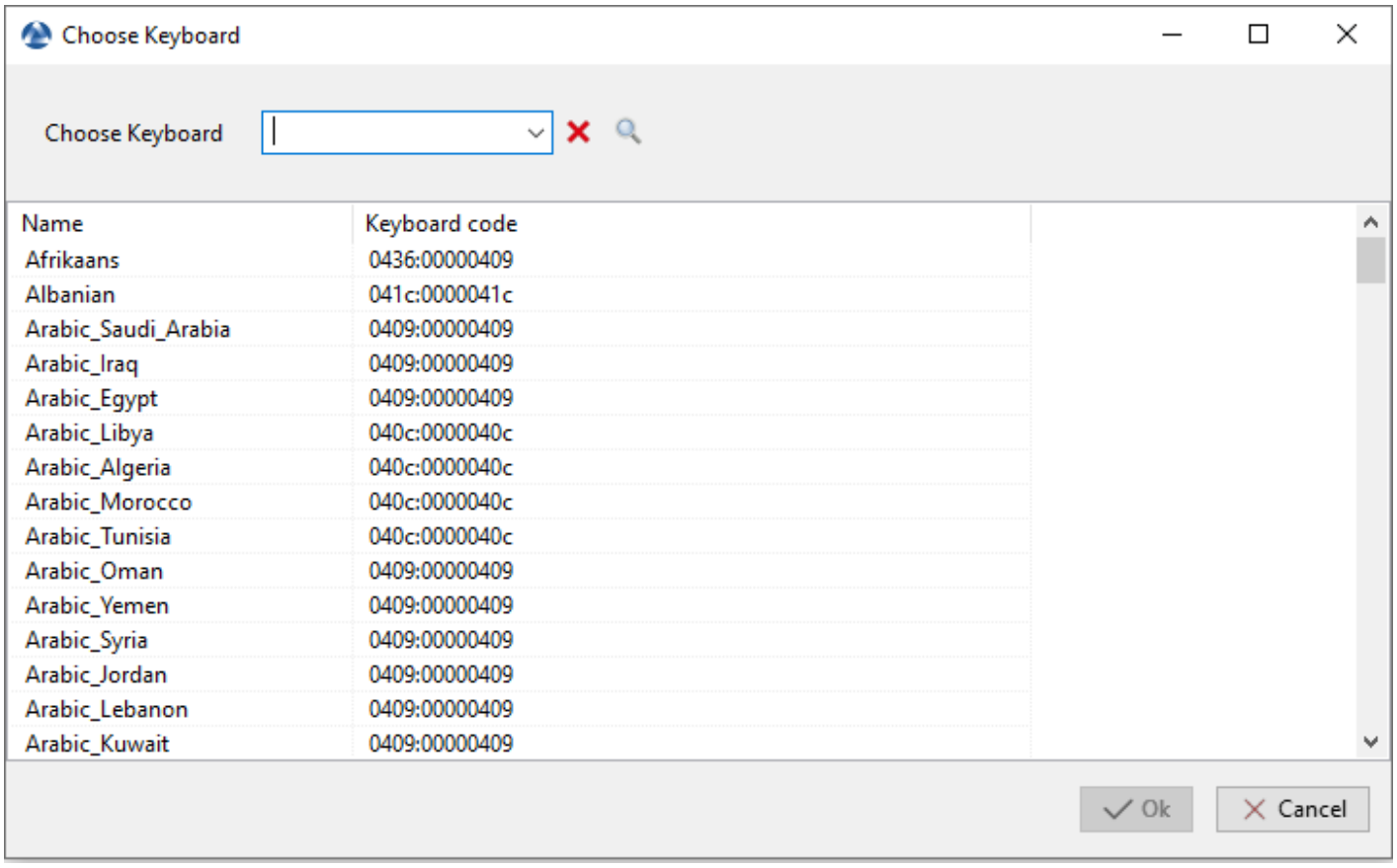


FIG. 14 – Boîte de dialogue pour sélectionner le clavier dans la console WADS

- Sélectionnez 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*.

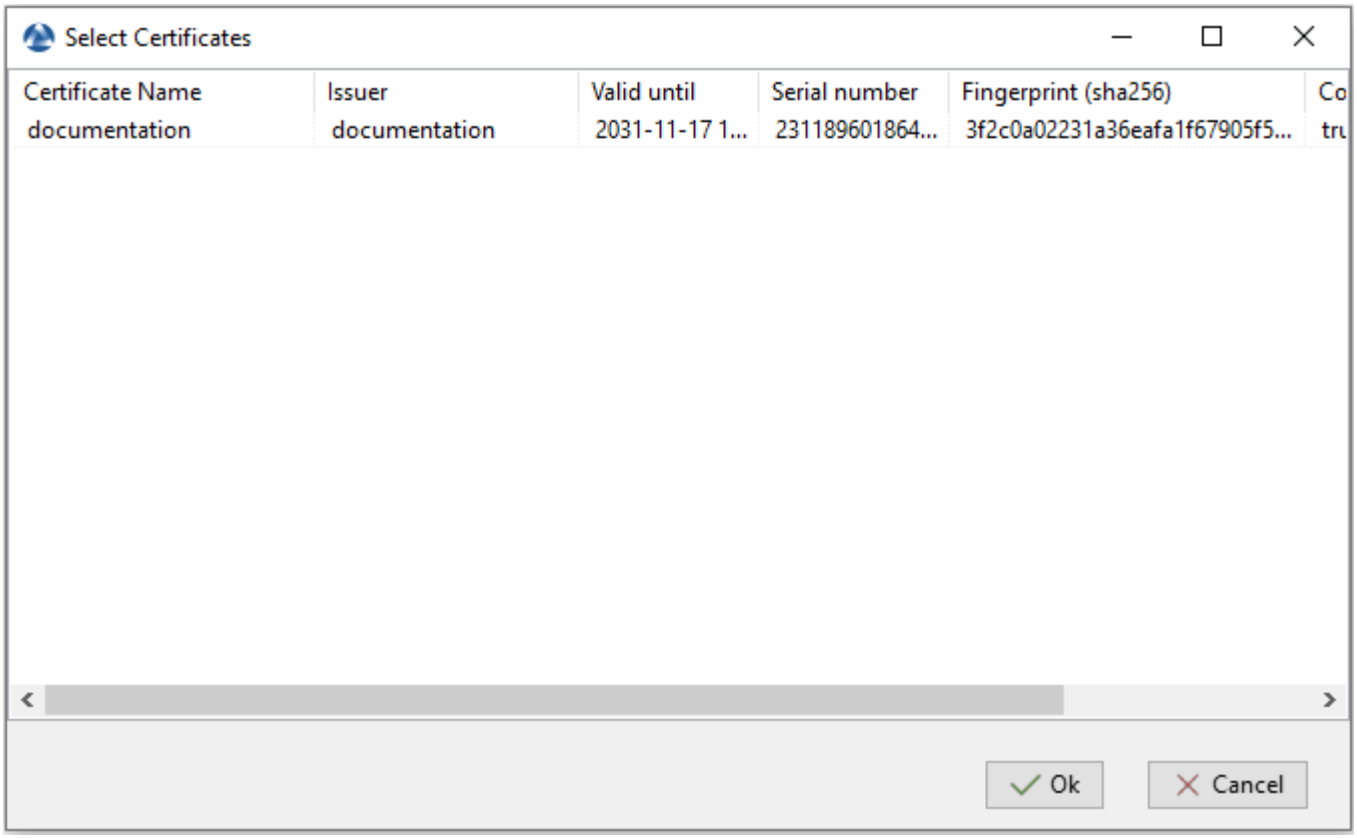


FIG. 15 – Boîte de dialogue pour la sélection du certificat dans la Console WADS

44.7.2 Démarrage de l'hôte avec le réseau

Note : 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 :

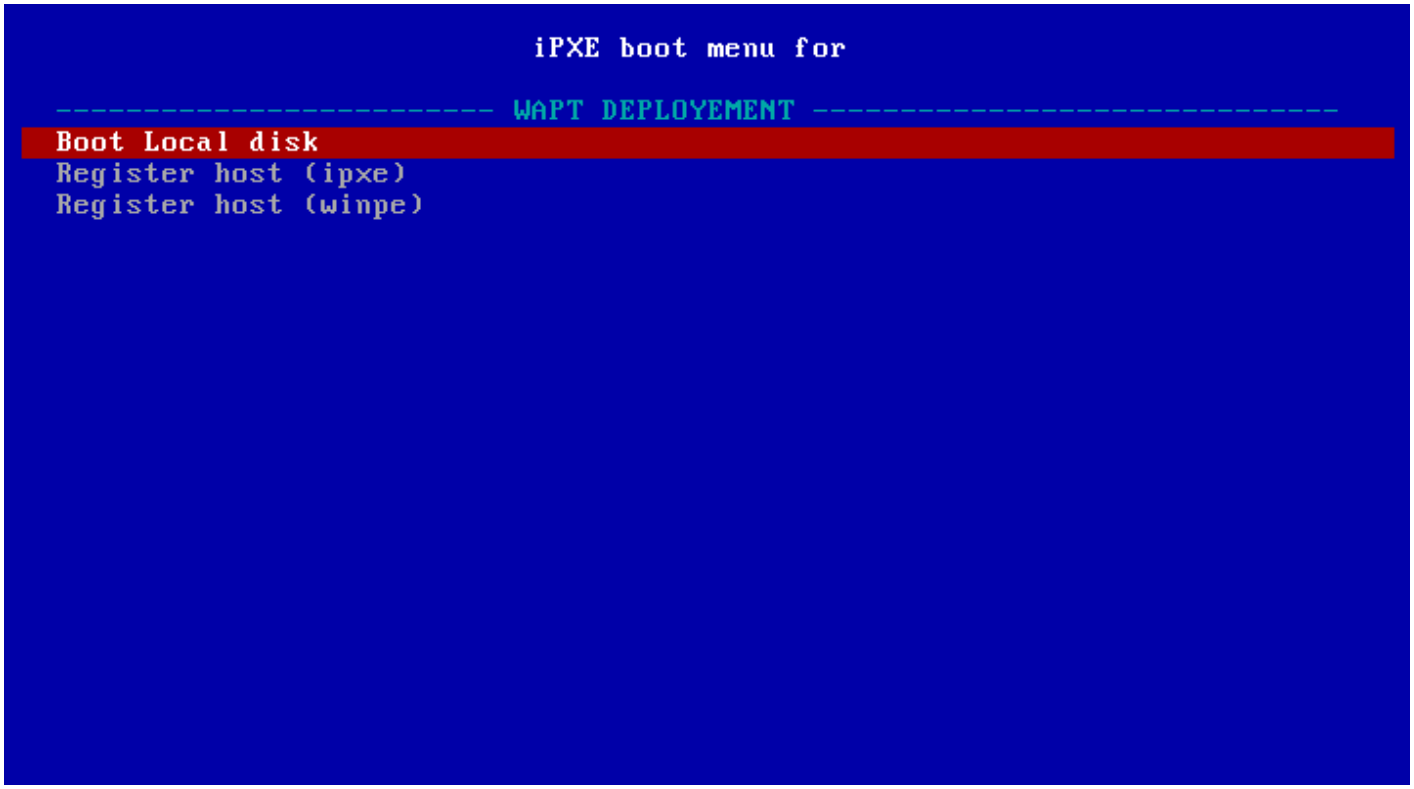


FIG. 16 – fenêtre du menu de démarrage iPXE

- *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 :

FIG. 17 – Fenêtre de terminal texte demandant un nom d'hôte lors de l'enregistrement par la méthode iPXE

Avertissement : Le clavier est de type qwerty

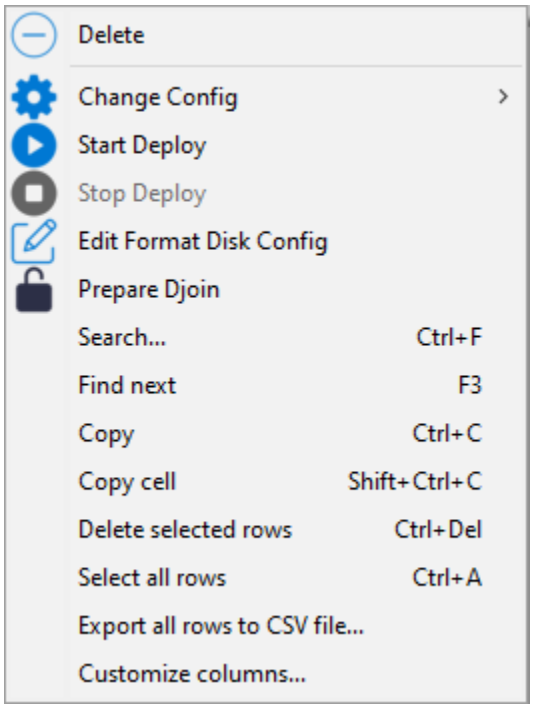
— Rafraîchissez la console WADS avec `F5`, l'hôte apparaît dans l'onglet *OS Deploy*.

Hostname	Mac Addresses	Status	Waiting to Deploy
documentation	08:00:27:19:0D:4B		False

FIG. 18 – 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

Avertissement : Si l'hôte doit être membre du domaine Active Directory, définissez les informations soit en utilisant :

— *un fichier de réponses* ;

— *la méthode Djoin*.

— 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 :

FIG. 19 – Fenêtre de terminal texte demandant un nom d'hôte lors de l'enregistrement à l'aide de la méthode WinPE

Note : Le clavier est dans la même disposition que celle définie lors de l'étape *WinPE* de cette documentation.

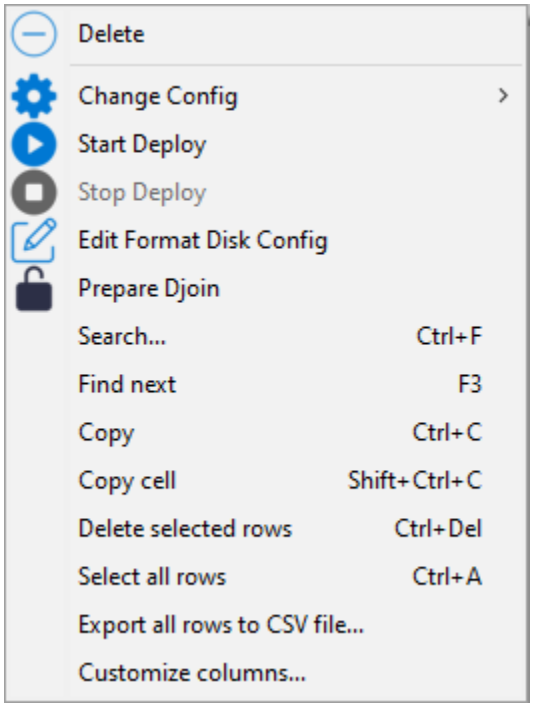
— Rafraîchissez la console WADS avec `F5`, l'hôte apparaît dans l'onglet *OS Deploy*.

Hostname	Mac Addresses	Status	Waiting to Deploy
documentation	08:00:27:19:0D:4B		False

FIG. 20 – 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

Avertissement : Si l'hôte doit être membre du domaine Active Directory, définissez les informations soit en utilisant :
- *un fichier de réponses* ;
- *la méthode Djoin*.

— Redémarrez l'hôte avec la même option de démarrage que précédemment (USB ou LAN), Windows commencera à s'installer.

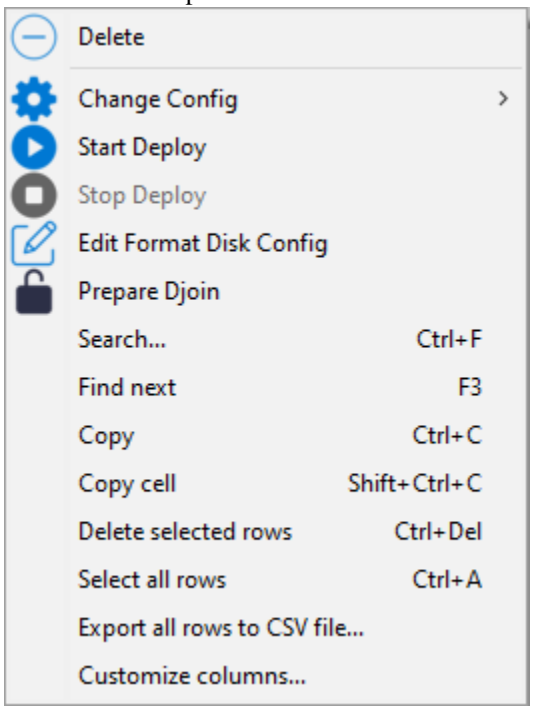
— Lorsque l'installation est terminée, l'onglet *OS Deploy*, l'état passe à `Done`.

44.8.1 Joindre l'hôte à un domaine Active Directory

Méthode en ligne 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.

— Le fichier `Djoin` est prêt à être utilisé pour joindre l'hôte en tant que membre du domaine Active Directory.

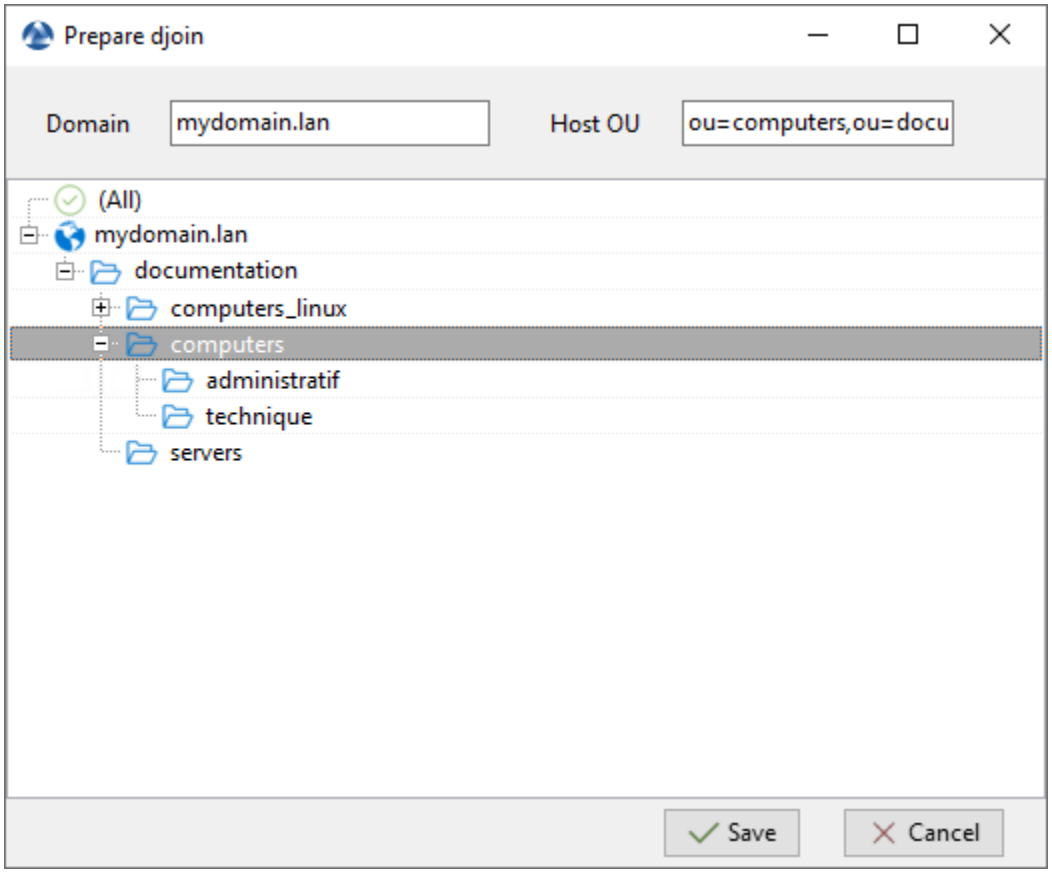


FIG. 21 – Sélection de l'unité organisationnelle à laquelle rattacher automatiquement l'hôte réimagé

CHAPITRE 45

Créer des paquets WAPT

Pour plus d'informations sur la structure du packaging, veuillez vous référer à <wapt-package-structure>

45.1 Créer son environment 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 environment 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'environment 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 ;
— 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* :

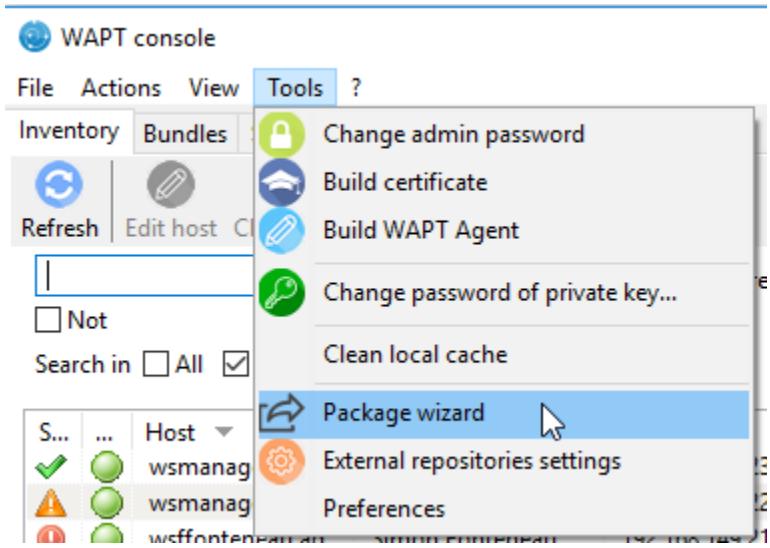


FIG. 1 – Les principes de création d'un paquet WAPT à partir d'un modèle depuis la console

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.

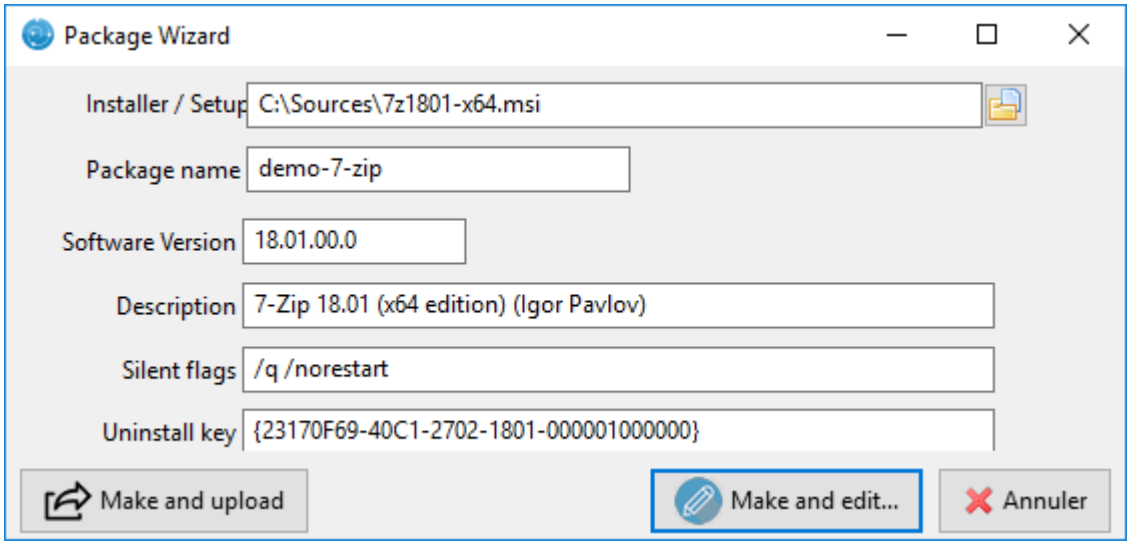


FIG. 2 – Boîte de dialogue demandant des informations lors de la création du packaging WAPT dans la console WAPT

— 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..*

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*.

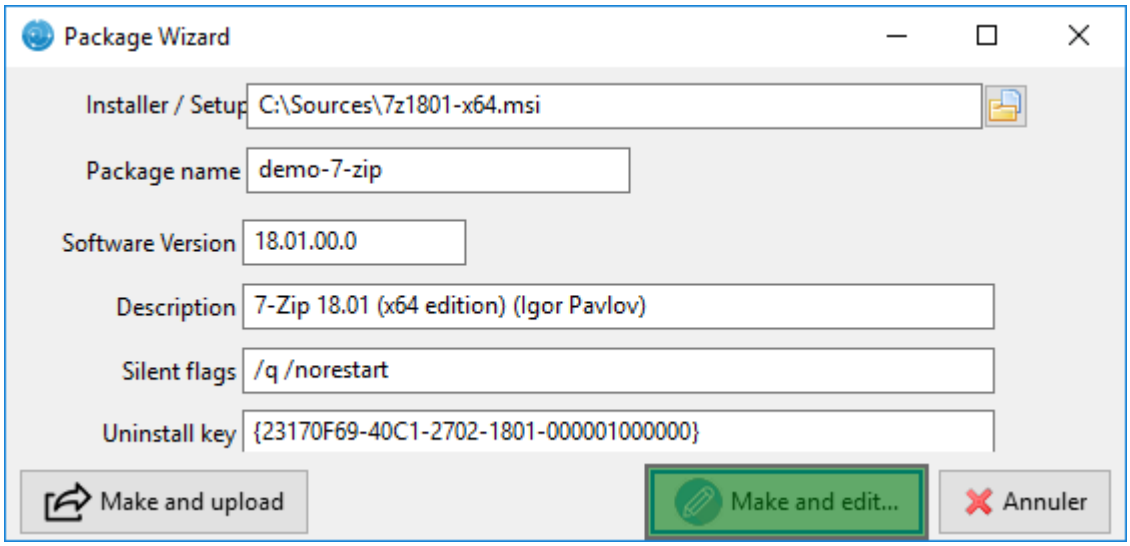


FIG. 3 – Boîte de dialogue mettant en évidence le bouton « Make and edit … » lors de la création du packaging WAPT dans la console WAPT

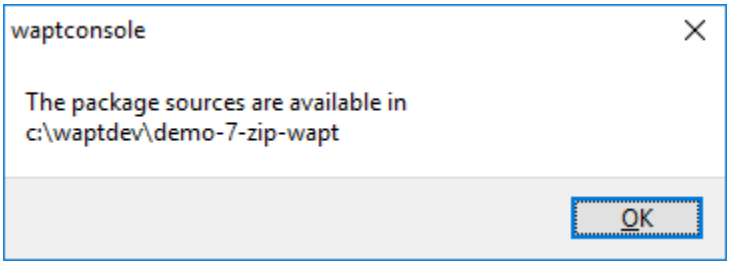


FIG. 4 – Fenêtre de message indiquant dans la console WAPT que le packaging WAPT a été téléchargé dans le référentiel WAPT

L'IDE **PyScripter** se lance et permet d'éditer les fichiers du paquet.

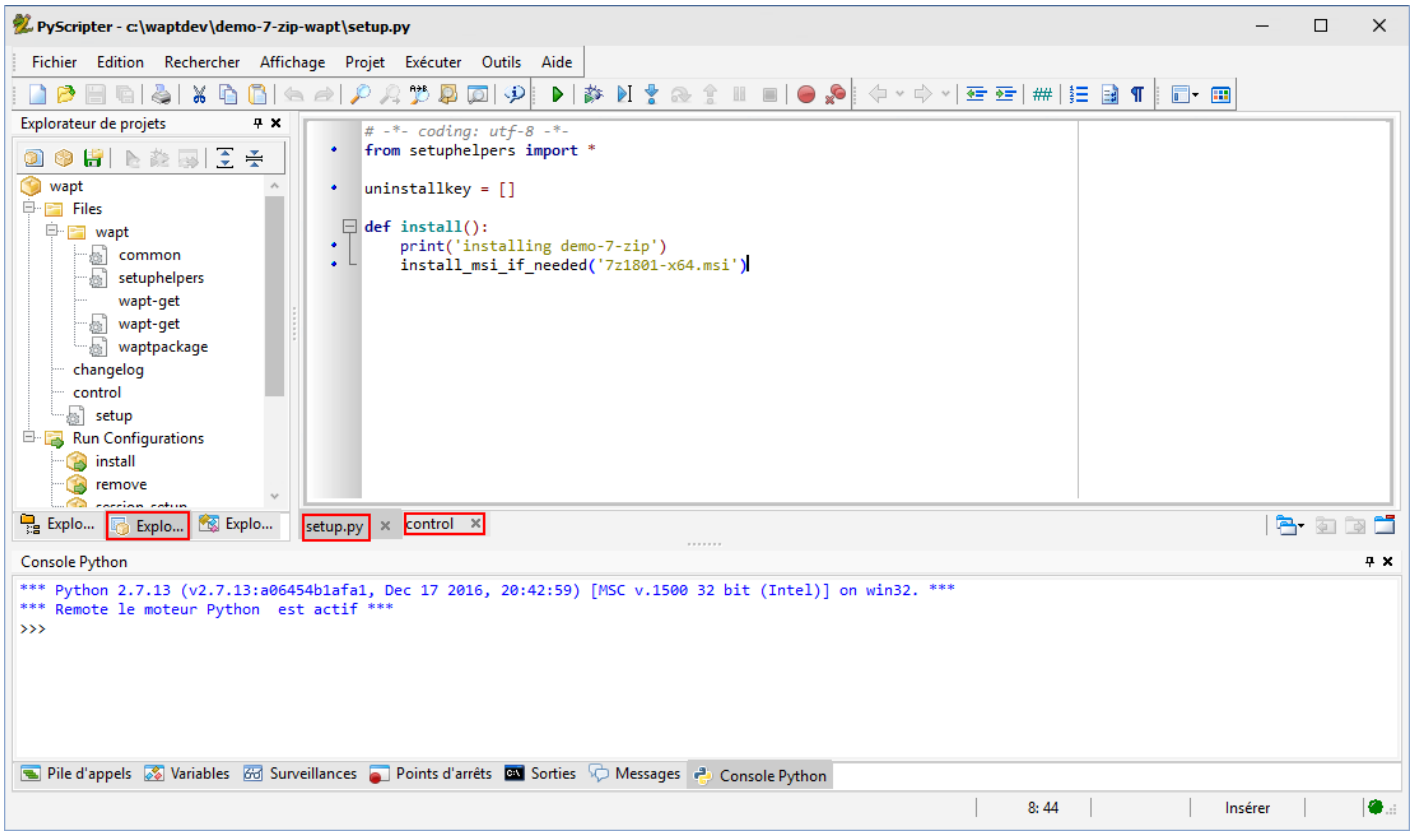


FIG. 5 – PyScripter - Personnalisation du paquet avec PyScripter

45.3 Présentation de PyScripter

45.3.1 L'explorateur de projets PyScripter

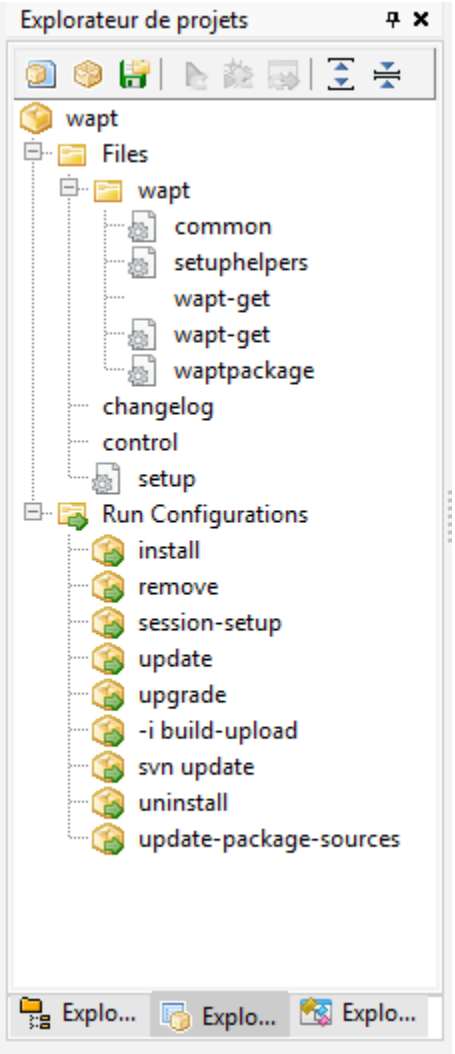


FIG. 6 – PyScripter - Navigation d'un projet dans l'explorateur de fichiers de PyScrypter

L'explorateur de projets PyScripter liste les différents fichiers dont vous pouvez avoir besoin, notamment le fichier `control` et le fichier `setup.py`.

45.3.2 Run Configurations

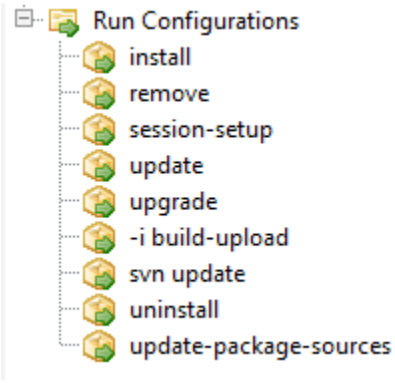


FIG. 7 – PyScripter - Naviguer dans les configurations d'exécution d'un projet dans PyScripter

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.

45.3.3 Zone d'édition

La Zone d'édition de **PyScripter** permet d'éditer le fichier `setup.py` ainsi que le fichier `control`.

45.3.4 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*.

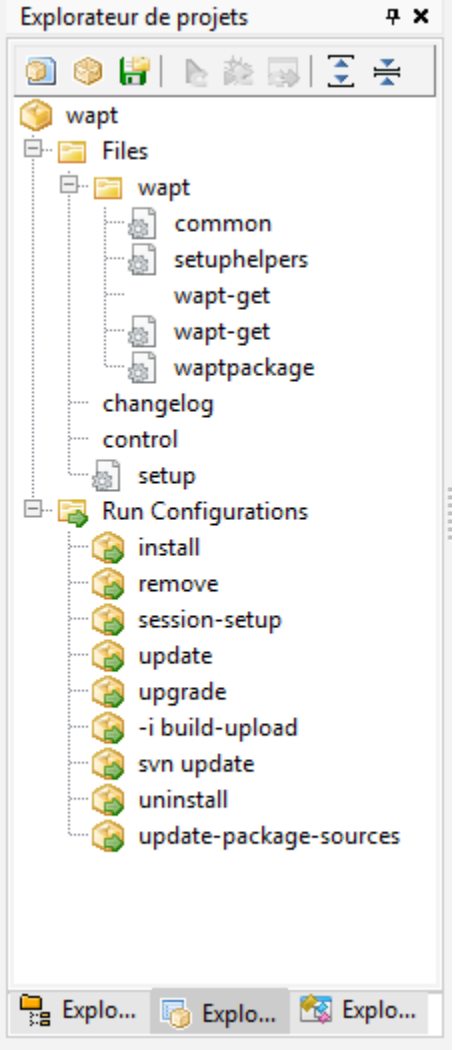


FIG. 8 – PyScripter - Personnalisation du paquet avec PyScripter

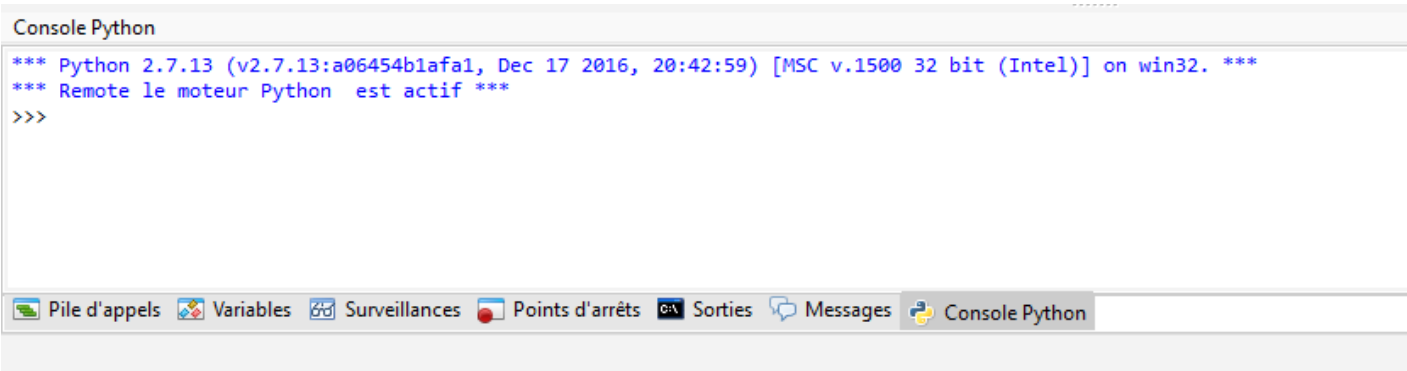
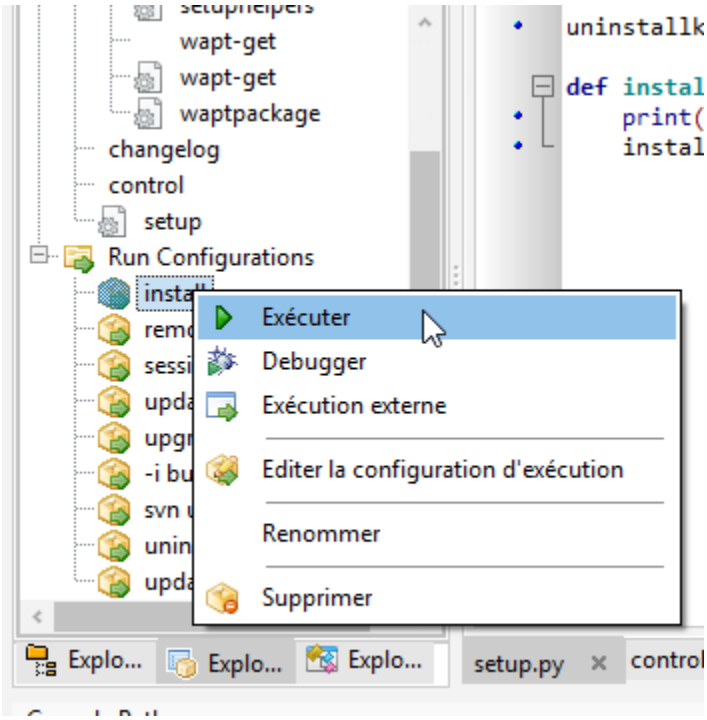


FIG. 9 – PyScripter - console python de PyScripter

45.3.5 Tester localement l'installation du paquet WAPT

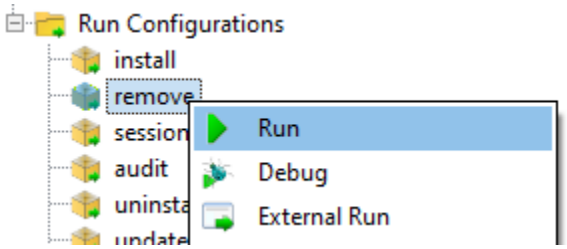
Vous pouvez ensuite tester le lancement d'une installation sur votre station de développement.



La Console PyScripter vous permet de vérifier si l'installation s'est bien déroulée.

45.3.6 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.

45.4 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.

Tableau 1 – Liste des arguments disponibles avec *install_exe_if_need*

Options (Option par défaut)	Description
`min_os_version`	version minimale au dessus de laquelle il mettra à jour.
`killbefore` (par défaut `None`)	liste des programmes à tuer avant de lancer l'installation.
`accept_returncodes` (par défaut `[0,3010]`)	codes de retour autres que 0 ou 3010 acceptés en retour par la fonction.
`timeout` (par défaut `300`)	durée d'attente maximale d'installation (en secondes).
`propriétés` (par défaut `None`)	propriétés supplémentaires à passer en argument au MSI pour l'installation.
`get_version` (par défaut `None`)	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

Note : 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`.

Indication : 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)
```

45.4.1 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}"]
```

(suite sur la page suivante)

(suite de la page précédente)

```
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')
```

45.4.2 Vidéo de démonstration

https://youtu.be/Z6wr6emPGCU

45.5 Packager des .exe (exemple)

— Télécharger l'installateur *.exe* à partir d'une source fiable.
Téléchargez 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 <https://chocolatey.org/packages/FirefoxESR>`_ .
    — 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*.
— 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
```

(suite sur la page suivante)

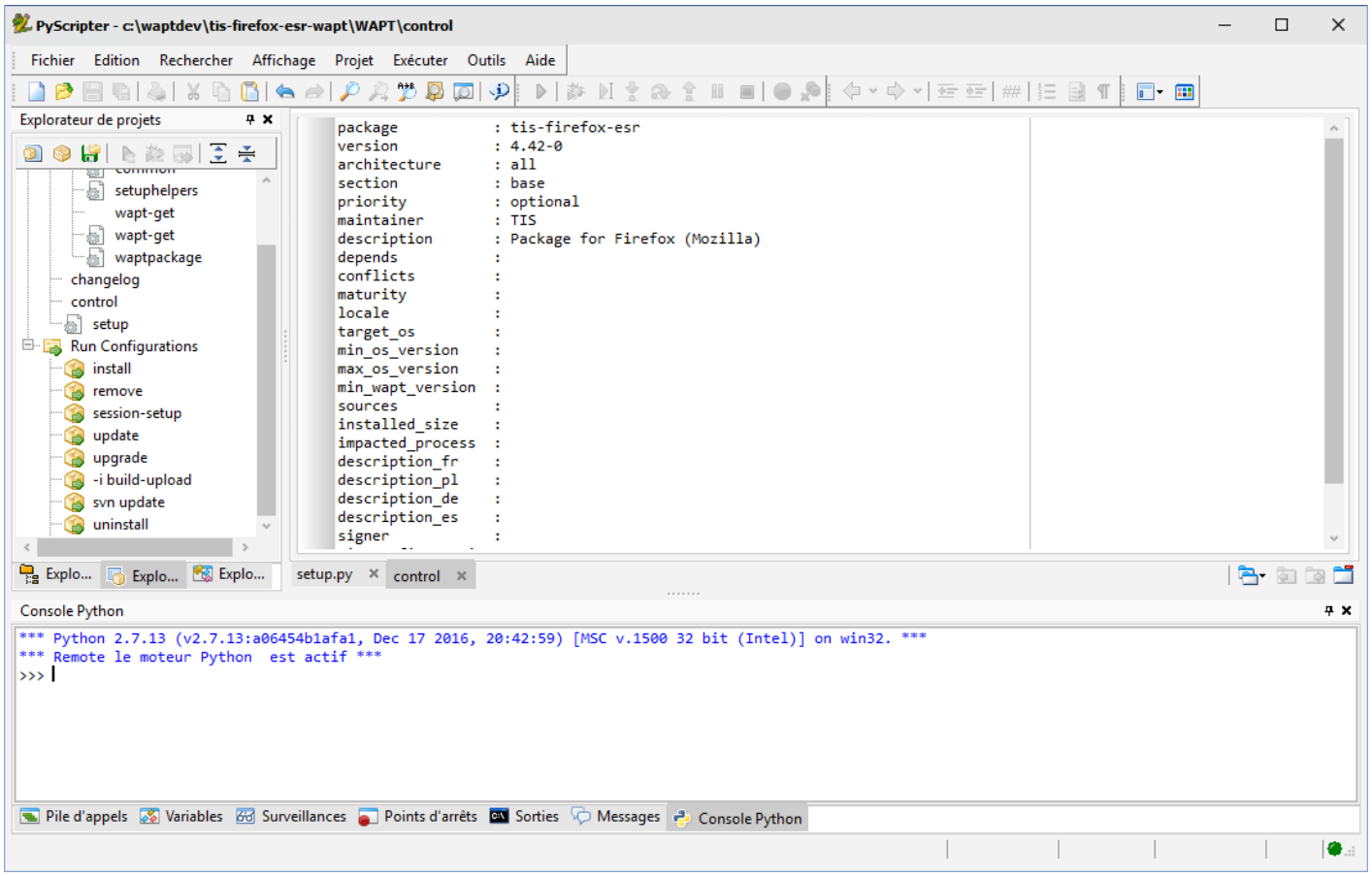


FIG. 10 – PyScripter - Ouverture du packaging WAPT de FirefoxESR

(suite de la page précédente)

```
description       : Mozilla Firefox 52.6.0 ESR
impacted_process  : firefox.exe
```

Note : 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")
```

Tableau 2 – 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`	version minimale au dessus de laquelle il mettra à jour.
`killbefore` (par défaut `None`)	liste des programmes à tuer avant de lancer l'installation.
`accept_returncodes` (par défaut `[0,3010]`)	codes de retour autres que 0 ou 3010 acceptés en retour par la fonction.
`timeout` (par défaut `300`)	durée d'attente maximale d'installation (en secondes).
`get_version` (par défaut `None`)	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

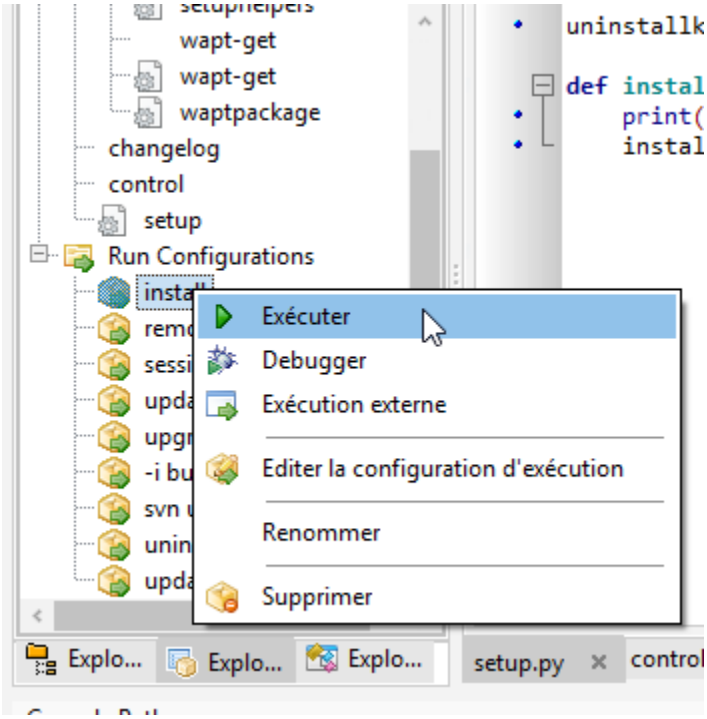
Le paquet aura alors ce comportement :
- Le logiciel Firefox s'installera uniquement si le logiciel n'est pas installé et si la version est strictement inférieure à 45.5.0, sauf si l'option `--force` est indiquée lors de l'installation du paquet.
- A l'installation, les processus **`firefox.exe`** en 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**.

45.5.1 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.

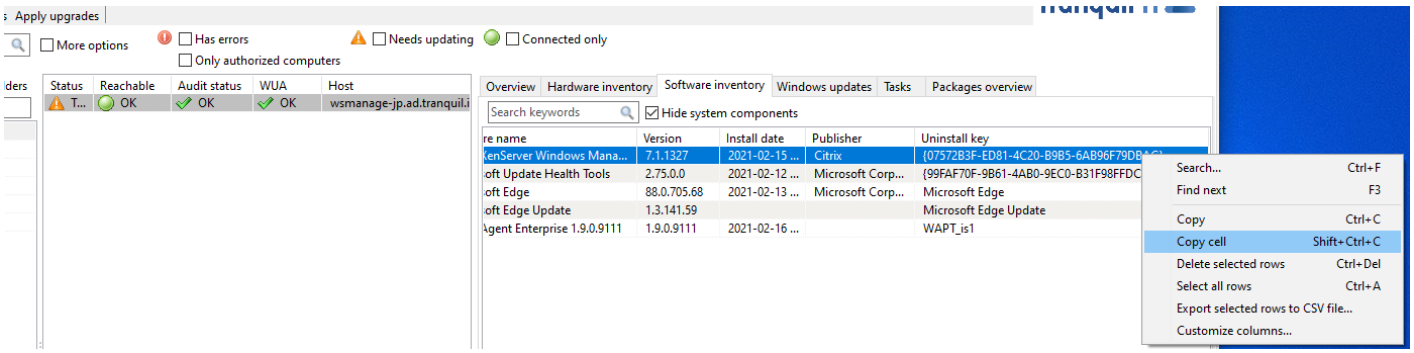


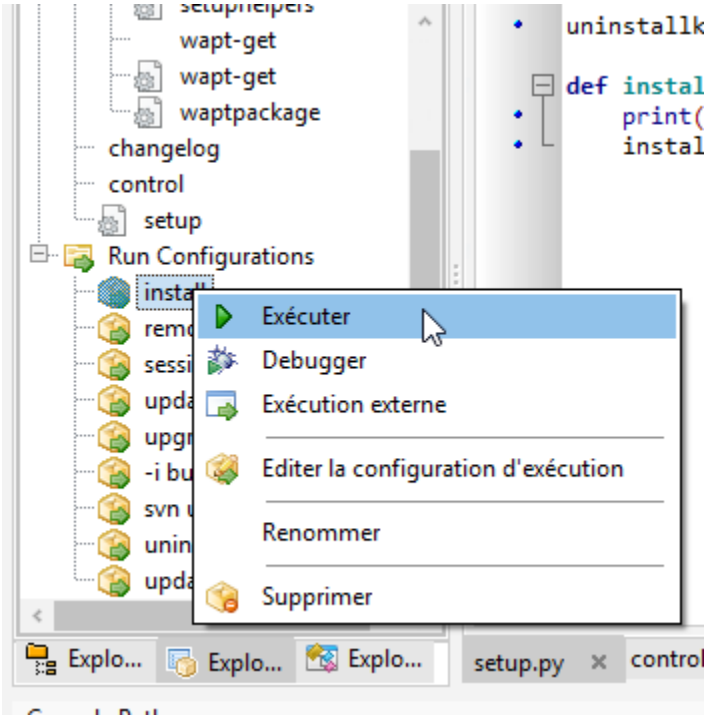
Fig. 11 – Récupérer une clé de désinstallation depuis la console

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 = []

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")
```

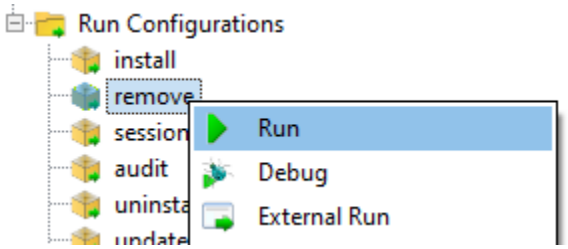
Pour tester que votre clé fonctionne correctement, vous devez relancer une installation dans **pyscripter**.



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
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.

45.5.2 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 ...

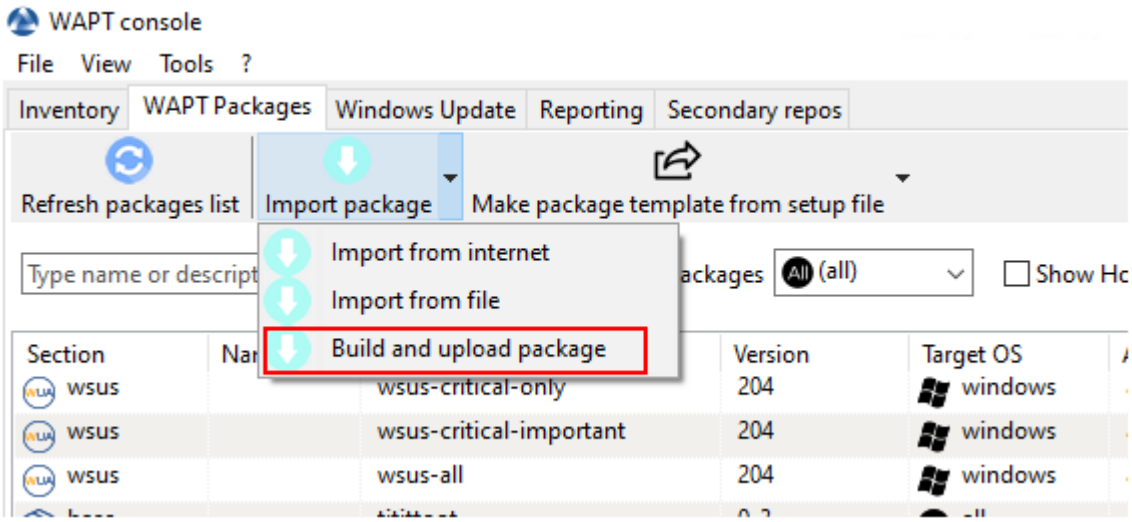
45.5.3 Vidéo de démonstration

https://youtu.be/z_EN2CBCTcY

45.6 Packager des paquets linux simples

45.7 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.

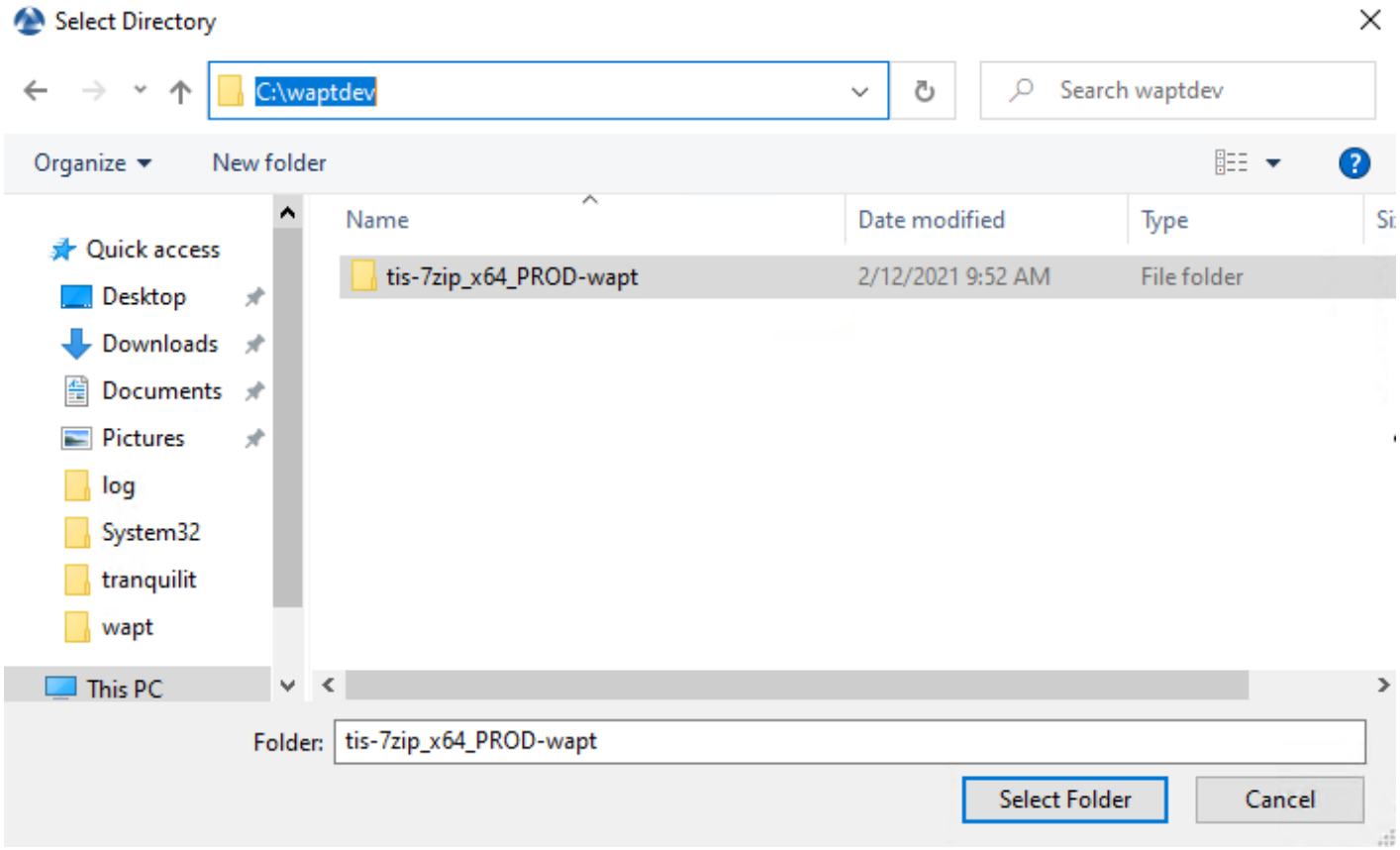


FIG. 12 – Fenêtre du navigateur permettant de sélectionner le packaging WAPT à importer dans le référentiel privé

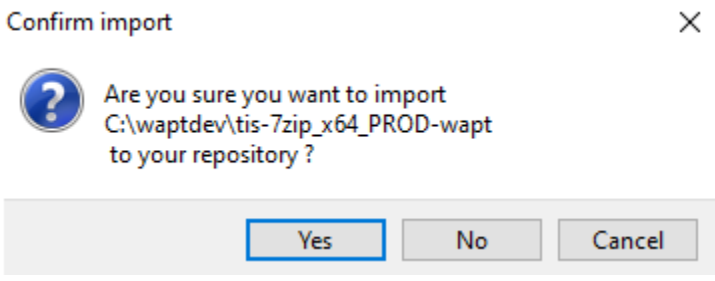


FIG. 13 – Boîte de dialogue de la console WAPT pour confirmer l'importation d'un packaging WAPT dans le référentiel privé

45.7.1 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.

```
# -*- 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>`_.

45.8 Exemples simples de fonctions du setuphelper couramment utilisées

Nous présentons ici quelques fonctions implémentées dans *Setuphelpers* et fréquemment utilisées pour développer des paquets WAPT.

45.8.1 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')
```

Indication : Pour plus d'informations ou pour connaître les paramètres complémentaires de la commande, consultez la documentation de référence (en anglais) en visitant :

https://www.wapt.fr/en/api-doc-1.5/source/setuphelpers.html?highlight=makepath#setuphelpers.makepath

Créer et supprimer des raccourcis

... crée le répertoire `C:\test`.

```
mkdirs('C:\\test')
```

Indication : Pour plus d'informations ou pour connaître les paramètres complémentaires de la commande, consultez la documentation de référence (en anglais) en visitant :

https://www.wapt.fr/en/api-doc-1.5/source/setuphelpers.html?highlight=mkdirs#setuphelpers.mkdirs

... détruit le répertoire `C:\tmp\target`.

```
remove_tree(r'C:\tmp\target')
```

Indication : Pour plus d'informations ou pour connaître les paramètres complémentaires de la commande, consultez la documentation de référence (en anglais) en visitant :

https://www.wapt.fr/en/api-doc-1.5/source/setuphelpers.html?highlight=remove_tree#setuphelpers.remove_tree

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')
```

Indication : Pour plus d'informations ou pour connaître les paramètres complémentaires de la commande, consultez la documentation de référence (en anglais) en visitant :

https://www.wapt.fr/en/api-doc-1.5/source/setuphelpers.html?highlight=isdir#setuphelpers.isdir

... vérifie que `C:\Program Files (x86)\software\file` est un fichier.

```
isfile(makepath(programfiles32,'software','file')):
    print('file exist')
```

Indication : Pour plus d'informations ou pour connaître les paramètres complémentaires de la commande, consultez la documentation de référence (en anglais) en visitant :

https://www.wapt.fr/en/api-doc-1.5/source/setuphelpers.html?highlight=isfile#setuphelpers.isfile

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')
```

Indication : Pour plus d'informations ou pour connaître les paramètres complémentaires de la commande, consultez la documentation de référence (en anglais) en visitant :

https://www.wapt.fr/en/api-doc-1.5/source/setuphelpers.html?highlight=dir_is_empty#setuphelpers.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'))
```

Indication : Pour plus d'informations ou pour connaître les paramètres complémentaires de la commande, consultez la documentation de référence (en anglais) en visitant :

https://www.wapt.fr/en/api-doc-1.5/source/setuphelpers.html?highlight=filecopyto#setuphelpers.filecopyto

Copier un dossier

... copie le dossier `sources` dans le répertoire `C:\projet`.

```
copytree2('sources','C:\\projet')
```

Indication : Pour plus d'informations ou pour connaître les paramètres complémentaires de la commande, consultez la documentation de référence (en anglais) en visitant :

https://www.wapt.fr/en/api-doc-1.5/source/setuphelpers.html?highlight=copytree2#setuphelpers.copytree2

Récupérer la version d'un fichier

La commande **`get_file_properties`** ...

```
get_file_properties(makepath(programfiles32,'InfraRecorder','infrarecorder.exe'))['ProductVersion']
```

Indication : Pour plus d'informations ou pour connaître les paramètres complémentaires de la commande, consultez la documentation de référence (en anglais) en visitant :

https://www.wapt.fr/en/api-doc-1.5/source/setuphelpers.html?highlight=get_file_properties#setuphelpers.get_file_properties

---

45.9 Manipulation de clés de registre

45.9.1 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')
```

Indication : Pour plus d'informations ou pour connaître les paramètres complémentaires de la commande, consultez la documentation de référence (en anglais) en visitant :

https://www.wapt.fr/en/api-doc-1.5/source/setuphelpers.html?highlight=registry_readstring#setuphelpers.registry_readstring

---

45.9.2 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'))
```

Indication : Pour plus d'informations ou pour connaître les paramètres complémentaires de la commande, consultez la documentation de référence (en anglais) en visitant :

https://www.wapt.fr/en/api-doc-1.5/source/setuphelpers.html?highlight=registry_readstring#setuphelpers.registry_readstring

45.9.3 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)
```

Indication : Pour plus d'informations ou pour connaître les paramètres complémentaires de la commande, consultez la documentation de référence (en anglais) en visitant :

https://www.wapt.fr/en/api-doc-1.5/source/setuphelpers.html?highlight=registry_setstring#setuphelpers.registry_setstring

45.10 Créer et supprimer des raccourcis

Avec WAPT setuphelper, il est possible de créer différents types de raccourcis.

45.10.1 Créer et supprimer des raccourcis

créer un raccourci personnalisé sur le bureau de l'utilisateur en cours

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')
```

Indication : Pour plus d'informations ou pour connaître les paramètres complémentaires de la commande, consultez la documentation de référence (en anglais) en visitant :

https://www.wapt.fr/en/api-doc-1.5/source/setuphelpers.html?highlight=create_desktop_shortcut#setuphelpers.create_desktop_shortcut

supprimer des raccourcis

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')
```

Indication : Pour plus d'informations ou pour connaître les paramètres complémentaires de la commande, consultez la documentation de référence (en anglais) en visitant :

https://www.wapt.fr/en/api-doc-1.5/source/setuphelpers.html?highlight=remove_desktop_shortcut#setuphelpers.remove_desktop_shortcut

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():
    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 personnalisé sur le bureau de l'utilisateur en cours

Un raccourci dans le menu démarrer

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')
```

Indication : Pour plus d'informations ou pour connaître les paramètres complémentaires de la commande, consultez la documentation de référence (en anglais) en visitant :

https://www.wapt.fr/en/api-doc-1.5/source/setuphelpers.html?highlight=create_desktop_shortcut#setuphelpers.create_desktop_shortcut

---

supprimer des raccourcis

La commande **remove_programs_menu_shortcut** supprime le raccourci *WAPT Console Management* du menu démarrer.

```
remove_programs_menu_shortcut('WAPT Console Management')
```

Indication : Pour plus d'informations ou pour connaître les paramètres complémentaires de la commande, consultez la documentation de référence (en anglais) en visitant :

https://www.wapt.fr/en/api-doc-1.5/source/setuphelpers.html?highlight=remove_desktop_shortcut#setuphelpers.remove_desktop_shortcut

45.10.2 Créer un raccourci personnalisé sur le bureau de l'utilisateur en cours

créer un raccourci personnalisé sur le bureau de l'utilisateur en cours

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')
```

Indication : Pour plus d'informations ou pour connaître les paramètres complémentaires de la commande, consultez la documentation de référence (en anglais) en visitant :

https://www.wapt.fr/en/api-doc-1.5/source/setuphelpers.html?highlight=create_user_desktop_shortcut#setuphelpers.create_user_desktop_shortcut

supprimer un raccourci personnalisé sur le bureau de l'utilisateur en cours

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')
```

Indication : Pour plus d'informations ou pour connaître les paramètres complémentaires de la commande, consultez la documentation de référence (en anglais) en visitant :

https://www.wapt.fr/en/api-doc-1.5/source/setuphelpers.html?highlight=remove_user_desktop_shortcut#setuphelpers.remove_user_desktop_shortcut

créer un raccourci personnalisé sur le bureau de l'utilisateur en cours

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')
```

Indication : Pour plus d'informations ou pour connaître les paramètres complémentaires de la commande, consultez la documentation de référence (en anglais) en visitant :

https://www.wapt.fr/en/api-doc-1.5/source/setuphelpers.html?highlight=create_user_desktop_shortcut#setuphelpers.create_user_desktop_shortcut

supprimer un raccourci personnalisé sur le bureau de l'utilisateur en cours

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')
```

Indication : Pour plus d'informations ou pour connaître les paramètres complémentaires de la commande, consultez la documentation de référence (en anglais) en visitant :

https://www.wapt.fr/en/api-doc-1.5/source/setuphelpers.html?highlight=remove_user_desktop_shortcut#setuphelpers.remove_user_desktop_shortcut

45.11 Environnement Windows / Logiciel / Services

45.11.1 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 : Pour plus d'informations ou pour connaître les paramètres complémentaires de la commande, consultez la documentation de référence (en anglais) en visitant :

https://www.wapt.fr/en/api-doc-1.5/source/setuphelpers.html?highlight=windows_version#setuphelpers.windows_version

Visitez également le site sur les numéros de version de Windows.

45.11.2 Vérifier si l'architecture est en 64bits

… vérifie que le processeur de la machine est 64bits.

```
if iswin64():
    print('Pc x64')
else:
    print('Pc not x64')
```

Indication : Pour plus d'informations ou pour connaître les paramètres complémentaires de la commande, consultez la documentation de référence (en anglais) en visitant :

https://www.wapt.fr/en/api-doc-1.5/source/setuphelpers.html?highlight=iswin64#setuphelpers.iswin64

45.11.3 La variable Program Files

— **vim**;

```
print(programfiles())
```

— **vim**;

```
print(programfiles32())
```

— **vim**;

```
print(programfiles64())
```

Renvoie les différentes localisations de *Program Files*

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.

45.11.4 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())
```

Indication : Pour plus d'informations ou pour connaître les paramètres complémentaires de la commande, consultez la documentation de référence (en anglais) en visitant :

https://www.wapt.fr/en/api-doc-1.5/source/setuphelpers.html?highlight=user_appdata#setuphelpers.user_appdata

… renvoie le profil *appdata* local de l'utilisateur courant (`C:\Users\%username%\AppData\Local`).

```
print(user_local_appdata())
```

Indication : Pour plus d'informations ou pour connaître les paramètres complémentaires de la commande, consultez la documentation de référence (en anglais) en visitant :

https://www.wapt.fr/en/api-doc-1.5/source/setuphelpers.html?highlight=user_local_appdata#setuphelpers.user_local_appdata

45.11.5 Désactiver temporairement le redirecteur wow3264

La commande **disable_file_system_redirection** …

```
with disable_file_system_redirection():
    filecopyto('file.txt',system32())
```

Indication : Pour plus d'informations ou pour connaître les paramètres complémentaires de la commande, consultez la documentation de référence (en anglais) en visitant :

https://www.wapt.fr/en/api-doc-1.5/source/setuphelpers.html?highlight=disable_file_system_redirection#setuphelpers.disable_file_system_redirection

45.11.6 Récupérer l'utilisateur courant

… affiche l'identifiant de l'utilisateur connecté

```
print(get_current_user())
```

Indication : Pour plus d'informations ou pour connaître les paramètres complémentaires de la commande, consultez la documentation de référence (en anglais) en visitant :

https://www.wapt.fr/en/api-doc-1.5/source/setuphelpers.html?highlight=get_current_user#setuphelpers.get_current_user

45.11.7 Récupérer le nom de l'ordinateur

La commande **get_computername** …

```
print(get_computername())
```

Indication : Pour plus d'informations ou pour connaître les paramètres complémentaires de la commande, consultez la documentation de référence (en anglais) en visitant :

https://www.wapt.fr/en/api-doc-1.5/source/setuphelpers.html?highlight=get_computername#setuphelpers.get_computername

45.11.8 Récupérer le nom de domaine

... renvoie le nom de la machine avec le domaine.

```
get_domain_fromregistry()
```

Indication : Pour plus d'informations ou pour connaître les paramètres complémentaires de la commande, consultez la documentation de référence (en anglais) en visitant :

https://www.wapt.fr/en/api-doc-1.5/source/setuphelpers.html?highlight=get_domain_fromregistry#setuphelpers.get_domain_fromregistry

45.11.9 Les 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}]
```

Indication : Pour plus d'informations ou pour connaître les paramètres complémentaires de la commande, consultez la documentation de référence (en anglais) en visitant :

https://www.wapt.fr/en/api-doc-1.5/source/setuphelpers.html?highlight=installed_softwares#setuphelpers.installed_softwares

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
```

Indication : Pour plus d'informations ou pour connaître les paramètres complémentaires de la commande, consultez la documentation de référence (en anglais) en visitant :

https://www.wapt.fr/en/api-doc-1.5/source/setuphelpers.html?highlight=uninstall_cmd#setuphelpers.uninstall_cmd

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*.

Indication : Pour plus d'informations ou pour connaître les paramètres complémentaires de la commande, consultez la documentation de référence (en anglais) en visitant :

https://dev.tranquil.it/sphinxdocs/source/setuphelpers.html?highlight=uninstall_cmd#setuphelpers.uninstall_cmd

---

Tuer des tâches

La commande **killalltasks** tue toutes les tâches portant le nom spécifié.

```
killalltasks('firefox')
```

Indication : Pour plus d'informations ou pour connaître les paramètres complémentaires de la commande, consultez la documentation de référence (en anglais) en visitant :

https://www.wapt.fr/en/api-doc-1.5/source/setuphelpers.html?highlight=killalltasks#setuphelpers.killalltasks

---

45.12 Utiliser les champs du fichier control

Il est possible d'utiliser les informatoins du fichier control dans le `setup.py`

45.12.1 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.

45.12.2 Récupérer le nom du logiciel

À faire : documentation à venir

45.13 Gérer un paquet WAPT avec un autre paquet WAPT

45.13.1 Installer un paquet

La commande **install** ...

```
WAPT.install('tis-scratch')
```

... installe *tis-scratch* sur la machine.

45.13.2 Supprimer un paquet

La commande **remove** ...

```
WAPT.remove('tis-scratch')
```

... désinstalle *tis-scratch* de la machine.

45.13.3 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*.

45.14 Améliorer mon paquet

45.14.1 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 *

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.

45.14.2 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',
    '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')
```

(suite sur la page suivante)

(suite de la page précédente)

```
    #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**.

45.14.3 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 = []
```

(suite sur la page suivante)

(suite de la page précédente)

```
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()
```

45.14.4 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 connecté, avait déjà été exécuté.

```
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
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`**.

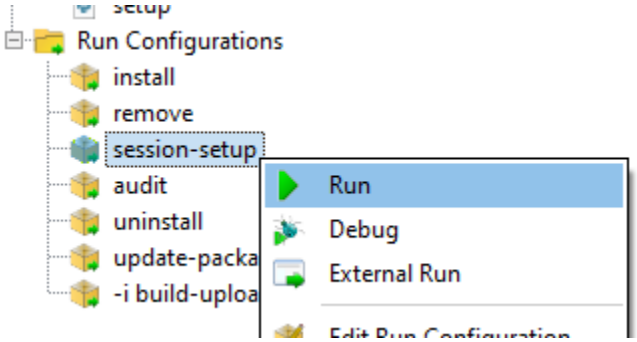


Fig. 14 – Pyscripter - Effectuer un session-setup

— Enfin, vérifier que l'icône est bien présente sur le bureau.

45.14.5 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
```

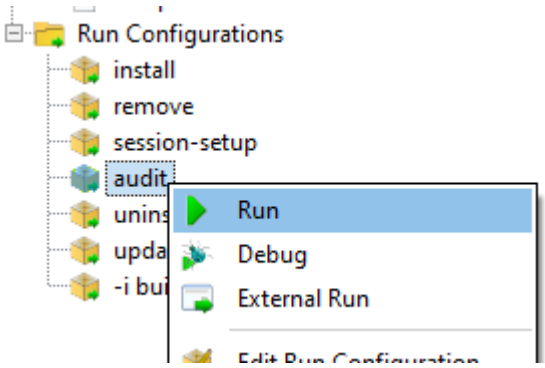
On définit la partie **audit** dans le fichier `setup.py` du paquet dans une fonction **def audit()** :

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.

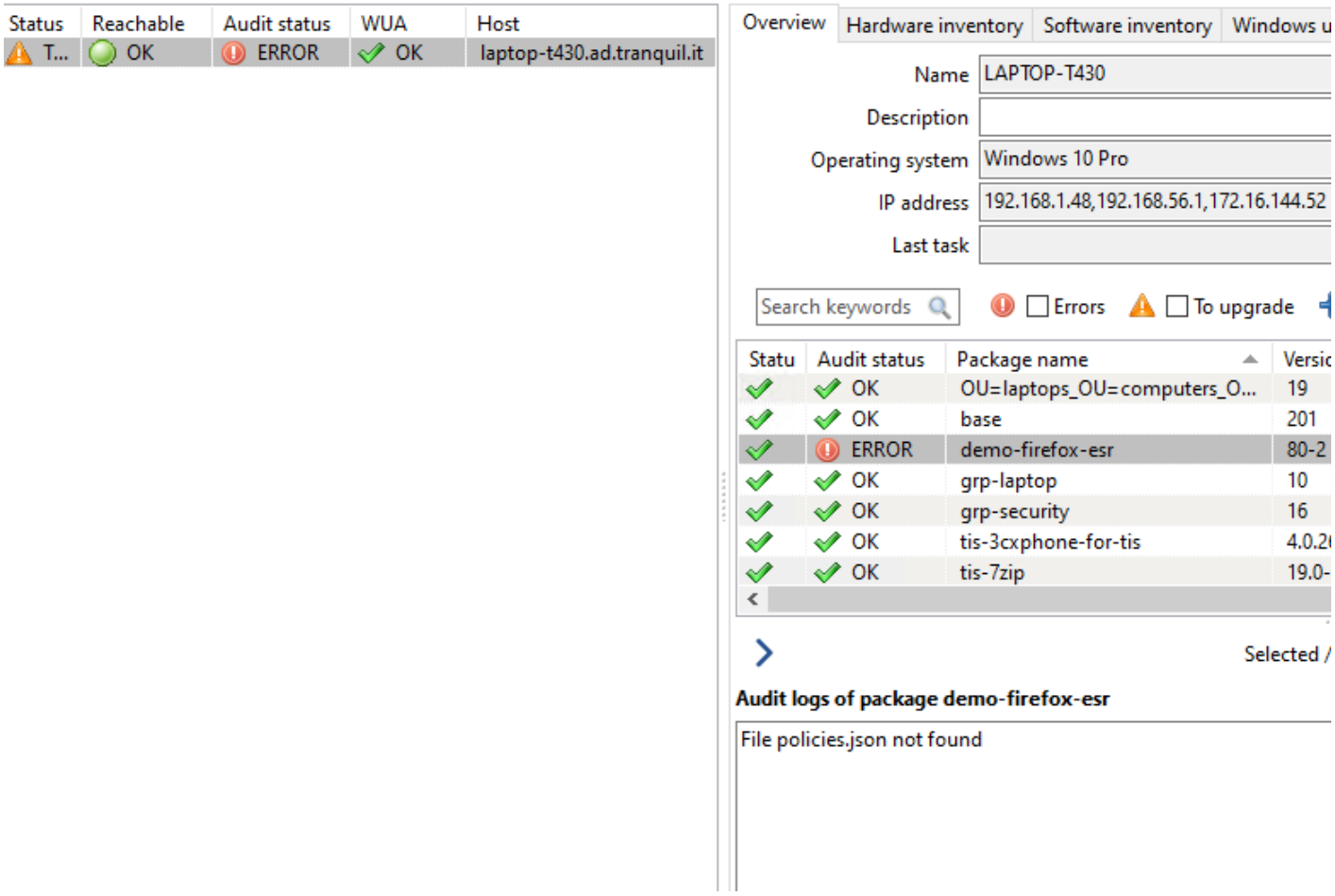


FIG. 15 – Vérifier l'état d'un audit dans la console WAPT

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 alors il faudra effectuer l'audit manuellement ou à partir de la console WAPT.

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).

Comportement par défaut de la fonction d'audit

Par défaut, 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.

45.14.6 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 du temps lorsque qu'il faut mettre à jour un paquet 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',' ')
```

(suite sur la page suivante)

(suite de la page précédente)

```
    #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** :

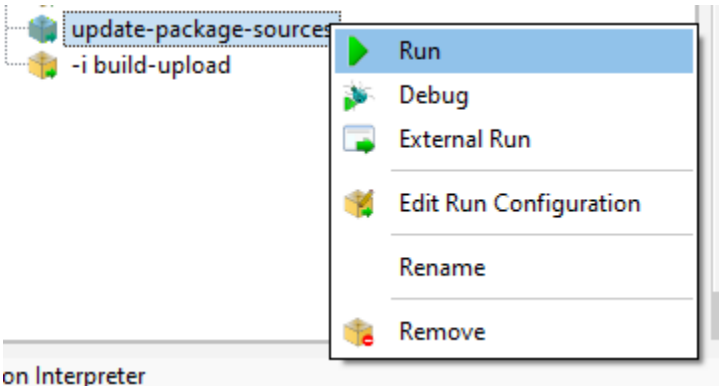


Fig. 16 – Pyscripter - Exécution d'un update-package-source

Vous trouverez de nombreux exemples d'*update_package* qui vous inspireront dans les paquets du store de Tranquil IT.

45.14.7 Exemple : déployer un logiciel portable avec WAPT

Un bon exemple de paquet applicatif WAPT est celui d'un logiciel dit *portable*. Pour cela, il faudra :

- 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()
```

45.14.8 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*.

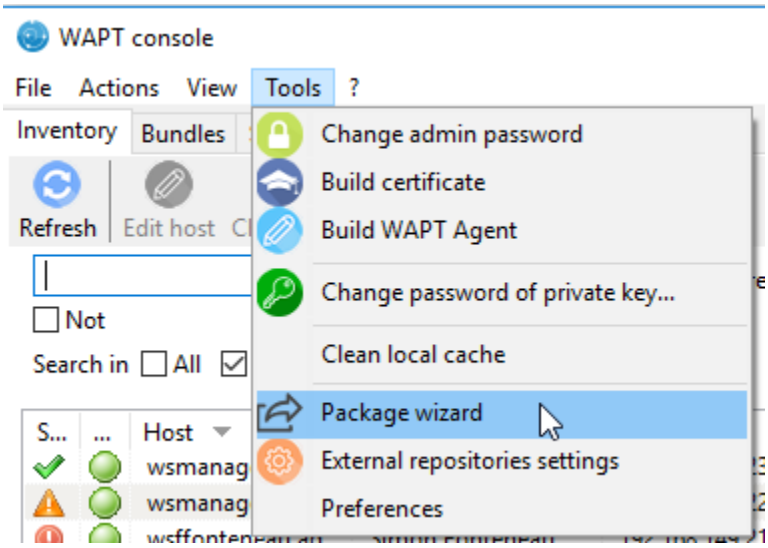


FIG. 17 – PyScripter - Menu pour la création de modèle de paquets depuis la console

— 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.

45.14.9 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 de votre dossier de développement de paquets :

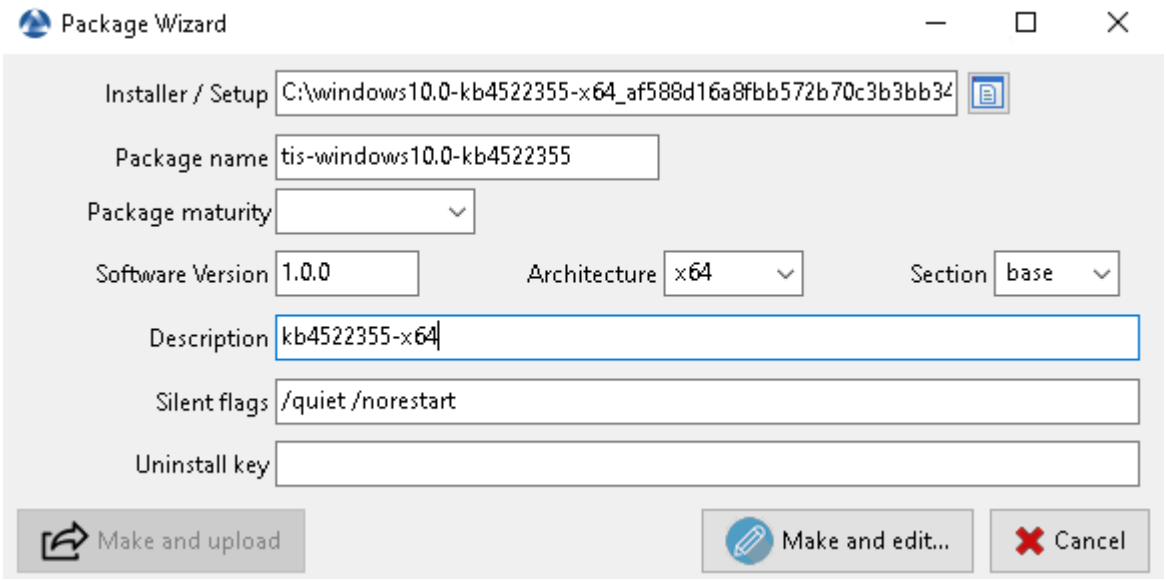


FIG. 18 – Informations requises pour la création du paquet MSU

— `settings.json`;
— `launch.json`;
Exemple avec **`TightVNC`** :

```
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.

— Vérifier le contenu du fichier `control`.
Vous devez donner une **description** à votre paquet, et renseigner le **target_os** et la **version** de votre paquet.

Indication : `os_target` pour unix est *linux*

Avertissement : Le numéro de `version` dans votre fichier `control` doit commencer à 0, et non le numéro de version du logiciel car nous ne savons pas précisément à partir de apt/yum repo quelle sera la version du logiciel.

— Fichier `control` d'origine.

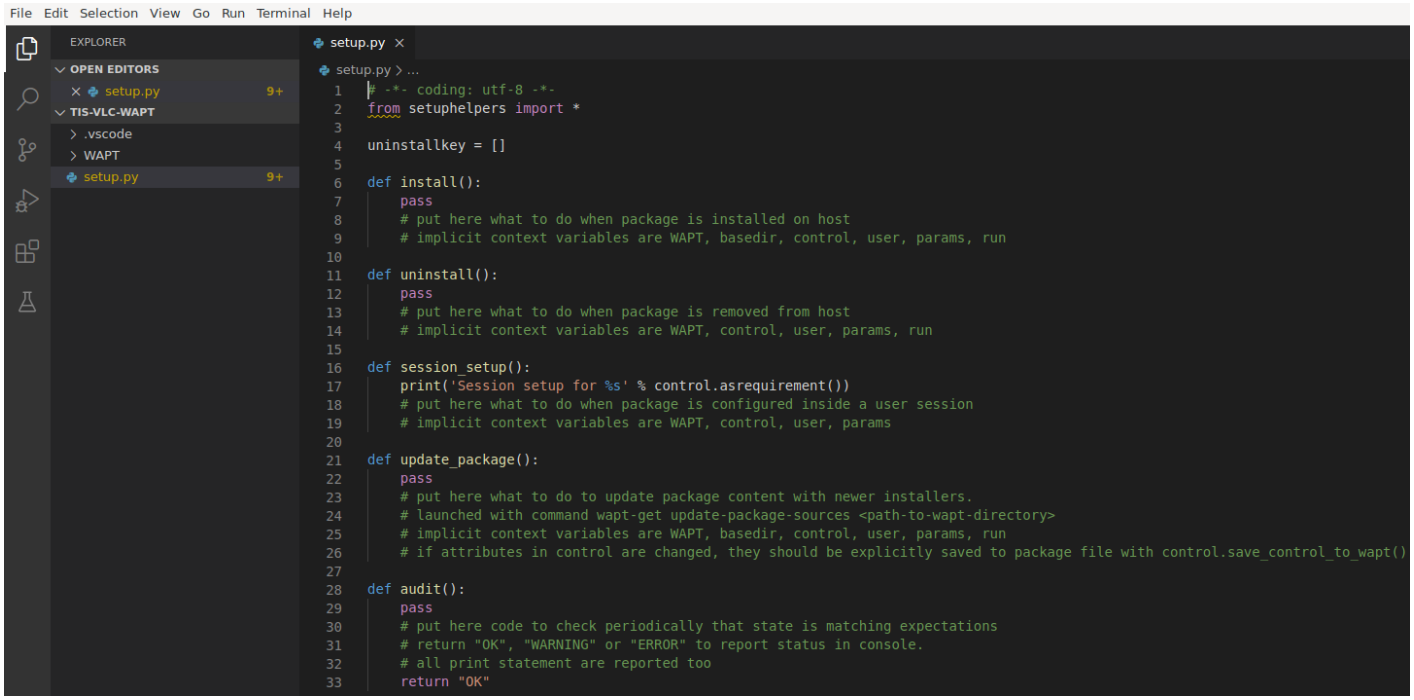


FIG. 19 – Ouverture du VSCode avec le focus sur le fichier *setup*

```
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*.

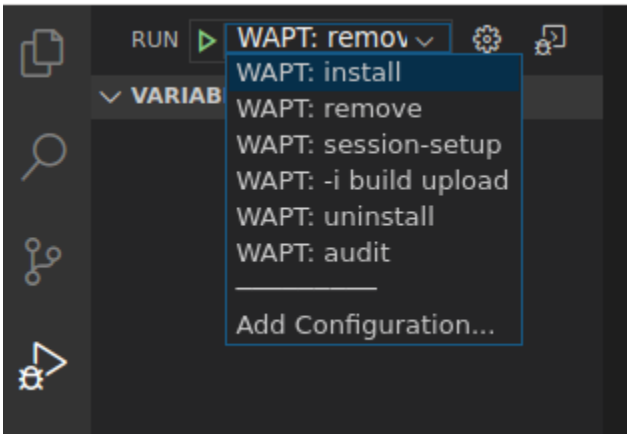


FIG. 20 – À l'issue de la désinstallation, le programme est désinstallé

— Vérifier que le logiciel a été correctement supprimé.

```
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*.

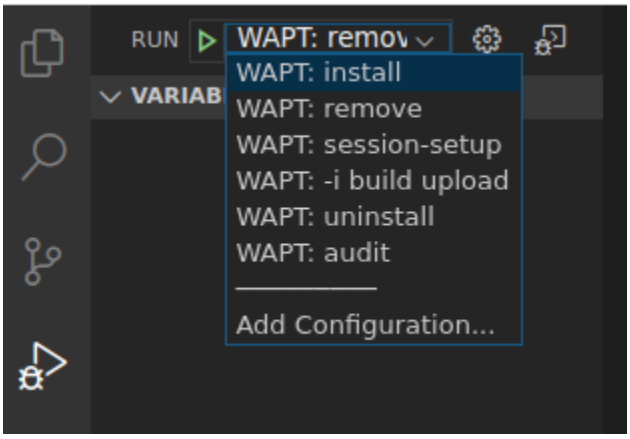


FIG. 21 – À l'issue de la désinstallation, le programme est désinstallé

Construire et téléverser le paquet

Vous trouverez votre paquet ici : `~/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*.

45.14.10 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)\waptprivate`.

- 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**.

45.15 Utiliser des IDE différents pour développer les paquet WAPT

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.

45.15.1 Sur Windows

Tableau 3 – Éditeurs de texte supportés en natif dans WAPT sous Windows

Récupérer le nom du logiciel	Logo de l'éditeur de texte
vscode;	**|pyscripteur|**
vscode;	
vscodium;	

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
```

45.15.2 Sur Linux / macOS

Tableau 4 – Éditeurs de texte supportés en natif dans WAPT sous Windows

Récupérer le nom du logiciel	Logo de l'éditeur de texte
vscode;	
vscodium;	
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
```

45.16 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}
```

45.16.1 Arguments personnalisés

Tableau 5 – 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

45.17 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.

Tableau 6 – Les principales différences de syntaxe

Syntaxe	Python 2	Python 3
print	print 'Hello'	print('Hello')
unicode string	{wapt_sourcesdir}	{wapt_sourcesdir}
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/

CHAPITRE 46

Structure d'un paquet WAPT

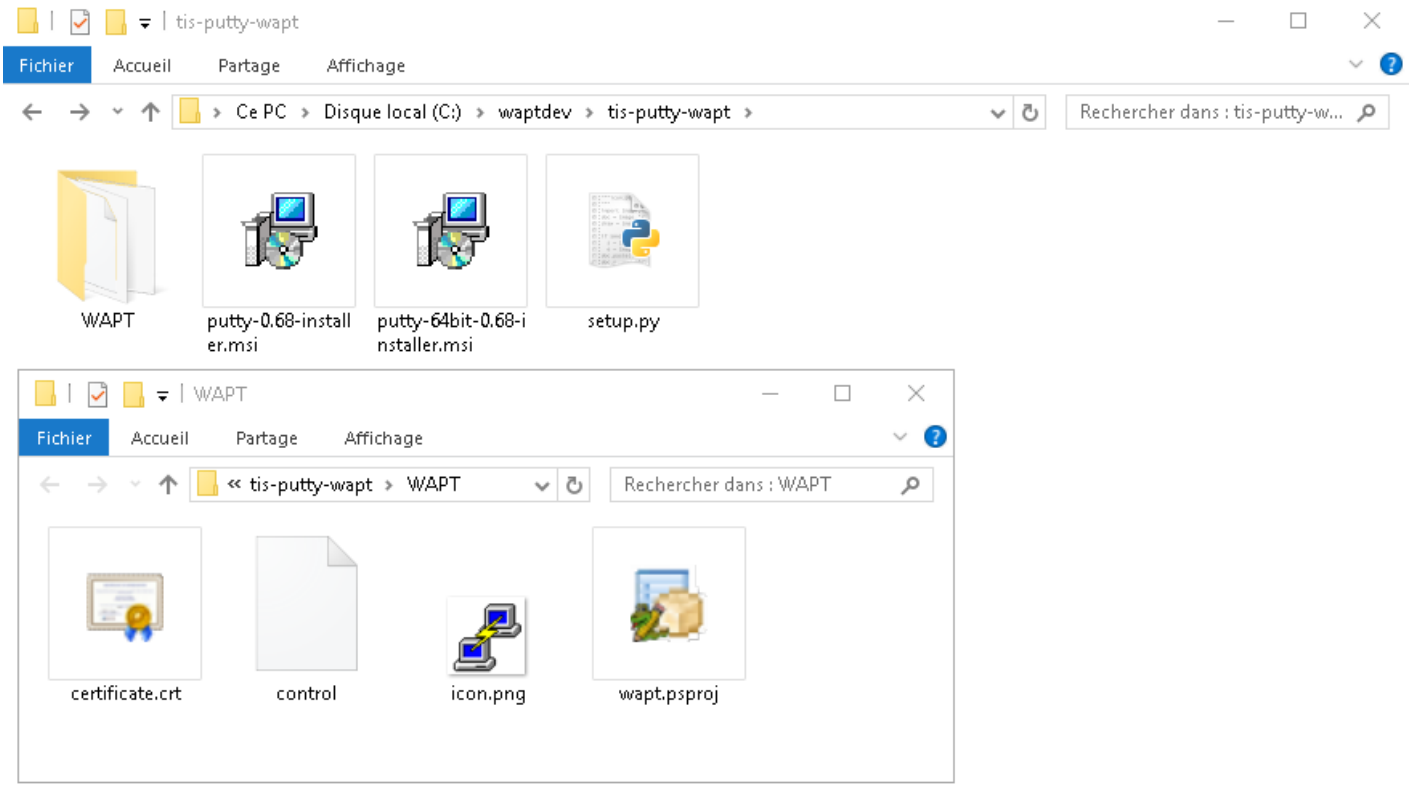


FIG. 1 – Structure du packaging WAPT affichée dans l'explorateur Windows

Un package WAPT est un fichier : mimetype : *.zip* contenant plusieurs éléments :

— un fichier `setup.py` à la racine ;

— un fichier ou plusieurs fichiers **binaire.exe** à la racine ;
— un fichier ou plusieurs autres fichiers additionnels à la racine ;
— 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 ;

46.1 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
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==
```

(suite sur la page suivante)

(suite de la page précédente)

```
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
```

Tableau 1: 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`	Version du paquet, ne peut pas contenir plus de 5 délimiteurs, le dernier chiffre étant le numéro de version de packaging 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 ; — **x86** : le paquet est destiné uniquement aux machines avec un processeur 32bits ;	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>
`description`	Description du paquet qui apparaîtra dans la console et sur l'interface web 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

suite sur la page suivante

Tableau 1 – suite de la page précédente

Paramètre	Description	Valeur Exemple
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 packaging 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 ajouter dans wapt-get.ini de l'agent WAPT la configuration *maturities*.	PROD
locale	Environment linguistique prévu pour le paquet 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.	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 1.8, le champ *target_os* peut être soit *windows*, *macos*, *linux* ou laissé vide.	windows,mac,linux
min_os_version	Version minimale de l'OS pour que le paquet soit vu par l'agent WAPT le paquet Pour un *windows target_os*, ce champ définit le minimum Windows Version 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

suite sur la page suivante

Tableau 1 – suite de la page précédente

Paramètre	Description	Valeur Exemple
`max_os_version`	Version minimale de l'OS pour que le paquet soit vu par l'agent WAPT le paquet 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`	Version minimale de WAPT pour un fonctionnement correct du paquet 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`	Lien de stockage des versions historisées du paquet (commande **`sources`**) 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`	Espace de disque dur libre minimal requis pour installer le paquet 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
`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`	Indique la périodicité pour l'exécution de la fonction audit du paquet 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

suite sur la page suivante

Tableau 1 – suite de la page précédente

Paramètre	Description	Valeur Exemple
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
licence	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	Indique une liste de mots clé correspondant au paquet 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	Identifiant unique du packaging. Il est généré automatiquement lors de la construction du packaging.	dc66ccd1-d987-482e-b792-04e89a3803f7
version	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
maintainer	Date/heure après laquelle le packaging ne peut pas être installé. L'agent refusera de l'installer après cette date. La chaîne est formatée selon la norme iso8601 : YYYY-MM-DDTHH :MM :SS	2022-02-23T00 :00 :00
installed_size	Date/heure après laquelle le waptagent déclenchera une installation forcée du packaging. La chaîne est formatée selon la norme iso8601 : YYYY-MM-DDTHH :MM :SS	2022-02-23T00 :00 :00
signer	Nom commun (CN) du signataire du paquet Il s'agit généralement du nom complet du signataire. La valeur est automatiquement insérée lors de la signature du packaging WAPT.	Tranquil IT
signer_fingerprint	Empreinte de la signature du signataire du paquet La valeur est automatiquement insérée lors de la signature du packaging WAPT.	2BA-FAF007C174A3B00F12E9CA1E74956
signature	Hash SHA256 du paquet La valeur est automatiquement insérée lors de la signature du packaging WAPT.	MLOz-Liz0qC(…)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)**.

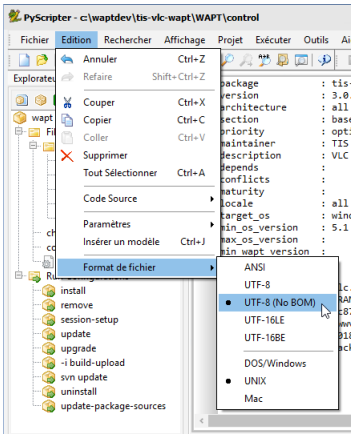


FIG. 2 – PyScripter - UTF-8 (No BOM)

46.2 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 integrée à WAPT, elle embarque des fonctions simplifiées pour aider à créer des paquets.

— 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()`
C'est la recette du paquet WAPT.

46.3 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.

46.4 Le fichier *icon.png*

On trouve dans le dossier WAPT un fichier `icon.png`.

Il permet d'associer une icône au paquet.

Cette icône apparaîtra dans l'interface web du self-service WAPT (http://127.0.0.1:8088).

Indication : L'icône devra être au format png 48px par 48px.

46.5 Le fichier *manifest.sha256*

On trouve dans le dossier WAPT un fichier `manifest.sha256`.

Il contient l'empreinte sha256 de chaque fichier du package.

46.6 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-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` ;

46.7 Autres fichiers

D'autres fichiers peuvent être embarqués dans le paquet WAPT. Par exemple :

- un installeur à côté du `setup.py` à appeler dans votre **setup.py** ;
- un fichier de réponses à passer à l'installeur du logiciel ;
- un fichier de licence ;
- etc.

CHAPITRE 47

WAPT Setuphelpers apidoc

47.1 Setuphelpers for Windows

47.1.1 WAPT 2.2.0

Link for Windows in 2.2.3

47.1.2 WAPT 2.2.1

Link for Windows in 2.2.1

47.1.3 WAPT 2.2.2

Link for Windows in 2.2.2

47.1.4 WAPT 2.2.3

Link for Windows in 2.2.3

47.2 Setuphelpers for Linux

47.2.1 WAPT 2.2.0

Link for Linux in 2.2.0

47.2.2 WAPT 2.2.1

Link for Linux in 2.2.1

47.2.3 WAPT 2.2.2

Link for Linux in 2.2.2

47.2.4 WAPT 2.2.3

Link for Linux in 2.2.3

47.3 Setuphelpers for MacOS

47.3.1 WAPT 2.2.0

Link for MacOS in 2.2.0

47.3.2 WAPT 2.2.1

Link for MacOS in 2.2.1

47.3.3 WAPT 2.2.2

Link for MacOS in 2.2.2

47.3.4 WAPT 2.2.3

Link for MacOS in 2.2.3

CHAPITRE 48

Problèmes fréquents

48.1 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..

48.2 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.

48.2.1 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.

48.2.2 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*.

48.3 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).

48.4 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.

48.4.1 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
```

48.5 Mon WAPTdeploy ne fonctionne pas

L'utilitaire **waptdeploy** n'arrive pas à installer l'agent WAPT.

48.5.1 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.

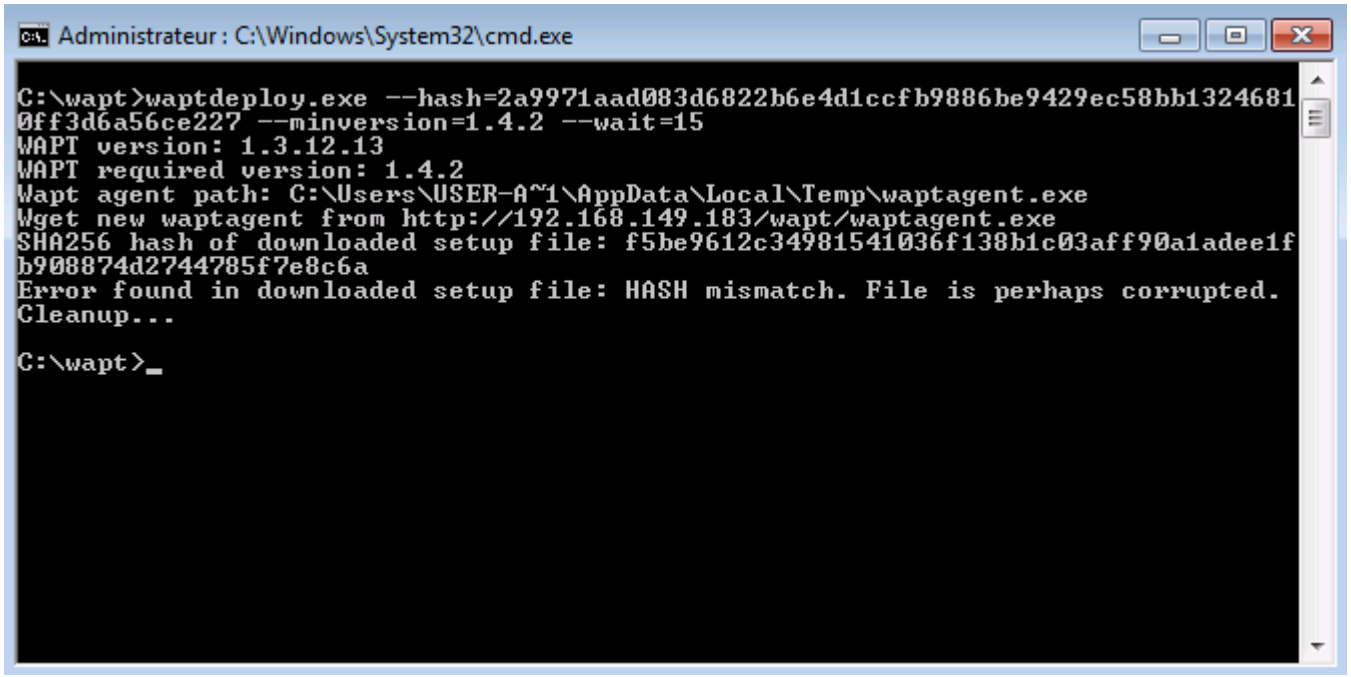


Fig. 1 – Erreur avec le hachage de l'utilitaire WAPT Deploy dans une fenêtre de terminal texte

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`.

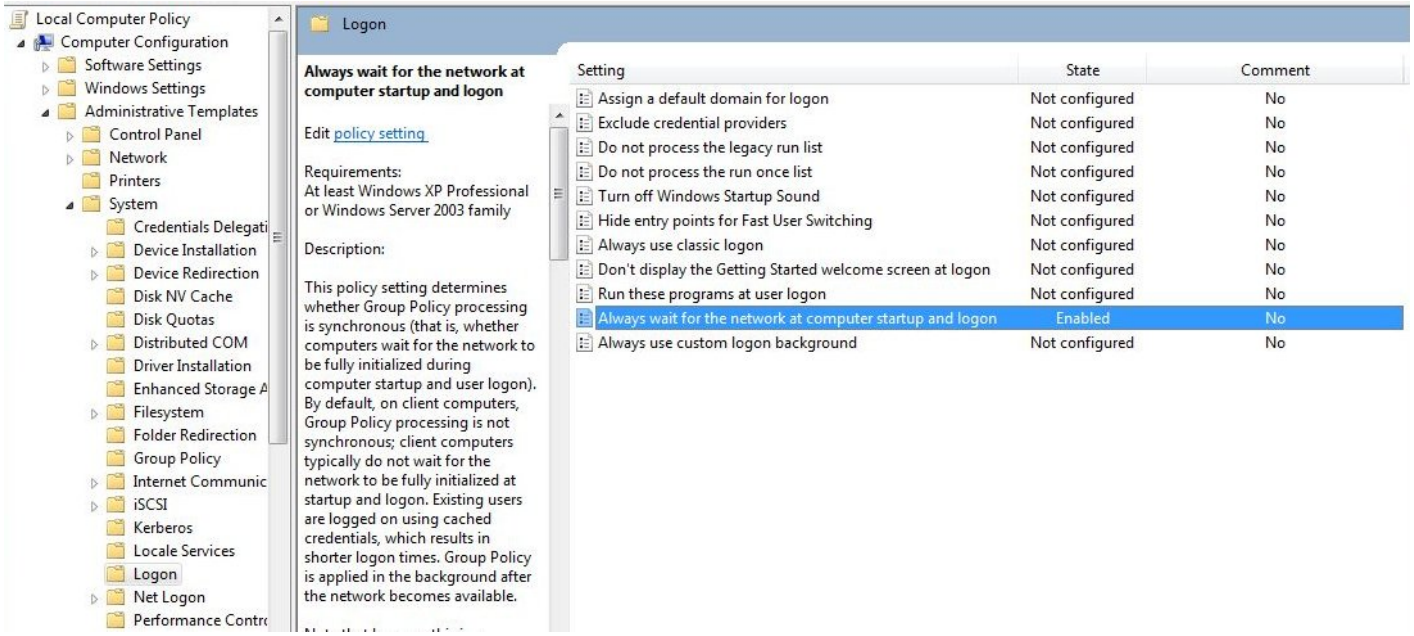
48.6 Windows n'attend pas le réseau au démarrage

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*



48.7 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.

48.7.1 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.

48.7.2 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`.

48.7.3 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.

48.8 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`

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`.

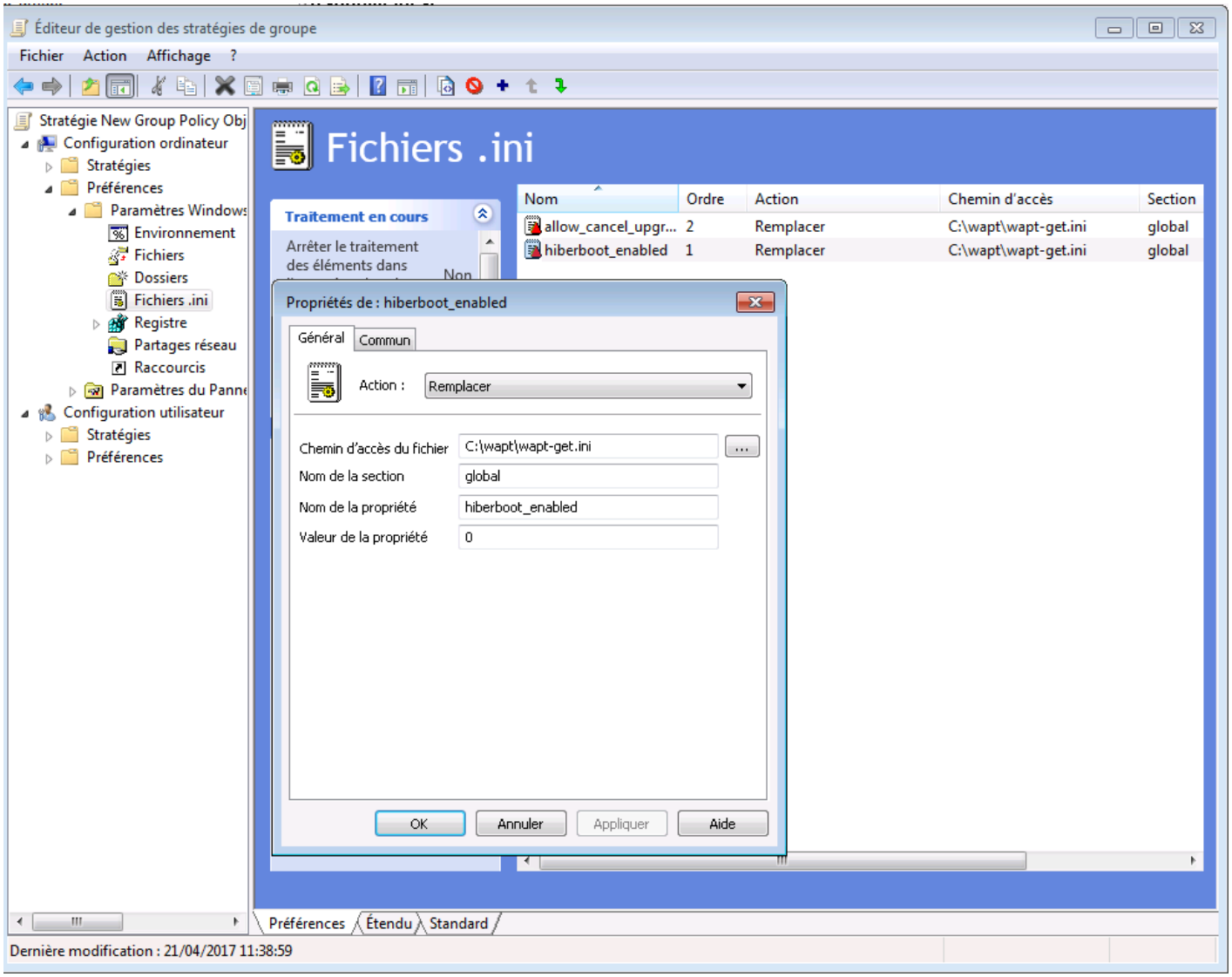
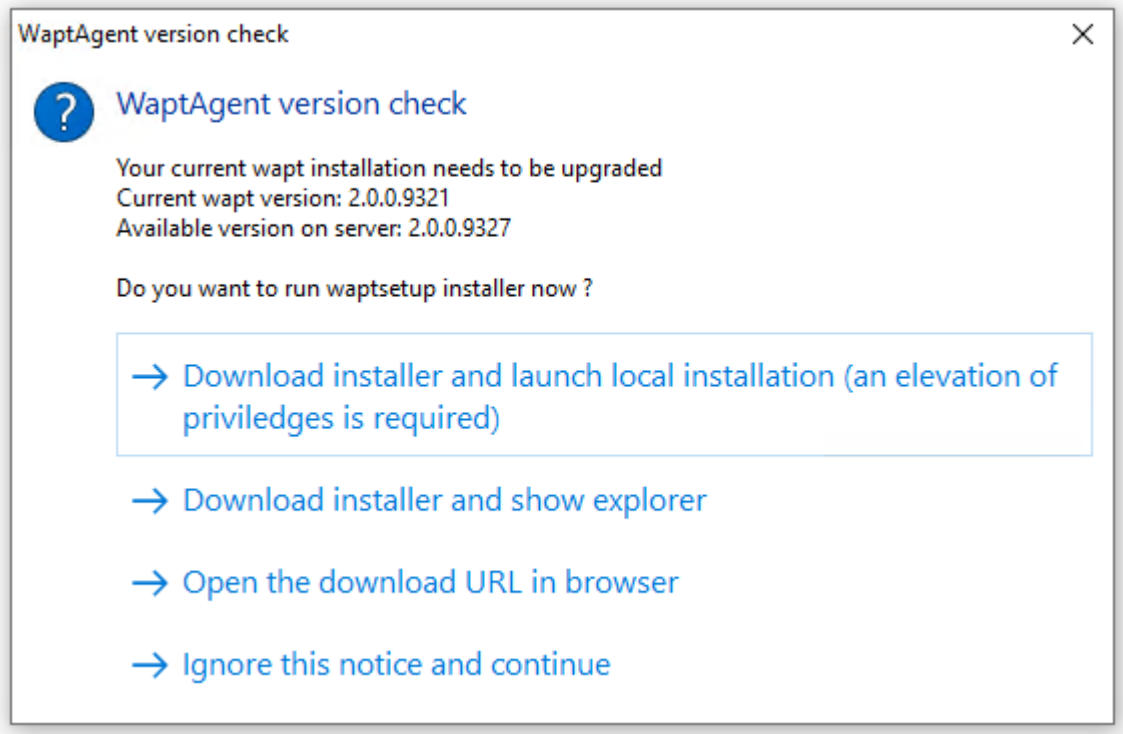


FIG. 2 – Utilisation d'un fichier GPO ini pour configurer le délai d'exécution du script

48.9 Message d'erreur à l'ouverture de la console

48.9.1 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.

48.9.2 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.

48.9.3 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
```

48.9.4 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.

48.10 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.

48.11 Problème lors du enable-check-certificate

48.11.1 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/`.

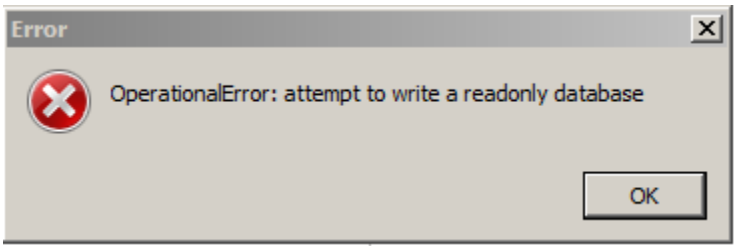
   Ensuite, relancez le script de postconf (le même que pendant l'installation initiale, avec les mêmes arguments et les mêmes valeurs).
   Enfin, vérifiez bien le nom renseigné lors de l'étape *FQDN for the WAPT serveur* est correcte.

— Vous pouvez maintenant retenter votre **`enable-check-certificate`**.

48.12 Problème avec la création de paquet

48.12.1 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.

48.12.2 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*.

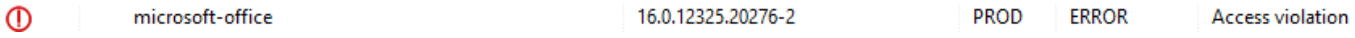
Indication : Le packaging WAPT dans `C:\waptdev`.

— 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
```

48.12.3 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*.

48.12.4 Paquet WAPT en erreur

Problème d'installation

J'ai un paquet en erreur et le logiciel n'est pas installé sur la machine quand je me déplace.

Explication

Une erreur est survenue pendant l'exécution de l'installation définie dans `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 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 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.

Vérifier avec la commande :

```
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 !**

48.13 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"
```

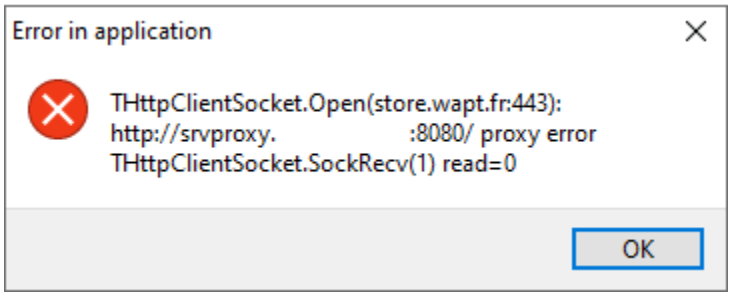
(suite sur la page suivante)

(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"
```

48.14 Je rencontre un problème avec mon proxy - THttpClientSocket.SockRecv(1) read = 0

Si vous avez ce problème :



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`.

CHAPITRE 49

Erreurs courantes

49.1 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`.

49.1.1 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:`.

49.2 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**.

49.3 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**.

49.4 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)*.

49.5 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.

CHAPITRE 50

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.

50.1 Pré-requis

— hôtes Debian Linux ou CentOS ;
— un sudoer account sur ces machines ;
— Ansible 2.8.

50.2 Installer le rôle Ansible

50.2.1 Découverte

50.2.2 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/
```

50.3 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 }
```

— Lancez 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* ! !

50.3.1 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
```

50.3.2 Exemple d'un *playbook* Ansible

Voici un exemple d'un *playbook* Ansible.

```
- hosts: srvwapt
  vars_files:
    - vars/main.yml
  roles:
    - tranquilit.waptserver
```

50.4 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.

50.4.1 Pré-requis

— hôtes Debian Linux ou CentOS ;
— un sudoer account sur ces machines ;
— Ansible 2.8 ;

50.4.2 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/
```

50.4.3 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 !

50.4.4 Paramètres des rôles Ansible

Les variables disponibles sont énumérées ci-dessous, ainsi que les valeurs par défaut (voir `defaults/main.yml`).

50.4.5 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"
```

50.4.6 Exemple d'un *playbook* Ansible

Voici un exemple d'un *playbook* Ansible.

```
- hosts: hosts
  vars_files:
    - vars/main.yml
  roles:
    - tranquilit.waptagent
```

CHAPITRE 51

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.

51.1 API V1

51.1.1 /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
```

51.1.2 /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
```

51.1.3 /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
```

51.2 API V2

51.2.1 /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
```

51.3 API V3

51.3.1 /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
```

51.3.2 /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
```

51.3.3 /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))
```

51.3.4 /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
```

51.3.5 /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
```

51.3.6 /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
```

51.3.7 /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.

51.3.8 /api/v3/upload_packages

À faire : Tests

51.3.9 /api/v3/upload_hosts

À faire : Tests

51.3.10 /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"
```

51.3.11 /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

51.3.12 /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"
```

51.3.13 /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}
```

51.3.14 /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}
```

51.3.15 /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}
```

51.3.16 /api/v3/trigger_host_action

À faire : Tests

51.3.17 /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"
```

51.3.18 /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))
```

CHAPITRE 52

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

CHAPITRE 53

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.

CHAPITRE 54

Présentation des principes 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.

54.1 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.

54.2 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.

54.3 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 **Community**).
- **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.
- **Administrateur Local**
  Utilisateur disposant des droits d'administration locaux sur les postes équipés de WAPT (**Enterprise** et **Community**) ;

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.

54.4 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) :
- Aisponibilité ;
- Intégrité ;
- Confidentialité ;
- Authenticité.

Les biens sensibles à protéger sont les suivants :

54.4.1 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.

54.4.2 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.

54.4.3 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.

54.4.4 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.

54.4.5 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.

54.4.6 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.

54.5 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 environment.

Les hypothèses suivantes sur l'environment d'exploitation de WAPT doivent être considérées :

54.5.1 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.

54.5.2 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.

54.5.3 Hypothèse H3 : Les binaires nécessaires au fonctionnement de WAPT sont intègres

Toutes les librairies et outils nécessaires au fonctionnement de WAPT sont considérées saines. A la réception d'une requête par l'agent WAPT, il vérifie que la requête est correctement signée.

54.5.4 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.

54.5.5 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.

54.5.6 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.

54.6 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 :

54.6.1 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.

54.6.2 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.

54.6.3 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.

54.6.4 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.

54.6.5 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.

54.7 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.

54.7.1 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 : risques traités
- — 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 ;

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 : risques traités (notamment MITM) :
- 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 : risques traités
- 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 : risques traités
- 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 : risques traités
- 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.

54.7.2 Fonction de sécurité F2 : Protection de l'intégrité du processus d'installation des paquets WAPT

Fonction de sécurité F2A : Signature des paquets WAPT

Note : risques traités
— 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 : risques traités
— 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 : risques traités
- 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.

Fonction de sécurité F2D : Restriction totale d'accès au répertoire de stockage du couple clé privé / certificat de signature d'inventaire

---

Note : risques traités
- 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.

54.7.3 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 : risques traités
— Une entité non-autorisée envoie des requêtes falsifiées aux agents WAPT.

Solution mise en place

Les commandes ci-dessous sont signées par le serveur WAPT avant d'être envoyées au travers de la Websocket à l'agent WAPT destinataire de la commande :

- `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.
- `wapt-get update-status` : ordonner à l'agent WAPT de renvoyer l'état de son inventaire actuel au serveur WAPT.
- `wapt-get upgrade` : ordonner à l'agent WAPT d'exécuter les paquets marqués **NEED UPGRADE**.
- `wapt-get update` (mettre à jour la liste des paquets disponibles).

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.

54.8 Présentation des processus cryptographiques

Date	janv. 09, 2024
Rédacteur	Hubert TOUVET
Applicable pour WAPT	>= 1.5.0.17
Version du document	1.5.0.17-0

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 client WAPT.
- Signature des inventaires et **statut des clients WAPT**.
- Authentification de la connexion Websockets du client WAPT sur le serveur.
- 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.

54.8.1 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 des paquets et actions.
- `<WAPT>sslserver` : répertoire par défaut pour stocker les certificats https du serveur (pinning).
- `<WAPT>\private` : répertoire par défaut pour les certificats permettant de signer l'inventaire et les connexions Websocket.
- `%LOCALAPPDATA%waptconsolewaptconsole.ini` : fichier de configuration de la console et des actions de développement de l'outil **`wapt-get`**.
- `%appdata%waptconsolessl` : répertoire par défaut pour les certificats de confiance pour l'import de paquets depuis un dépot externe (c.à.d. les *modèles de paquets*).

54.8.2 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.
— **Postes clients 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* ;

54.8.3 Synthèse des modules crypto mis en oeuvre par la solution WAPT

Coté client WAPT (WAPT 1.5.0.12) :
— module *ssl* standard de **Python 2.7.13** linké sur **`OpenSSL 1.0.2j 26 Sep 2016`** pour les connexions https entre les clients WAPT et serveur WAPT.
— **cryptography==1.9** linké sur **`openssl 1.1.0f`** pour toutes les opérations crypto RSA, génération de clés, de certificat X509, de signature et vérification.
— **kerberos-sspi==0.2** et **requests-kerberos==0.11.0** pour l'authentification du client WAPT lors de son enregistrement initial sur le serveur.
— **pyOpenSSL==17.0.0** : pour récupérer la chaîne de certificats du serveur WAPT.
— **certifi==2017.4.17** : base pour les certificats d'Autorité Racine.
— **dll Openssl 1.0.2l** pour la partie waptcommon.pas écrite avec la bibliothèque FPC Indy et la classe TIdSSLIOHandlerSocketOpenSSL.
Coté serveur WAPT :
— **nginx/1.10.2** : configurée pour TLS1.2, chiffre "EECDH+AESGCM :EDH+AESGCM :AES256+EECDH :AES256+EDH".
— module *ssl* standard de **python 2.7.5** linké sur **`OpenSSL 1.0.1e-fips 11 Feb 2013`**.
— **cryptography==1.9** linké sur **`OpenSSL 1.0.1e-fips 11 Feb 2013`** pour toutes les opérations crypto RSA, X509, signature et vérification.

54.8.4 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` .

Validité du certificat de l'Administrateur

Jusqu'à la version 1.5.0.12 incluse, Le client WAPT ne gère pas la vérification de la révocation du certificat de l'*Administrateur* lors du processus de vérifcation des paquets, attributs et actions de l'*Administrateur*.

Il ne vérifie que les dates de validité (attributs *notValidBefore* / *notValidAfter*). Le certificat est valide si (**Now** >= *notValidBefore* et **Now** <= *notValidAfter*).

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%waptconsolewaptconsole.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*.

54.8.5 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 :
- 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**.

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 : tette 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\<fqdn du poste>.pem.
— Le certificat généré a les attributs suivants :
  — Sujet.COMMON_NAME = <device FQDN>.
  — *Subject.ORGANIZATIONAL_UNIT_NAME* = nom de l'*Organisation* enregistré la base de registre du client Windows.
  — *SubjectAlternativeName.DNSName* = <device FQDN>.
  — *BasicConstraint.CA* = True.
  — *Validity* = 10 ans.
  — *Serialnumber* = aléatoire ;
— Le certificat est sauvegardé dans le fichier <WAPT>private<device FQDN>.crt.

Note : Seuls 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 status 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* ou UUID du client WAPT.

— Après avoir initialement authentifié le client WAPT avec Kerberos, le serveur reçoit le certificat envoyé par le client et il le stocke dans son inventaire, dans la table *hosts* (champ *host_certificate* en format PEM).

Note : Si le poste client WAPT est renommé, la paire de clés 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 authentification kerberos ; ainsi le nouveau certificat sera enregistré dans la base de données.

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 d'autorité**.

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.

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`
  cette méthode ne fonctionnera bien que si le serveur https est configuré pour renvoyer son certificat et les certificats intermédiaires à l'initialisation de communication TLS.
- *verify_cert* = <chemin vers fichier .pem>
  vérifie le certificat du serveur https 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`
  ne pas vérifier le certificat du serveur https ;

Conventionnellement, on stocke le bundle de l'*Autorité de Certification* approuvées dans le répertoire <WAPT>`sslserver`.

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>`sslserver<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>`sslserver` et le paramètre *verify_cert* de la section `[global]` du fichier <WAPT>`wapt-get.ini` est renseigné pour désigner le bundle.

54.8.6 Communications HTTPS entre les clients WAPT et les dépôts WAPT

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**.

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.

54.8.7 Communications entre la console WAPT et le serveur WAPT

Déployer des certificats d'autorité

Paramètre *verify_cert* de la section `[global]` du fichier `%LOCALAPPDATA%waptconsolewaptconsole.ini` :

- `verify_cert = True`
  cette méthode ne fonctionnera bien que si le serveur https est configuré pour renvoyer son certificat et les certificats intermédiaires à l'initialisation de communication TLS.
- *verify_cert* = <chemin vers fichier .pem>
  vérifie le certificat du serveur https 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`
  ne pas vérifier le certificat du serveur https ;

Conventionnellement, on stocke le bundle de l'"*Autorité de Certification* approuvées dans le répertoire `<WAPT>sslserver`.

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>sslserver<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 **wapt-get enable-check-certificate**.

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%waptconsolewaptconsole.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%waptconsolewaptconsole.ini` sont réputés de confiance.

Si le paramètre n'est pas mentionné explicitement, il est initialisé à `%appdata%waptconsolessl`.

Ce répertoire n'est pas automatiquement rempli par WAPT. Il est de la responsabilité de l'*Administrateur* de copier/coller les fichiers `PEM` d'autres *Administrateurs* ou les certificats des Autorités de Certification.

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 l'attribut *issuer_subject_hash* du certificat. La signature du certificat est effectuée par la classe **`x509.verification.CertificateVerificationContext`** ;

54.8.8 Processus de signature d'un paquet

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>`.
- commande shell `wapt-signpackage.py <WAPT-package-list>`.
- sauvegarde d'un paquet *host* dans la console WAPT.
- sauvegarde d'un paquet *group* dans la console WAPT.
- import direct d'un paquet depuis un dépôt externe.
- wizard de création de paquets à partir de MSI et de setup.

Paramètres initiaux

- Fichier ZIP du paquet ;
- 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)* ;

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_certificate* sont ajoutés à la structure du fichier `control` :
  - *signed_attributes* : liste des noms d'attributs avec séparateur virgule (,) ;
  - *signer* : commonName de l'objet du certificat du signataire ;
  - *signature_date* : date et heure en cours (UTC) sous la forme "%Y-%m-%dT%H :%M :%S" ;
  - *signer_fingerprint* : empreinte `sha256` du certificat encodée en hexadécimal obtenue par la fonction **`fingerprint`** de la classe **`cryptography.x509.Certificate`**.
- Les attributs de la structure control sont encodés en JSON.
- 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`.

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 ZIP du paquet.
- Le certificat X509 du signataire depuis le fichier est stocké dans le fichier <WAPT>`certificate.crt` du paquet WAPT.
- Les empreintes *sha256* de tous les fichiers contenus dans le paquet WAPT sont codées en hexadécimal et stockées sous forme de liste JSON [(nom de fichier, hachage),] dans le fichier <WAPT>`manifest.sha256` dans le 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.

54.8.9 Vérifier la signature des attributs d'un paquet

La vérification a lieu :

- Lors de la mise à jour de l'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`.

54.8.10 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.

54.8.11 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**.

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* : commonName de l'objet du certificat du signataire.
  - *signature_date* : date et heure en cours (UTC) sous la forme "%Y-%m-%dT%H :%M :%S".
  - *signer_certificate* : certificat *X509* du signataire encodé en base64.
- La structure est encodée en JSON.
- La signature du JSON est calculée à partir de la clé privée RSA du signataire 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*.

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.

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`.

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*.

54.8.12 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.

CHAPITRE 55

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.

55.1 Variables d'environnement

— Les variables d'environnement existent depuis le DOS. Elles peuvent vous faciliter la vie (et la mienne).

55.2 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.

55.3 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.

55.4 Codes d'erreur

— Utilisez les codes d'erreur standard lorsque vous quittez votre programme.

55.5 Impression

— Utilisez l'"API d'impression Windows et n'utilisez pas l'impression directe dans votre programme.

55.6 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**.

55.7 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.

55.8 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** !

55.9 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**.

55.10 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*.

55.11 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 !

55.12 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.

CHAPITRE 56

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.

56.1 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.

CHAPITRE 57

Bulletin de sécurité

57.1 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.

CHAPITRE 58

Changelog

58.1 WAPT-2.2 Serie

58.1.1 WAPT-2.2.3.12485 (2022-12-16)

hash : 1724df7f

This is a bugfixe release.

- [FIX] Fixed fresh install issue on WAPTServer Windows installer
- [FIX] Fixes waptexit freeze when in Discovery Edition (no licence registered) and the server is not accessible during WAPT Agent shutdown

58.1.2 WAPT-2.2.3.12481 (2022-11-30)

hash : ad3855c9

This is a security release with a few related bugfixes. All Wapt 2.0 version below 2.2.3.12481 are affected.

Note : if you are using WAPTAgent deployment though GPO, don't forget to update your waptdeploy binary in the definition GPO.

WAPT Core

- [SEC] upgrade python from 3.8.13 to 3.8.15
- [SEC] upgrade openssl from 1.1.1k to 1.1.1s
- [SEC] upgrade agent kerberos lib from 1.19.3 to 1.20.1 (linux/mac)
- [SEC] upgrade 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 install dir.**
don't run wapt-get to check working condition.
— [FIX] Add fallback method to get domain in get_hostname
— [FIX] windows, replace « wapt-get.exe » –hide by « waptpythonw.exe wapt-get.py » to run session-setup because –hide does not actually hide shell window
— [FIX] wakeonlan relays
— [REF] code cleanup for agent common.py. removed unused imports
— [FIX] waptexit : fix only_priorities argument when starting waptexit from service.
— [IMP] MacOS : update build script to handle binary file signing and better debugging

WAPT Console

— [UPD] wads : include hostname in template ipxe debian linux
— [IMP] waptconsole : don't display empty confirmation messagebox

WAPT Server

— [FIX] server postconf : force path when running psql command in postconf (linux)

58.1.3 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, 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 : Fix 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 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
— [UPD] waptserver db : change primary of HostPackagesStatus, HostExtData, Packages, HostSoftwares, HostGroups, HostWebsocket, HostAuditData, ReportingSnapshots, HostWsus, LogsAPI to bigint
— [FIX] postconf nginx : bad error string format

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
— [FIX] fix Access Violation at startup when no server is defined in inifile
— [FIX] waptconsole : when deleting package in private repo page, package is still listed until console is restarted but actually deleted on server.
— [FIX] waptconsole : random timeout error when running commands from waptconsole

WAPT Agent

— [FIX] setuphelpers. reintroduce running_as_system for linux and mac (uid==0)
— [FIX] start waptservice only if wapt-get.ini config is exists
— [FIX] add PYTHONNOUSERSITE=1 to all .sh scripts to avoid spoiling PYTHONPATH with locally installed lib in user home directory
— [FIX] 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 service can be started without any prior configuration.
— [FIX] WAPT Agent for MacOS : use system_profiler_info for dmi_info on macosx for support for Apple m1 architecture
— [FIX] WAPT Agent for MacOS : plistlib.readPlistFromBytes deprecation fix
— [FIX] WAPT Agent for MacOS : core macos : use uuid from system_profiler_info instead of dmidecode
— [FIX] WAPT Agent for MacOS : change postinst script for launchctl compatibility
— [FIX] WAPT Agent for MacOS : macos core get_hostname return binary string instead of str -> update_status loop
— [IMP] WAPT Agent for MacOS : rationalize pkg filename

58.1.4 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] Fix icacls for migrate_pg_db

WAPT Agent

— [FIX] start waptservice only if wapt-get.ini config is exists
— [FIX] add PYTHONNOUSERSITE=1 to all .sh scripts to avoid spoiling PYTHONPATH with locally installed lib in user home directory
— [FIX] remove_file() was unable to remove symlinks
— [FIX] waptconsole : fix AV at startup when no server is defined in inifile

WAPT Agent for MacOS

— [FIX] use system_profiler_info for dmi_info on macosx for support for Apple m1 architecture
— [FIX] plistlib.readPlistFromBytes deprecation fix
— [FIX] 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

58.1.5 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)

58.1.6 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 package from wapt store will use this format to make setup.py more readable. Older wapt version are not impacted for package import and package install, but may be impacted if one want to update directly from waptconsole 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] check that downloaded waptsetup version is same or newer than server
— [NEW] download from wapt.tranquil.it and upload on local waptserver agents for Linux and macOS directly from the console
— [NEW] Add a popumenu copy to clipboard as json for audit data.
— [NEW] displays 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] add licence validity check tolerance +1 day
— [FIX] trigger downloads when triggering updates from console
— [FIX] allow ~ in package names (for spaces in Org units packages)
— [UPD] icons on windows update status for WUA
— [NEW] new option check_package_version in waptconsole.ini
— [FIX] Fix saving empty value in Editor for packages
— [UPD] waptconsole reporting : add 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. #4568
— [UPD] Delete host popup
— [NEW] add feature to download packages when asking hosts for update
— [UPD] trigger_host_update adding possibility to download the package after update
— [FIX] Saving language param
— [UPD] add a NEED-SCAN waptwua.status, updated when Wapt.update() is called.
— [FIX] fix layout on Windows Update part
— [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. Popup menu
— waptconsole : Html auditdataview template filename (wapttemplates) calculated from section and key, or section
— [FIX] waptconsole drag/drop audit values
— [IMP] waptconsole : Load AD 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] add multiserver endpoint for multiserver 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] 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. (must this is not yet mandatory)
— [IMP] rename waptsetup-tis.exe to waptsetup.exe on server
— [IMP] include waptsetup.exe in waptserversetup.exe on windows
— [IMP] Download from TIS / upload to wapt server of agent installation packages

— [UPD] create a full version 1.2.3.rev-hash into file wapt/version-full
— [IMP] add htst header to nginx template
— [DEL] Remove 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] improve websockets performance and reliability. Now websocket ids are stored in memory instead being written in the database

WAPT Agent

— [FIX] fix threading exception in WAPTExit and WAPTTray that could prevent status updates
— [NEW] WAPTWUA superseded support. option include_potentially_superseded_updates in config wizard
— [NEW] Add snap software inventory
— [FIX] waptmessage unable to load sqlite on Linux and macOS
— [FIX] custom waptmessage logo linux
— [FIX] waptservice configuration : set the configs_dir relative to wapt-get.ini full path.
— [FIX] waptservice “start_waptexit” with arguments
— [FIX] bad arguments sent to waptservice triggering upgrades with “only_priorities” and “only_if_not_process_running”
— [FIX] Wapt.write_audit_data_if_changed : write data if previous data has expired.
— [IMP] wapt-get add-config-from-url : provide a meaningful message when hash is not provided
— [FIX] update template of dynamic json config packages to match new location and naming of json config related functions.
— [IMP] improve dynamic configuration handling for agent
— [FIX] waptservice : ensure a random secret_key for local waptservice session
— [FIX] wapt-get update-package-sources : handle properly relative path to package sources.
— [IMP] wapt-get edit now open changelog.txt, VSCod* now open control file too
— [UPD] change default log path to wapt/log if writable.
— [IMP] waptservice waptself : localauth with file token (ie. nopassword). Handle local groups
— [NEW] use –not-interactive with register if install run in silent mode en 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 : fix registration failing in some cases
— [IMP] macOS : add mpkg support
— [FIX] 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] patch with_md5sum in make_package_filename
— [IMP] add options for update-package-sources
— [UPD] wapt core : use datetime in UTC for audit_data
— [NEW] wapt core : allow usage of an envirnment 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] upgrade 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] waptcrypto : cert filename attribute not set when loading a cert chain
— [FIX] new option copytree2 replace_at_next_reboot
— [FIX] Avoid errors on get_version_from_binary() getting params
— [FIX] fix keyword and name with installed_softwares in macos and linux

58.1.7 WAPT-2.2.1.11957 (2022-06-02)

WAPT Deployment Server (WADS)

— [FIX] fix wapttftpserver restart on linux
— [IMP] added xml for windows 11
— [FIX] if verify_cert empty so verify_cert=0

WAPT Console

— [FIX] CheckLicence => licence is now valid one day before the real beginning

WAPT Agents

— [FIX] fix harakiri on linux

58.1.8 WAPT-2.2.1.11949 (2022-05-18)

hash : 1b2dfbee

This is a bugfix release

WAPT Deployment Server (WADS)

— [FIX] waptconsole : use ROOT in addition to CA windows system certificates stores when building winpe with verify_cert=1
— [FIX] fix selinux rules for WADS
— [FIX] fix 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)

58.1.9 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

58.1.10 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)

58.1.11 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.

58.2 WAPT-2.1 Serie

58.2.1 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

58.2.2 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

58.2.3 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)

58.2.4 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.

58.2.5 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.

58.2.6 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`

58.3 WAPT-2.0 Serie

58.3.1 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)

58.3.2 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.

58.3.3 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.

58.3.4 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.

58.3.5 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.

58.3.6 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.

58.4 WAPT 1.8 Serie

58.4.1 WAPT-1.8.2.7393 (2021-11-16)

hash : 75a5de09

This is a security release. **All WAPT 1.8 version below 1.8.2.7393 are vulnerable**.

— [SEC] Upgraded babel python module from 2.5.1 to 2.9.1.
— **[UPD] Updated python lib upgrades urllib3, and requests :**
chardet==4.0.0 requests==2.26.0 urllib3==1.26.7

58.4.2 WAPT-1.8.2.7388 (2021-10-07)

This is a security release. **All Wapt 1.8 version below 1.8.2.7388 are vulnerable**.

Security changelog wapt-1.8.2.7388*

— [SEC] Fixed 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] Sanitized 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.
— [FIX] Fixed the WAPT Console config : when retrieving WAPT Server side https certificate, do not write UTF16 strings in waptconfig. Removed wildcards from CN of certificate to compose certificate filename.
— **[UPD] Updated python modules requirements following urllib3 upgrade**
certifi==2021.5.30 chardet==3.0.2 idna==2.8 requests==2.21.0 urllib3==1.24.3

58.4.3 WAPT-1.8.2.7373 (2021-08-10)

hash : e96e569c

This is a security fix version affected by CVE-2021-38608.

Please visit the *security bulletin* to learn more.

58.4.4 WAPT-1.8.2.7372 (2021-06-21)

WAPT Agent

— [FIX] Fixed regression on macOS build after dependency upgrade.
— [FIX] Fixed `_update_db` : error in for the calculation of `next_update_on` for WAPT package attributes `valid_until` and `forced_install_on`.
— [IMP] Be sure to not use waptguihelper when running as system user.
— [UPD] Added `--use-gui` for **`vscode`** / **`pyscripter`** build-upload of a WAPT package.

The WAPT Server

— [FIX] Fixed regression on proxy setting for the WAPT Server.

SetupHelpers

— [IMP] Added the function `split_arg_string` to split a command line into executable / args list.

58.4.5 WAPT-1.8.2.7357 (2021-02-09)

WAPT Core :

— [FIX] Be tolerant with `target_os = all` in windows.
— [FIX] Fixed `installed_softwares`, ignore error when key can not be opened because of encoding issues (`_winreg` does not handle unicode, but ansi).
— [IMP] show "" instead of None in wapt-get tables.
— [FIX] Updated timestamping of the WAPT Server and openssl hash : http://timestamp.globalsign.com/scripts/timestamp.dll.
— [FIX] Be tolerant if no "id" attribute in installed packages report.
— [FIX] Match properly packages with `target_os = all`.
— [IMP] Prepared `installed_packages` for upgrade to wapt 1.9.
— [IMP] Added `Timeit` class for test purposes.
— [IMP] Disabled sending unused data `waptwua_rules_packages`.
— [FIX] Fixed waptupgrade : regression Bug introduced in revision 85686e4d631adb6e13b25146f3a81f3c09ca082d.
— [FIX] Fixed CA certificate PEM string stored as utf16 in certificate chain when creating a certificate signed by a CA (**enterprise**).

WAPT Console :

— [IMP] Increased width of AD-Site combobox.
— [IMP] Report packages install_id in the WAPT Console.

WAPT Agent :

— [IMP] wapt-get : add `--newest-only` for search.
— [IMP] Improved the WAPT Exit utility : add ExceptionLogger. Change the way exceptions are handled in threads to try to fix issues when the WAPT Exit utility hangs and can not be closed.

WAPT Server :

— [FIX] Do not actually update the listening websocket session ID if it is already set in hosts table.
— [IMP] wapttasks : force remove tasks locks at service startup.

58.4.6 WAPT-1.8.2.7334 (2020-12-03)

hash : 2d15afd9

This is a bugfix release. Ubuntu 16.0.4 amd64 and Debian 10 armhf clients are now supported.

Fixes and enhancements

— [FIX] Fixed base proxy string « » when editing a *profile* package.
— [FIX] Fixed « Unable to create file » when editing a *profile* package.
— [FIX] Do not allow to save a *self-service* rules packages without a name.
— [FIX] Fixed Access violation when importing from file.
— [FIX] Fixed issue with download_icons.
— [FIX] Improved search in the WAPT Console (search on concatenation of software name and software version).
— [FIX] Fixed extract CN from ssl client certificate authentication for `get_auth_token` when windows client computer has an organization (in this case client csr/certificate has a CN=<uuid>,O=<org> subject).
— [FIX] Fixed regression on wakeonlan introduced by backported code from 1.9.
— [FIX] PostgreSQL database not correctly migrating from some 1.8.1.X.
— [IMP] Added key param for `install_msi_if_needed` in **`setuphelpers_windows.py`**.
— [FIX] Fixed `no_fallback` in repositories rules.
— [FIX] Soupsieve python lib is set to 1.9.6 in requirements because later version are Python3 only.
— [FIX] Patch for **`SocketIO`** with proxy.
— [FIX] Fixed triggers for repository sync in PostgreSQL who were not correctly migrated (**Enterprise**).
— [IMP] Two new builders for both the WAPT Server and WAPT Agents : Ubuntu 16.0.4 LTS / ARM x86 Debian 10 (**Enterprise**).
— [IMP] Revert dhparam bits size to 1024 bits in Windows WAPT Server because it took too much time to generate. It can be generated afterward.
— [IMP] Increase default clockskew for signed action to 6 hours (before it was only 1 hour).
— [FIX] Fixed security in waptcrypto : prevent infinite loop in SSLCABundle. :code :*certificate_chain* if issuer certificate and signed certificate have the same subject but one has no `authority_key_identifier`.
— [FIX] waptcrypto : fixed `revoke_cert`, handle list of DNS names for certificates, fixed `AuthorityKeyIdentifier` when regenerating certificate from CSR.
— [FIX] Fixed the WAPT service for `verify_cert_ldap` in the WAPT Agent.
— [FIX] Patched **`pltis_utils`** to display properly long integer in WAPTWUA. The `wsusscn2.cab` file may report KBs with incorrect huge download size up to 1TB.
— [FIX] On a fresh install the admin ACL rights were not properly set up which required a service restart to get them fixed.
— [FIX] Force admin password change on upgrade if the old hash is SHA-1.
— [FIX] Minor fixes for **`uWSGI`** support.
— [FIX] Fixed temporary directories not removed after package import or edit.
— [FIX] Fixed duplicated **`auth_module_ad.py`** module in bad waptwaptenterprise directory on a windows WAPT Server.
— [IMP] Warning of WAPT licence expiration message changed from 14 days to 60 days before expiration.
— [FIX] Fixed broadcast for wakeonlan.
— [FIX] Fixed additional WAPT Server password issues when non ascii character.

Library changes

— [UPD] Update OpenSSL binary from 1.0.2r to 1.0.2u.
— [UPD] Update Python4Delphi lib to 20201020 release.
— [UPD] Build now with Lazarus 2.0.8 and FPC 3.0.4.

58.4.7 WAPT-1.8.2.7269 (2020-06-16)

hash : 757cdc76

— [FIX] Fixed database schema upgrade script for upgrade from WAPT version 1.8.1-6742. Fresh 1.8.2 installation or upgrade from 1.7 or from 1.8.0 or 1.8.1-6758 should not have the issue.
— [IMP] Add key for `install_msi_if_needed`.
— [FIX] Fixed for `no_fallback` for waptwua (**Enterprise**).

58.4.8 WAPT-1.8.2.7267 (2020-06-12)

hash : 46f40312

— [FIX] Fixed database schema upgrade script for upgrade from WAPT version 1.8.1-6742. Fresh 1.8.2 installation or upgrade from 1.7 or from 1.8.0 or 1.8.1-6758 should not have the issue.

58.4.9 WAPT-1.8.2.7265 (2020-06-11)

hash : 339f1996

This is mostly a bugfix release. Support for Linux and macOS clients has also been greatly improved.

Notable enhancements

— [IMP] Improved support for the WAPT Agent running on Linux and macOS. Now the support is almost identical on Windows, Linux and MacOS (all versions) :
— [IMP] The WAPT Agent installs as a service with kerberos registration.
— [IMP] the WAPT Self-service gui available on the 3 platforms (note : support for the latest version of macOS, Catalina, is expected for 1.8.3).
— [IMP] Improved the WAPT Exit utility (on Linux and macOS it is not yet started on system shutdown, it can be triggered by a scheduled task).
— [IMP] `session-setup` for configuring user sessions.
— [IMP] Send message to users and propose upgrades (**Enterprise** only).
— [IMP] OU handling (**Enterprise** only).
— [IMP] The WAPT Self-service authentication can be delegated to the WAPT Server (**Enterprise** only).
— better *SetupHelpers* coverage.
— [IMP] New supported platforms. Now WAPT for linux (WAPT Server and Agent) and macOS (WAPT Agent only) supports :
  — Ubuntu 18.04 and 20.04 ;
  — Debian 8, 9 and 10 ;
  — Centos7 (CentOS 8 as a preview) ;
  — MacOS Sierra, HighSierra, Mojave (note : support for MacOS Catalina expected for WAPT 1.8.3).
— [IMP] Streamlining of development environment for packaging on Linux using VSCode.
— [FIX] Better handling of websocket cleanup when a host is not properly registered. Should improve stability on large WAPT installations.

— [IMP] The selfservice can now be configured for external authentication for desktops that are not in an Active Directory Domain.
— [IMP] The selfservice users can now authenticate on the WAPT Server even when out of the corporate network.
— [IMP] The session setup in run for all packages immediately after **wapt-get upgrade** or **wapt-get install**, so that new packages are already configured in the context of each logged in users (no need to logout / login) (**Enterprise** only).
— [IMP] If secondary repositories are defined in `waptconsole.ini`, additional packages can be selected when editing hosts, groups or self-service packages.
— [IMP] When editing group or self-service packages, one can define the Target OS of the package.
— [IMP] Remote message to logged in users is using the same custom dialog box for Windows, Linux and macOS.
— [IMP] Remote message to logged in users can display the same custom logo as self-service (**Enterprise** only).
— [IMP] The IP/Subnet match in repository access rules is based on the « main IP » of the host (source IP from which the host is reaching the WAPT Server, if the WAPT Server is public, this is usually the external IP of the router) (**Enterprise** only).
— [IMP] Added Remote host Shutdown and remote host Reboot from the WAPT Console if enabled in `wapt-get.ini` (`allow_remote_shutdown` and `allow_remote_reboot`) (**Enterprise** only).
— [IMP] Added a *no fallback* checkbox in repositories access rule to prevent host using main repository in case secondary ones are not reachable (when main repository bandwidth is limited, having all hosts reaching the main repository can slow down access to the main site) (**Enterprise** only).
— [FIX] Make sure the WUA install task are executed after packages are installed (**Enterprise** only).

Other enhancements

— [IMP] The **Cmd** console is hidden when **wapt-get session-setup** is running, to limit annoyance for users.
— [IMP] Improved WUA direct download option in the WAPT Console (**Enterprise** only).
— [IMP] Can now use Microsoft url for WUA in rules (**Enterprise** only).
— [FIX] Improved background icons loading in WAPT Self-service.
— [FIX] Better inventory of `lastboottime` and `get_domain_info`.
— [FIX] Better handling of other local install of Python on client computer (eg. conflict with local Anaconda Python installation).
— [IMP] Allows to have multiple private repo content displayed in the WAPT Console.
— [IMP] Remote repository : it is now possible to prevent a fallback.
— [FIX] Better handling of icons in the WAPT Self-service.
— [IMP] Improved support for **VSCode**.
— [FIX] Better handling of ipv6 in the WAPT Console and the inventory.
— [IMP] `wapt_admin_filter` : local admininistrators can be filtered out like normal user in the WAPT Self-service.
— [IMP] Larger support for *SetupHelpers* on macOS.
— [FIX] The WAPT Server logs are properly redirected to `/var/log/waptserver.log`.
— [FIX] Fixed package caching : packages are deleted after each successful installation (rather than at the end of the whole upgrade) to better preserve local disk space.
— [IMP] Allow usage of url for changelog in control file.
— [IMP] Better support for Windows Update download directly from Microsoft if the WAPT Server is not reachable.
— [FIX] Better handling of upgrade from Community version to Enterprise version.
— [IMP] Improved local store skin and translations.
— [FIX] Bugfixes and minor GUI improvements.

Library changes in WAPT-1.8.2.7265

— [CHANGE] Replaced **python-ldap** with **ldap3**.
— [FIX] Upgraded **ujson** on the WAPT Server and the WAPT Agent running on Linux.

Removed features with WAPT-1.8.2.7265

— [REMOVED] Autoconfiguration of repositories based on SRV DNS fields (it was not working anymore anyway).

Caveats when using WAPT-1.8.2.7265

— [CAV] **waptexit** is not run automatically on shutdown on Linux or macOS (current issue with **systemd** / launched integration).
— [CAV] **wapttray** is not yet available on Linux and macOS.
— [CAV] MacOS Catalina is supported by the WAPT Agent, however **WAPTSelfService** and **waptexit** are not yet supported.

58.4.10 WAPT-1.8.2.7265 RC2 (2020-05-29)

hash git : 339f1996

Avertissement : This is a Release Candidate version for testing and evaluation only and should not be installed on production system.

This is mostly a bugfix release. Support for Linux and macOS clients has greatly improved.

Notable enhancements over 1.8.2 RC1

— [IMP] Improved the session setup in run for all packages immediately after :command :`` or install, so that new packages are already configured in the context of each logged in users (no need to logout / login) (**Enterprise** only).
— [IMP] If secondary repositories are defined in `waptconsole.ini`, additional packages can be selected when editing hosts, groups or self-service packages.
— [IMP] When editing group or self-service packages, one can define the target OS of the package.
— [IMP] Remote message to logged in users is using the same custom dialog box for windows, linux and macOS.
— [IMP] Remote message to logged in users can display the same custom logo as self-service (**Enterprise** only).
— [IMP] The IP / Subnet match in repository access rules is based on the *main IP* of the host (source IP from which the host is reaching the WAPT Server, if the WAPT Server is public, this is usually the external IP of the router) (**Enterprise** only).
— [IMP] Added remote host shutdown and remote host reboot from the WAPT Console if enabled in wapt-get.ini (`allow_remote_shutdown` and `allow_remote_reboot`) (**Enterprise** only).
— [IMP] Added a *no fallback* checkbox in repositories access rule to prevent hosts using main repository in case secondary repositories are not reachable (when main repository bandwidth is limited, having all hosts reaching the main repository can slow down access to the main site) (**Enterprise** only).
— [FIX] Make sure WUA install task are executed after packages install (**Enterprise** only).

Other enhancements over 1.8.2 RC1

— [IMP] the `cmd` console is hidden when session-setup is running, to limit annoyance for users.
— [IMP] WUA direct download option in the WAPT Console (**Enterprise** only).
— [IMP] Can now use Microsoft url for WUA in rules (**Enterprise** only).
— [IMP] Improved background icons loading in self-service.

Removed features

None

Caveats

Same as RC1

58.4.11 WAPT-1.8.2.7165 RC1 (2020-05-29)

hash git : 1387b38f

Avertissement : This is a Release Candidate version for testing and evaluation only and should not be installed on production system.

This is mostly a bugfix release. Support for Linux and macOS clients has greatly improved.

Notable enhancements in WAPT-1.8.2.7165 RC1

— [IMP] improve support for the WAPT Agent on Linux and macOS. Now the support is almost identical on Windows, Linux and macOS (all versions) :
— [IMP] The WAPT Agent installs as a service with kerberos registration.
— [IMP] waptselfservice gui available on the 3 platforms (note : support for the lastest version of MacOS, Catalina, is expected for 1.8.3).
— [IMP] Improved the WAPT Exit utility (on Linux an macOS it is not yet started on system shutdown, it can be triggered by a scheduled task).
— [IMP] session-setup for configuring user sessions.
— [IMP] send messagebox to users and propose upgrades (**Enterprise** only).
— [IMP] OU handling (**Enterprise** only).
— [IMP] waptselfservice authentication can be delegated to the WAPT Server (**Enterprise** only).
— [IMP] Better *SetupHelpers* coverage.
— [IMP] add new supported platform. Now WAPT for linux (WAPT Server and Agent) and MacOS (WAPT Agent only) supports :
  — Ubuntu 18.04 and 20.04 ;
  — Debian 8, 9 and 10 ;
  — Centos7 (CentOS 8 as a preview) ;
  — MacOS Sierra, HighSierra, Mojave (note : support for MacOS Catalina expected for WAPT 1.8.3).
— [IMP] streamlining of development environment for packaging on Linux using VSCode.
— [FIX] better handling of websocket cleanup when a host is not properly registered. Should improve stability on large WAPT installation.
— [IMP] selfservice can now be configured for external authentication for desktops that are not in an Active Directory Domain.
— [IMP] selfservice users can now authenticate on selfserver even when out of the corporate network.

Other enhancements in WAPT-1.8.2.7165 RC1

— [FIX] Better inventory of `lastboottime` and `get_domain_info`.
— [FIX] Better handling of other local install of Python on client computer (eg. conflict with local Anaconda Python installation).
— [IMP] Allow to have multiple private repo content displayed in the WAPT Console.
— [IMP] Improved remote repository to make possible to prevent a fallback.
— [FIX] Better handling of icons in selfservice.
— [IMP] Improved support for VSCode.
— [FIX] Better handling of ipv6 in the WAPT Console and inventory.
— [IMP] `wapt_admin_filter` : local admin can be filtered out like normal user in selfservice.
— [IMP] Added a larger support for *SetupHelpers* on macOS.
— [FIX] WAPT Server logs are properly redirected to `/var/log/waptserver.log`.
— [FIX] Better support for package caching : packages are deleted after each successful installation (rather than at the end of the whole upgrade) to better keep local disk space.
— [IMP] Allow usage of url for changelog in control file.
— [IMP] Better support for Windows Update download directly from Microsoft if the WAPT Server is not reachable.
— [FIX] Better handling of upgrade from Community version to Enterprise version.
— [IMP] Improved local store skin and translation.
— [FIX] Bugfixes and minor gui improvements.

Library changes in WAPT-1.8.2.7165 RC1

— [REF] replaced **`python-ldap`** with **`ldap3`**.
— [FIX] upgraded **`ujson`** on the WAPT Agent and Server on Linux.

Removed featured with WAPT-1.8.2.7165 RC1

— autoconfiguration of repositories based on SRV DNS fields (it was not working anymore anyway).

Caveats when using WAPT-1.8.2.7165 RC1

— [CAV] The WAPT Exit utility is not run automatically on shutdown on Linux or MacOS (current issue with systemd / launched integration).
— [CAV] the WAPT System Tray utility is not yet available on Linux and macOS.
— [CAV] MacOS Catalina is supported by the the WAPT Agent, however WAPTSelfService and the WAPT Exit utility are not yet supported.

58.4.12 WAPT-1.8.1-6758 (2020-03-06)

(hash bb93ce41)

WAPT Server :
— [REF] Refactoring in **`postconf.py`** to remove old migration scripts from MongoDB.
— [REF] Refactoring for **`winsetup.py`** to create now a `dhparam` for **`nginx`** on Windows.
— [REF] Refactoring for repositories : changed `repo_diff` to `remote_repo_diff` and added the parameter `remote_repo_websockets` (default `True`) to the WAPT Server.
— [IMP] disable cache on **`nginx`** for Windows and Linux on wapt packages / exe.
WAPT Agents :
— [REF] Changed the parameter `waptservice_admin_auth_allow` to `waptservice_admin_filter`.

— [REF] Deleted resync functions for remote repo.
— [IMP] Improved the default parameter `local_repo_sync_task_period` to `2h`.
— [FIX] Fixed wapt-get / WAPT service debug when downloading a WAPT package on Linux while not using a sudo account.
— [FIX] Fixed **`plist`** in macOS.
— [IMP] Can now have relative path for WAPT packages / directories in **`wapt-get`**.
— [IMP] Templates have by default setup_uninstall / update etc…
— [IMP] Improved templates for **`vscode`**.

The WAPT Console

— [IMP] Added possibility of template packages for *`.deb`* / *`.rpm`* / *`.pkg`*.
— [FIX] Fixed error with *`.msi`*, *`.exe`*, etc in PackageWizard explorer.
— [IMP] Can now choose `editor_for_packages` directly in the WAPT Console configuration file.
— [UPD] Some cosmetic / translations improvements for GUI to deploy the WAPT Agent.

58.4.13 WAPT-1.8.1-6756 (2020-02-17)

(hash 43394f3b)

Bug fixes and small improvements

— [IMP] Improved the WAPT Console : improve the refresh of hosts grid when a lot of hosts are selected (improved by a factor of around 5).
— [FIX] Fixed the WAPT Server Database connections management : do not close the database on teardown as it should not occur, and seems to trigger some issue when triggering a lot of tasks on remote hosts (error « database is closed »).
— [FIX] Fixed the WAPT Console : Do not « force » install when triggering the upgrade on remote hosts, to avoid reinstalling softwares when already up to date.
— [IMP] use *ldap authentication* only if session and admin fail (avoid waiting for timeout when ldap is not available but one wants to login with plain admin user).
— [FIX] wapt-get upload : encode user and password in `http_upload_package` to allow non ascii in admin password.
— [IMP] Improved the WAPT Console : Disable auto search on keywords.
— [IMP] Use DMI `System_Information.Serial_Number` information for serialnr Host field instead of `Chassis_Information.Serial_Number` because System_Information is more often properly defined.
— [ADD] Added `uuid` to the list of searched fields when only “host” is checked in filters in the WAPT Console.
— [IMP] Improved **`Nginx`** config : disable caching.
— [IMP] Fixed **`vscode`** project template.

58.4.14 WAPT-1.8.1-6742 (2020-02-12)

(hash 80dbdbe7)

Major changes

— [ADD] In the WAPT Console, added a page to show packages install status summary (merge) of all selected hosts, grouped by `package`, `version`, `install status`, with count of hosts ;

Context menu allow to apply selectively the pending actions. On enterprise, one can apply safely the updates (only packages for which there is no running process on client side).

— [IMP] Prevent users from saving a host package if targeted host(s) do not accept their personal certificate. (Checked on the WAPT Console when editing / mass updating host packages, and on the WAPT Server when uploding packages).

The personal certificate file *`.crt`* **MUST** contain at first the personal certificate, followed by the issuer CA certificates, so that WAPT can rebuild the certificate chain and check intersection with host’s trusted certificates.

Important note about SSL client side authentication

In the **Nginx** configuration, be sure to reset the headers `X-Ssl-Authenticated` and `X-Ssl-Client-DN` as the WAPT Server **trusts** these headers if the SSL cient side authentication is enabled in `waptserver.ini`.

If SSL client side authentication is setup these headers can be populated by `proxy_set_header` with result of `ssl_verify_client` as explained in ./wapt-security/security-configuration-certificate-authentication.html#enabling-client-side-certificate-authentication.

Fixes and detailed changelog

— [FIX] Fixed security and updated **waitress** module to 1.4.3 (CVE-2020-5236).
— [FIX] Fixed security with blank `X-Ssl*` headers in default **nginx** templates.
— [FIX] Fixed regression in **kerberos register_host** did not work anymore.
— [IMP] On the WAPT Server, <repository root>/wapt/ssl dir is moved automatically on winsetup / postconf to (per default) <repository root>/ssl, a /ssl location is added. This /ssl should be accessible from clients at the location specified by the WAPT Server parameter `clients_signing_crl_url` (in `waptserver.ini`).
— [IMP] Improved logs readability. Log count of used database connections from pool on the WAPT Server to troubleshoot database connection issues. Log level can be specified by subcomponent with `loglevel_waptcore`, `loglevel_waptserver`, `loglevel_waptserver.app`, `loglevel_waptws`, `loglevel_waptdatabase` defined in `waptserver.ini` ;
— [IMP] Reworked explicit database Open/close on the WAPT Server to not get a database connection from pool if not useful. It prevents exhaustion of database connections ;
— [IMP] waptwinsetup : do not create unused directories `wapt-group` and `waptserverlog` ;
— [ADD] Added *.msu* and *.msix* extensions for Package wizard setup file dialog ;
— [ADD] Fallback with os._exit(10) for the WAPT service restart. Added a handler in **nssm.exe** configuration to honor the restart ;
— [IMP] Increased waitress threads to 10 on the WAPT service ;
— [IMP] Lowered the default number of pooled database connections (`db_max_connections`) to 90, to be lower than postgresql default of 100 ;
— [IMP] Improved the WAPT Server : allow kerberos or ssl authentication check in the WAPT Server only if enabled in `waptserver.ini` config file ;
— [IMP] Improved the WAPT Console : Allow update of host package only if user certificate is actually allowed on the host (based on last update of host status in database) ;
— [ADD] Added in the WAPT Console / build of the WAPT Agent : added checkbox to specify to include or not non certificate authority certificates in build. The normal setup would be to uncheck this, to not deploy non CA certificates, on wapt root CA ;
— [IMP] Add and option to disable automatic hiding of panels…
— [IMP] Add explicit `AllowUnauthenticatedRegistration` task to the Windows waptserversetup.
— [IMP] waptsetup : Remove explicit VCRedistNeedsInstall task. Use `/VCRedistInstall` = `True` / `False` if you need to force install or force not install vcredist VC_2008_SP1_MFC_SEC_UPD_REDIST_X86.
— [FIX] **wapt-get.exe** : use wapt-get.ini for **wapt-get scan-packages** and **wapt-get update-packages**" actions.
— [FIX] **wapt-get** : authentication asked when checking if the WAPT Server is available (ping) and client ssl authentication is enabled.
— [IMP] WAPT client : if client ssl authentication failed with http error 400, retry without ssl authentication to be able to ask for new certificate signing.
— [FIX] Fixed the WAPT Server register behavior : revert over rev 6641 : sign host certificate if an authenticated user is provided or data is signed with a key which can be verified by existing certificate in database for this host uuid.
— [IMP] Improved the WAPT Server register behavior : when receiving 401 from the WAPT Server when registering, retry registering without ssl authentication.
— [IMP] wapt client : be sure to have proper host private key saved on disk when receiving signed certificate from the WAPT Server.
— [IMP] Improved the WAPT Console : advanced filters for selected host packages status. Filter on *Install status* and *Section* + *keyword*. *Pending* button to show only pending installations / removes.
— [ADD] **wapt-get make-template / edit package** : Add `.vscode` directory. Add template project for vscode ;

- [FIX] Fixed the WAPT Console : ssl authentication for mass package dependencies / conflicts updates ;
- [FIX] Fixed the WAPT Console : import packages from external repos with ssl authentication.
- [IMP] Backports from master :
  - target OS in import packages ;
  - choose editor for packages in linux in cmdline.
- [IMP] backports from master :
  - Refactoring for `HostCapabilities.waptos` ;
  - Added new `target_os` unix for mac and linux so `target_os` = `windows`, `darwin` (for mac), `linux` or `unix`.
- [FIX] `WAPT.wapt_base_dir`.
- [FIX] `makepath` in Linux / macOS.
- [IMP] Refactoring / fixes for *SetupHelpers*.
- [FIX] Fixed `rights_to_check` in repo-sync client.
- [FIX] for repo-sync :
  - [ADD] Added two *SetupHelpers* for linux : `type_debian` and `type_redhat`.
  - [IMP] Indent the local `sync.json`.
- [IMP] Use `get_os_version` and `windows_version_from_registry` instead of `windows_version`.
- [IMP] Improved `windows_version_registry` for `get_os_version` on windows.
- [IMP] Backported `host_capabilities.os` from master.
- [FIX] Fixed **`make-template`** for malformed *`.exe`* installer.
- [ADD] Added automatic maintenance of a CSR for client authentication certificates signed by the WAPT Server :
  - Default CSR lifetime to 30 days.
  - Check renewal of client certificate CSR every hour.
- [ADD] Added a parameter for the next update time of CRL.
- [ADD] Added `clients_signing_crl_url`, `clients_signing_crl_days`, `known_certificates_folder` to the parameters of the WAPT Server.
- [ADD] Added a `/ssl` location in nginx templates.
- [ADD] Added `crl_urls` in client authentication signed certificates.
- [ADD] Added a scheduled task to renew the WAPT Server side CRL.
- [ADD] Added `clients_signing_crl` WAPT Server parameter to add the WAPT client certificate to the WAPT Server's CRL when host is unregistered.
- [ADD] Added **`revoke_cert`** method to `SSLCRL` class.
- [ADD] Added a `authorityKeyIdentifier` to the client authentication CSR.
- [IMP] Force restart if Windows task is broken.
- [FIX] Fixed the WAPT service : use `sys._exit(10)` to ask **`nssm`** to restart service in case of unhandled exception in the WAPT service (loops, etc.).
- [FIX] Fixed the WAPT Agent : do not log / store into database Wapt.runstatus if not changed.
- [FIX] Fixed the WAPT Server postconf for rights on some wapt directories.
- [ADD] Added mutual conflicts to deb/rpm packages for the WAPT Agent and the WAPT Server to avoid simultaneous install.

58.4.15 WAPT-1.8.0-6641 (2020-01-24)

(hash 3dbb3de8)

Major changes

— [ADD] Added WAPT Agent for Linux Debian 8, 9 , 10, Linux Centos 7, Ubuntu 18, 19 and MacOS. The packages are named `wapt-agent` and available in https://wapt.tranquil.it/wapt/releases/latest/.
— [IMP] Improved the repository access rules defined in the WAPT Console. Depending of client IP, site, computername, one can define which secondary repository URL to use (**Enterprise** only).

As a consequence, the DNS query method (with SRV records) is no more supported for repositories

— [IMP] The package and signature process has been changed to be compatible with **`python3`**. Serialization of dict is now sorted by key alphabetically to be deterministic across python versions. The WAPT Agents prior to version 1.7.1 will not be able to use new packages. (see git hash SHA-1 : f571e55594617b43ed83003faeef4911474a84db).
— [NEW] A WAPT Agent can now be declared as a secondary remote repository. Integrated syncing with main WAPT Server repository is handled automatically. (**Enterprise** only)
— [NEW] The WAPT Console can now run without elevated privileges. The build of `waptagent` / `waptupgrade` package are done in a temporary directory. **When editing a package from the WAPT Console, :program :`PyScripter`should be launched with elevated privileges**.

Note : One could deploy the WAPT Agent with GPO without actually rebuilding a `waptagent`. Command line options are available on stock waptsetup-tis.exe to configure repo url (`/repo_url =`), WAPT Server url (`/wapt_server =`), WAPT Server certificate bundle location (`/CopyServersTrustedCA =`), packages certificates checking (`/CopyPackagesTrustedCA =`), `/use_random_uuid`, `/StartPackages`, `/append_host_profiles`, `/DisableHiberBoot`, `/waptaudit_task_period`.

Some options are still missing and may be added in a future release.

— [IMP] package filename now includes a hash of package content to make it easier to check if download is complete and if package has been scanned (improved speed for large number of packages).
— [SEC] the WAPT admin password **MUST** be regenerated (with postconf) ; if it is not *pbkdf2* based. See in your `waptserver.ini` file, `wapt_password` **MUST** start with **$pbkdf2-**.

Fixes and detailed changelog

— [SEC] The WAPT Agent can optionally be digitally signed, if (1) Microsoft **`signtool.exe`** is present in <`wapt>utils` and `(2) if there is a pkcs#12 :mimetype:`.p12` file with the same name as the personal certificate `.crt` file, and (3) the certificate is encrypted with the same password ;
— [IMP] wapt-get.py can be run on linux and macos in addition to windows ;
— [IMP] Improved the WAPT Console host's packages status reporting : now displays current version with *NEED-UPGRADE*, *NEED-REMOVE*, *ERROR* status and future version with *NEED-INSTALL* status ;

The status is stored in the WAPT Server's database `HostPackagesStatus` so it can be queried for reporting ;
— [IMP] Improved *SetupHelpers* : there now different *SetupHelpers* for each operating system family ;
— [ADD] Added in the WAPT Console : action to safely trigger upgrades on remote hosts only if associated processes (`impacted_process` control attribute) are not running, to avoid disturbing users (**Enterprise** only) ;
— [ADD] **`wapt-get --service upgrade`** : added handling of `--force`, `--notify_server_on_start = 0/1`, `notify_server_on_finish`= ``0/1` switches ;
— [IMP] package signature's date is now taken in account when comparing packages ;
— [ADD] `host_ad_site` key in `[global]` in `wapt-get.ini` to define a *fake* Active Directory site for the host ;
— [ADD] Added in the WAPT Console / packages grid : if multiple packages are selected, the associated *show clients* grid shows the status of packages for all selected clients (**Enterprise** only) ;
— [ADD] `waptagent` build : added checkbox to enable repository rules lookup when installing The WAPT Agent (**Enterprise** only) ;
— [ADD] Added in the WAPT Console / import packages : do not reimport existing dependencies. Checkbox to disable import of dependencies ;

— [IMP] wapt-scanpackages speed optimizations : do not re-extract certificates and icon for skipped package entries. use md5 from filename if supplied when scanning.
— [FIX] Fixed arguments in the WAPT Exit utility for `only_if_not_process_running` and `install_wua_updates` (bool) ;
— [FIX] Fixed the WAPT Agent / WAPT WUA enabled setting reset to *False* when upgrading with `waptagent` and enabled ;
— [FIX] Fixed the WAPT Server / waptwua repository : all cabs files are now in root directory instead of microsoft original file tree. The files are moved when upgrading to 1.8 ;
— [IMP] waptupgrade package : increment build number if building a new WAPT Agent of the same main wapt version ;
— [NEW] New WAPT Server parameter `trusted_signers_certificates_folder` :
Path to trusted signers certificate directory. If defined, only packages signed by this trusted CA are accepted on the WAPT Server when uploading through the WAPT Server ;
— [NEW] New WAPT Server parameter `remote_repo_support` : if true, a task is scheduled to scan repositories (`wapt`, `waptwua`, `wapt-hosts`) that creates a `sync.json` file for remote secondary repositories ;
— [IMP] when building the WAPT Agent, do not include non CA packages certificates by default in the WAPT Agent. A checkbox is available to still enable non CA certificates to be scanned and added ;
— [IMP] when building the WAPT Agent, one can add or remove certificates in the grid with `Ctrl+Del` or drag and drop ;
— [FIX] Fixed the WAPT Console / host packages status grid : fixed `F5` refresh ;
— [IMP] Improved the WAPT Console / build of the WAPT Agent : build an Enterprise WAPT Agent even if no valid licence (**Enterprise** only) ;
— [FIX] `forced_update_on` control attribute : do not take into account for `next_update_on` if in the past ;
— [IMP] Improved the WAPT Console : try to accept the WAPT Server password with non ASCII characters ;
— [REMOVED] waptstarter : remove *socle* from default host profile ;
— [IMP] `waptagent` build : rework of the WAPT Server certificate path relocation when building / installing ;
— [SEC] do not sign the WAPT Agent certificate if no valid human authentication (admin, passwd or ldap) or kerberos authentication has been provided :
— be explicit on authentication methods ;
— store registration authentication method in database only if valid human authentication or kerberos authentication has been provided ;
— when registering, be sure we trust an already signed certificate with CN matching the host ;
— store the signed host certificate in the WAPT Server database on proper registration ;
— [IMP] some syntax preparation work for future python3 ;
— [IMP] some preparation work for detailed ACL handling (**Enterprise** only) ;
— [FIX] Do not enable client ssl authentication by default in the WAPT Server as nginx reverse proxy server is perhaps misconfigured ;

Python libraries / modules updates

— use **`waitress`** for the WAPT service wsgi server instead of unmaintained **`Rocket`**` ;
— **`Flask-SocketIO 3.0.1`** –> **`Flask-SocketIO 4.2.1`** ;
— **`MarkupSafe 1.0`** –> **`MarkupSafe 1.1.1`** ;
— **`python_ldap-2.4.44`** –> **`python_ldap-3.2.0`** ;

58.5 WAPT 1.7 and older

58.5.1 WAPT-1.7.4-6237 (2019-11-18)

(hash 1c00cefd)

- [FIX] Fixed the WAPT Server : add fix to workaround flask-socketio bug (AttributeError : "Request" object has no attribute "sid");
- [IMP] Improved the WAPT Server : be sure the database is closed before trying to open it (for dev mode);
- [IMP] Improved the WAPT Server : add logs messages when an exception message is sent back to the user;

58.5.2 WAPT-1.7.4-6234 (2019-11-14)

(hash ad237eee)

- [IMP] Improved the WAPT Server : upgrade **peewee** database python module to 3.11.2. Explicit connection handling to database to track potential limbo connections (which could lead to database pool exhaustion);
- [FIX] waptwua : trap exception when pushing WU to Windows cache to allow valid updates to be installed even if some could not be verified properly;

58.5.3 WAPT-1.7.4-6232 (2019-10-31)

(hash2090b0e6d52cecfb04f8fa4c279e7c0a0252d6e2

- [FIX] **wapt-get session-setup** : fix bad print in **session_setup**. Regression introduced in b30b1b1a550a4 (1.7.4.6229);

58.5.4 WAPT-1.7.4-6230 (2019-10-23) (not released)

(hash 391d382f)

- [IMP] return the WAPT Server git hash version and edition in ping and `usage_statistics`;
- [IMP] be sure to have `server_uuid` on windows when during setup;
- [FIX] `.git` partially included in built package `manifest`;

58.5.5 WAPT-1.7.4-6229 (2019-10-23)

(hash b30b1b1a)

- [FIX] 100% cpu load on one core on the WAPT Server even when Idle;
- **python-engineio** upgrade to 3.10.0;
- **python-socketio** upgraded to 4.3.1;
- [IMP] Do not try run **session_setup** on packages which do not have one defined;
- [IMP] limit text output on the WAPT Console (for faster output);

58.5.6 WAPT-1.7.4-6223 (2019-10-15)

(hash 86ddeaa2d)

- [FIX] Newlines in packages installs logged output ;
- [FIX] Allow nonascii utf8 encoded user and password for the WAPT Server basic authentication ;
- [UPD] Updated the WAPT Console : Default package filtering to x64 and the WAPT Console locale to avoid mistakes when importing ;
- [IMP] Improved the WAPT Console : increase default Port Socket listening test timeout (for rdp, remote service access etc..) to 3s instead of 200ms ;
- [IMP] Improved the WAPT Console : sort OU by description in treeview :

Right click changes current row selection in OU treeview ;

- [NEW] option to set `waptservice_password` = `NOPASSWORD` in waptstarter installer ;
- [FIX] grid sorting for package / version / size of packages ;
- [FIX] Do not create the WAPT Console link for starter ;
- [NEW] **`wapt-scanpackages`** : add an option to update the local packages database table from `Packages` file index ;
- [FIX] regression introduced in previous build : `maturities` = `PROD` and `maturities` = `''` are equivalent when filtering allowed packages ;
- [FIX] Fixed the WAPT Console : grid headers too small for highdpi ;
- [UPD] waptupgrade package filename : keep old naming without *all* arch (for backward compatibility) ;
- [IMP] `waptservice_timeout` = `20` seconds now ;
- [FIX] Active Directory authentication for the WAPT Console with non ASCII chars ;
- [IMP] missing french translations for columns in *Import packages* grid ;
- [FIX] be sure to terminate output threads in waptwinutils.run ;
- [IMP] avoid showOnTop flickering for VisLoading ;
- [IMP] `setuphelpers.run_powershell` : added `$ProgressPreference` = `SilentlyContinue` prefix command ;
- [SEC] Secured the WAPT service : protect test of `host_cert` date if file is deleted outside of service scope ;
- [IMP] WaptBaseRepo class :
- packages cache handling when repo parameters (filters…) are changed ;
- allow direct setting of cabundle for WaptBaseRepo ;
- keep a fingerprint of input config parameters ;
- [UPD] set a fallback calculated `package_uuid` value in database for compatibility with old package status reports ;

58.5.7 WAPT-1.7.4-6196 (2019-09-27)

(hash f9cb3ebd)

- [IMP] revert package naming of waptupgrade to previous one to ease upgrade from previous wapt ;
- [IMP] increase `waptservice_timeout` to 20 seconds per default ;
- [FIX] Active Directory authentication when there are non ascii chars (encoding) ;
- [FIX] missing french translations for columns in Import packages grid ;
- [IMP] set a fallback calculated `package_uuid` in database for old package without `package_uuid` attribute in database status report ;
- [NEW] **`wapt-scanpackages`** : add an option to update the local Packages database table from Packages file index ;
- [NEW] option to filters `maturities` ;

58.5.8 WAPT-1.7.4-6192 (2019-09-17)

(hash 3e00ac6688)

— [SEC] update python modules **python-engineio** and **werkzeug** to fix vulnerability CVE-2019-14806
GHSA-j3jp-gvr5-7hwq

— [UPD] Python modules :
  — **eventlet 0.24.1** –> **eventlet 0.25.1**;
  — **flask 1.0.2** –> **flask 1.1.1**;
  — **greenlet 0.4.13** –> **greenlet 0.4.15**;
  — **itsdangerous 0.24** –> **itsdangerous 1.1.0**;
  — **peewee 3.6.4** –> **peewee 3.10**;
  — **python-socketio 1.9.0** –> **python-socketio 4.3.1**;
  — **python-engineio 3.8.1** –> **python-engineio 3.9.3**;
  — **websocket-client 0.50** –> **websocket-client 0.56**;
— [UPD] default `request_timeout` = **15s** for client websockets;
— [FIX] when building packages, excluded directories (for example *.git* or *.svn*) were still included in `manifest` file;
— [UPD] Do not canonicalize package filenames by default when scanning The WAPT Server repository to ease migration from previous buggy wapt;
— [FIX] package filename not rewritten in `Packages` when renaming package;
— [NEW] **wapt-scanpackages** : added explicit option to trigger rename of packages filenames which do not comply with canonic form;
— [NEW] **wapt-scanpackages** : added option to provide proxy;
— [UPD] return **OK** by default in package's audit skeleton;
— [IMP] Improved the WAPT Console cosmetic : minheight 18 pixels for grid headers
— [FIX] Fixed the WAPT Server database model : bad default datatype in `model.py` for `created_by` and `updated_by` (were not used until now);
— [FIX] `ensure_unicode` for *.msi* output : try *cp850* before *utf16* to avoid Chinese garbage in run output;
— [NEW] added `connected_users` to `hosts_for_package` provider;
— [FIX] use **win32api** to get local connected IPV4 IP address instead of socket module. In some cases, socket can not retrieve the IP;
— [FIX] **wapt-get unregister** command not working properly;
— [NEW] Waptselfservice : added option in `wapt-get.ini` to disable unfiltered packages view of local admin;
— [IMP] Waptselfservice : 4K improvements;
— [FIX] Waptselfservice :

— packages *restricted* were shown in selfservice / now corrected;
— if the repo have no packages segmentation error / now corrected;
— if the repo have changed segmentation error / now corrected;

58.5.9 WAPT-1.7.4.6165 (2019-08-02)

(hash f153fab4)

Improvements

— [NEW] added **unregister** action to wapt-get ;
— [UPD] improvements with the alt logo in the self-service ;

Changes

— [UPD] use version to build the package name of unit, groups and profile type package, like for base packages ;
— [UPD] added logs to **uwsgi** ;

Fixes

— [FIX] bugfixes with the icons of the app self-service ;
— [FIX] bugfixes with the logos in the self-service ;
— [UPD] Updated the WAPT Exit utility : do not cancel tasks on CloseQuery ;
— [UPD] patch `server.py` earlier to avoid *execute cannot be used while an asynchronous query is underway* ;
— [FIX] Fixed the WAPT Exit utility doing nothing if `allow_cancel_upgrade = False` and `waptexit_disable_upgrade = False` ;
— [FIX] fix issue with merge of wsus rules (can cause memory errors if more than one wsus package is applied on a host) (**Enterprise** only) ;
— [FIX] fix wua auto `install_scheduling` issue ;
— [FIX] Fixed the WAPT Exit utility : add a watchdog to workaround some cases where it hangs (threading issue ? ?) ;

58.5.10 WAPT-1.7.4.6143 (2019-06-25)

(hash da870a2c)

Improvements

— [IMP] wapt self service application is now fully usable. It is available in `<wapt>waptself.exe` ;
— [ADD] option to set a random UUID instead of BIOS UUID at setup. This is to workaround for bugged BIOS with duplicated ids ;
— [IMP] better Sphinxdocs for WAPT Libraries ;

Changes

— [UPD] behavior change : Use computer FQDN from tcpip registry entry (first NV Hostname key) then fixed domain then DHCP ;
— [FIX] inverted Zip and signature steps in package build operations to workaround issue with Bad Magic Number when signing already zipped big packages ;
— [NEW] Add `use_ad_groups` wapt-get `[global]` parameter to activate groups from AD (this is a time consuming task, so better not activate it…) ;

Fixes

- [FIX] appendprofile infinite loop during setup;
- [FIX] read forced uuid from `wapt-get.ini` earlier to avoid loading a bad host certificate in memory if changing from bios uuid to forced uuid;
- [FIX] setting `use_random_uuid` in `waptagent.iss`;
- [FIX] waptstarter setup : force deactivate the WAPT Server, hostpackages;
- [FIX] include waptself in waptstarter, do not include innosetup in waptstarter;
- [FIX] `ensure_unicode` : add *utf16* decoding test before *cp850*;
- [FIX] add `ensure_unicode` for tasks logs to avoid unicode decode errors in **`get_tasks_status`** callback;
- [NEW] host status : add `boot_count` attribute;
- [FIX] fix potential float / unicode error when scanning windows updates (**Enterprise** only);
- [FIX] handles properly excluded files in package signatures;
- [FIX] Fixed the WAPT Exit utility : avoid some work after checking if the WAPT service is running if it is not running;
- [FIX] a case where WAPTLocalJsonGet could loop forever if authentication fails;
- [FIX] `setup.pyc` in `manifest` but not in zipped package :
- exclude exactly [”*.svn*”,”*.git*”, “*.gitignore*”,”`setup.pyc`”] when signing and zipping;
- **`inc_build`** before signing;
- [UPD] add `use_ad_groups` setting in the WAPT Agent build. Default to *False* (**Enterprise** only);
- [FIX] better detection of `waptbasedir` for `python27.dll` loading;
- [FIX] allow to sign source package directory to workaround a bug in python zipfile (bad magic number);
- [NEW] added a `htpasswd` password file method for restricted access to only **`add_host`** method :

allows **`add_host`** if provided host certificate is already signed by the WAPT Server and content can be verified;
- [FIX] **`wapt-get.exe`** crash with « can not load… » when python 3.7 is installed from MS store;
- [FIX] load `private_dir` conf parameter earlier;
- [UPD] put a *rnd-* in front of randomly generated uuid;

added a checkbox to use random uuid (if not already defined in `wapt-get.ini`);
- [UPD] SSL CA certifi library;
- [IMP] utf8 decode user /password in localservice authentication;
- [UPD] allow authentication on the local WAPT service with token;
- [NEW] filter packages on hosts based on the `valid_from` and `valid_until` control attributes;

force update sooner if `valid_from` or `valid_until` or `forced_install_on` is sooner than regular planned `update_period`;
- [FIX] events reporting from service tasks;
- [FIX] Fixed the WAPT Exit utility : **`waptexit`** not closing of writing for running tasks but auto upgrade has been disabled;
- [ADD] added `waptexit_disable_upgrade` option to **`waptexit`** to remove the triggering of upgrade from the WAPT Exit utility, but keep the waiting for pending and running tasks :
  - “running_tasks” key in the WAPT service checkupgrades.json. Was not reflecting an up to date state.
- [NEW] add new packages attributes : `name`, `valid_from`, `valid_until`, `forced_install_on`;
- [FIX] regression on *profile* packages not taken in account;

58.5.11 WAPT-1.7.4.6082 (2019-05-20)

(hash 38e08433)

Fixes

— [FIX] **waptexit** not closing if waiting for running tasks but auto upgrade has been disabled ;
— [FIX] events reporting from service's tasks ;

Updated

— [ADD]] new packages attributes : name, valid_from, valid_until, forced_install_on ;
— [ADD] waptexit_disable_upgrade option to **waptexit** to remove the triggering of upgrade from the WAPT Exit utility, but keep the waiting for pending and running tasks ;
— [IMP] added running_tasks key in the WAPT service heckupgrades.json. Was not reflecting an up to date state.
— [IMP] waptself :

— early support of high DPI ;
— loading of icons in the background ;

58.5.12 WAPT-1.7.4.6078 (2019-05-17)

(hash 5b6851ae)

Fixes

— [FIX] takes *profile* packages (AD based groups) into account (**Enterprise** only)

58.5.13 WAPT-1.7.4.6077 (2019-05-15)

(hash 4be40c534c4627)

Fixes

— [FIX] Fixed regression on the WAPT Deployment utility unable to read current waptversion from registry ;
— [FIX] be more tolerant to broken or inexistent *wmi* layer (for the WAPT Console on **wine** for example) ;

58.5.14 WAPT-1.7.4.6074 (2019-05-09)

(hash 95a146c002)

Fixes and improvements over RC2

— [IMP] **waptself.exe** preview application updated. Loads icons in the background.

Known issues :
— does not work with repositories behind proxies and client side authentication ;
— WAPT https Server certificate is not checked when downloading icons) ;
— High DPI not handled properly ;
— Cosmetic and ergonomic improvements still to come ;

— [IMP] Improved the WAPT Server setup on windows : opened port 80 on firewall in addition to 443 ;

— [IMP] Improved the WAPT Server on Debian : added *www-data* group to *wapt* user even if user *wapt* already exists ;
— [IMP] Improved the WAPT Server on CentOS : added *waptwua* directory to SELinux `httpd_sys_content_t` context ;
— [FIX] Improved the WAPT Server client authentication : commented out `ssl_client_certificate` and `ssl_verify_client` ;

By default because old client's certificate does not have proper `clientAuth` attribute (error http 400) ;
— [FIX] problem accessing to 32bit uninstall registry view from 32bit wapt on Windows server 2003 x64 and Windows server 2008 x64 :
it looks like it is not advisable to try to access the virtual Wow6432Node virtual node with disabled redirection ;
— [FIX] Fixed *SetupHelpers* `installed_softwares` regular expression search on name ;
https://github.com/tranquilit/WAPT/issues/7
— [IMP] Improved the WAPT service : for planned periodic upgrade, use single WaptUpgrade task like the one used in websocket ;
— [IMP] Improved the WAPT Exit utility : cancel all tasks if closing the form ;
— [FIX] wapt-get : wapt-get service mode with events : refactor using uWAPTPollThreads ;
— [FIX] **`veyon`** cli executable name updated ;
— [IMP] wapt-get : check *CN* and *subjectAltNames* in lowercase for **`enable-check-certificate`** action ;

58.5.15 WAPT-1.7.4 RC2 (2019-04-30)

(hash 5ef3487)

Security

— upgrade **`urllib3`** to 1.24.2 for CVE-2019-11324 (high severity) ;
— upgrade **`jinja2`** to 2.10.1 for CVE-2019-10906 ;

New

— [NEW] Wapt self service application preview ;

Improvements

— [IMP] propose to copy the newly created CA certificate to ssl local service dir, and restart the WAPT service. Useful for first time use ;

Fixes

— [FIX] `sign_needed` for wapt-signpackages.py ;
— [FIX] missing *StoreDownload* table create ;
— [FIX] bug in fallback `package_uuid` calculation. It didn't include the version ;

58.5.16 WAPT-1.7.4 RC1 (2019-04-16)

(hash 4cdcaa06c83b)

Changes

- [UPD] handling of *subjectAltName* attribute for the WAPT https Server certificates checks in the WAPT Console (useful when certificate is a multi hostname commercial certificate). Before, only CN was checked against host's name ;
- [UPD] client certificate authentication for the WAPT Console ;
- [UPD] versioning of wapt includes now the Git revision count ;

Details

- [FIX] replace openssl command line call with waptcrypto call to create tls certificate on linux server WAPT install ;
- [FIX] Added dnsname *subjectAltName* extension to self signed certificate of the WAPT Server on linux wapt nginx server configuration ;
- [FIX] pkcs12 export ;
- [NEW] handling of *SubjectAlternativeName* in certificates for the WAPT Server X509 certificate check in addition to CN :

Added a *SubjectAltName* when creating self signed certificate on linux wapt nginx server in postconf ;

For old installation, the certificate is not updated. It should be done manually ;

- [FIX] fix **check_install** returning additional packages to install which are already installed (when private repository is using `locale` or `maturities`) :

Added missing attributes in waptdb.installed_matching ;

- [NEW] added client certificate path and client private key path for the WAPT Console access to client side ssl authentication protected servers ;
- [FIX] fix regression with **wapt-get edit <package>** :

made `filter_on_host_cap` a global property of Wapt class instead of a function parameter ;

- [FIX] regression if there are spaces in OU name. The WAPT Console was stripping space for https://roundup.tranquil.it/wapt/issue911and https://roundup.tranquil.it/wapt/issue908 ;
- [IMP] allow "0"..."9", "A"..."Z", "a"..."z", "-","_","=","~","." in package names for OU packages. Replaces space with ~ in package names and "," with "_" ;
- [IMP] make sure we have a proper package name in packages edit dialogs ;
- [IMP] Improved the WAPT service config : allow `waptupdate_task_period` to be empty in `wapt-get.ini` to disable it in the WAPT service ;
- [FIX] waptutils : fix regression on wget() if *user-agent* is overridden ;
- [FIX] waptwua : fix an error in install progress % reporting for wua updates ;
- [IMP] Refactored the WAPT System Tray utility for consistency. Makes use of *uwaptpollthreads* classes ;
- [IMP] Improved the WAPT Exit utility : some changes to try to fix cases when it does not close automatically ;
- [IMP] build : add git Revcount (commit count) to exe metadata.
- [FIX] Fixed the WAPT Console : hosts for package grid not refreshed if not focused.
- [FIX] internal : use synapse httpsend for the WAPT Exit utility / wapt-get / the WAPT System Tray utility local service http queries to workaround authentication retry problems with **indy**.
- [ADD] **wapt-get.exe** : added `--locales` to override temporarily locales form `wapt-get.ini`.
- [ADD] Added `WaptServiceUser` and `WaptServicePassword` / `WaptServicePassword64` command line parameters in **wapt-get.exe**.
- [FIX] Fixed timeout checking in `checkopenport`.
- [ADD] core : Added logs for WAPT Self-service authentication.
- [ADD] Added to the WAPT service : `keywords.json` service action.
- [ADD] Added to the WAPT service : filter keywords (*.csv*) on `packages.json` provider.

— [IMP] Improved the WAPT Console : replace tri-state checkbox by a radio group for wua enabled setting in the *Create teh WAPT Agent* dialog.
— [IMP] Improved the WAPT service local webservice : temporary workaround to avoid costly icons retrieval in local service.
— [FIX] Simplified `installed_wapt_version` in waptupgrade package to avoid potential install issues.
— [IMP] Improved the WAPT Console layout : anchors for running task memo.
— [FIX] Makefullyvisible for main form to avoid forms outside the visible area when disconnecting a second display.
— [FIX] Fixed layout of tasks panel for Windows 10.
— [FIX] Added `token_lifetime` to the WAPT Server side (instead of using clockskew for token duration).
— [UPD] Updated default unit **days** instead of **minutes** for wua scan download install and install_delay.
— [ADD] Added optional export of key and certificate as `PKCS12` file in *create key* dialog. (to check SSL client authentication in browsers…).
— [FIX] Fixed **`winsetup.py`** for backslashes in **`nginx`**.
— [FIX] Fixed **`wapt-get`** json output / flush error.
— [IMP] Improved the cache `host_certificate_fingerprint` and issuer id in local database so that we do not need to read private directory to get `host_capabilities`. It allows to use **`wapt-get list-upgrade`** as normal user.
— [UPD] Do not make DNS query in the WAPT Console Login / waptconfig to avoid DNS timeout if domain DNS server is not reachable.
— [FIX] Fixed warning message introduced in previous revision when adding a new ini config on login (**Enterprise** only).
— [FIX] Fixed waptwua to handle redirect for wsusscn2 head request (**Enterprise** only).
— [UPD] Report only 3 members on the `wapt_version` capability attribute.
— [IMP] Improved WAPT core : refactor WaptUpgrade task : check task to append and then append them to tasks queue in WaptUpgrade.run instead of doing it in caller code. Avoid timeout when upgrading ;
— [IMP] Improved WAPT core : self service rules refactoring ;
— [IMP] Improved WAPT core : notify the WAPT Server when audit on waptupgrade ;
— [IMP] Improved WAPT core : fix `update_status` not working when old packages have no `persistent_dir` in the database ;
— [IMP] Improved core : tasks, events action in the WAPT service : timeout in milliseconds instead of seconds for consistency ;

58.5.17 WAPT-1.7.3.11 (2019-03-25)

(hash 92ccb177d5c)
— [FIX] Fixed the WAPT Console : use repo specific ca bundle to check remote WAPT repo Server certificate (different from main wapt repo) ;
— [FIX] Fixed the WAPT Console / hosts for packages : fixed `F5` to do a local refresh ;
— [FIX] Improved update performance with repositories with a lot of packages ;
— [FIX] Improved the WAPT System Tray utility reporting :
  — fix faulty inverted logic for `notify_user` parameter ;
— [FIX] Fixed the WAPT Console : bad filtering of hosts for package (**Enterprise** only) ;
— [FIX] Fixed the WAPT Exit utility to close even if Running task if no pending task / no pending updates ;
— [FIX] Fixed the WAPT Exit utility : fixed potential case where the WAPT Exit utility remains running with high cpu load ;
— [FIX] Fixed the WAPT Console : fixed HostsForPackage grid not filtered properly (was unproperly using Search expr from first page) ;
— [FIX] Fixed the WAPT service : None has no `check_install_is_running` error on startup of the WAPT service ;
— [FIX] Fixed WAPT core : set `persistent_dir` and `persistent_source_dir` attributes on setup module for install_wapt ;
— [FIX] Fixed WAPT core : fixed bug in guessed `persistent_dir` for dev mode ;
— [FIX] Fixed WAPT core : fixed error resetting status of stuck processes in local database (check_install_running) ;
— [FIX] Fixed the WAPT service : trap error setting runstatus in database in tasks manager loop :
  — Do not send runstatus to the WAPT Server each time it is set ;
— [UPD] Updated WAPT core : define explicitly the `private_dir` of Wapt object ;
— [UPD] WAPT Server : do not refuse to provide authtoken if FQDN has changed (this does not introduce specific risk as request is signed against UUID) ;

— [UPD] Updated WAPT core : if `package_uuid` attribute is not set in package's `control` (old wapt), it is set to a reproductible hash when package is appended to local waptdb so we can use it to lookup packages faster (dict) ;
— [NEW] New in the WAPT Console : added audit scheduling setup in the WAPT Agent dialog (**Enterprise** only) :
  — added `set_waptaudit_task_period` in innosetup installers ;
— [IMP] Improved *SetupHelpers* : add win32_displays to default wmi keys for report ;
— [IMP] Improved WAPT Server setup : create X509 certificate / RSA key for hosts ssl certificate signing and authentication during setup of the WAPT Server ;
— [IMP] Improved the WAPT Exit utility : added sizeable border and icons ;
— [IMP] Improved showing the progress of long tasks ;
— [IMP] Improved the WAPT service : process update of WAPT packages as a task instead of waiting for its completion when upgrading (to avoid timeout when running upgrade the WAPT service task) :
  — added `update_packages` optional (default `True`) parameter for upgrade the WAPT service action ;
— [NEW] Added audit scheduling setup in the WAPT Agent compilation dialog (**Enterprise** only) ;
— [NEW] New in *SetupHelpers* : added `setuphelpers.get_local_profiles` ;
— [IMP] Improved the WAPT Server : do not refuse to provide authentication token for websockets authentication if FQDN has changed ;
— [IMP] Flush *stdout* before sending status to the WAPT Server ;
— [IMP] Improved waptcrypto handling alternative object names in CSR build ;
— [IMP] Improved wapt-get : `--force` option on **`wapt-get.exe`** service mode ;
— [NEW] Use client side authentication for waptwua too ;
— [CHANGE] WAPT Server setup : nginx windows config : relocate logs and pid ;
— [ADD] Added conditional client side ssl authentication in nginx config ;
— [CHANGE] In the WAPT Console : refactored wget, wgets for the WaptRemoteRepo and the WAPT Server to use requests.Session object to handle specific ssl client authentication and proxies :
Be sure to set privateKey password dialog callback to decrypt client side ssl authentication key ;
— [IMP] Improved waptcrypto : added waptcrypto.is_pem_key_encrypted ;
— [IMP] Improved the WAPT Console : make sure the WAPT Agent window is fully visible ;
— [IMP] Improved the WAPT Console : make sure Right click select row on all grids ;
— [ADD] Aded in the WAPT Console : import from remote repo : add certificate and key for client side authentication ;

58.5.18 WAPT-1.7.3.10 (2019-03-06)

(hash ec8aa25ef)

Security

— [UPD] upgraded **`OpenSSL`** dlls to 1.0.2r for https://www.cert.ssi.gouv.fr/avis/CERTFR-2019-AVI-080/ (moderate risk) ;

New

— [IMP] much reworked wizard pages embedded in **`waptserversetup.exe`** windows server installer. Install of the WAPT Server on Windows is easy again :

— register server as a client of the WAPT Server ;
— create new key / certificate pair ;
— build **`waptagent.exe`** and **`waptupgrade.exe`** package ;
— configure package prefix ;

— [NEW] if client certificate signing is enabled on the WAPT Server (`waptserver.ini` config), the WAPT Server will sign a CSR for the client when the client is first registered. See *Configuration de l'authentification par certificat côté client* .

— [NEW] wapt-get : added new command `create-keycert` to create a pair of RSA key / x509 certificate in batch mode. Self signed or signed with a CA key/certificate :

(options are case sensitive…)

— option `/CommonName` : CN to embed in certificate ;
— options `/Email`, `/Country`, `/Locality`, `/Organization`, `/OrgUnit` : additional attributes to embed in certificate ;
— option `/PrivateKeyPassword` : specify the password for private key in clear text form ;
— option `/PrivateKeyPassword64` : specify the password for private key in base64 encoding form ;
— option `/NoPrivateKeyPassword` : ask to create or use an unencrypted RSA private key ;
— option `/CA` = `True` (or False)) : create a certification authority certificate if True (default to True) ;
— option `/CodeSigning` = `True` (or False)) : create a code signing certificate if True (default to True) ;
— option `/ClientAuth` = `True` (or False) : create a certificate for authenticating a client on the WAPT https Server with ssl authentication. (default to True) ;
— option `/CAKeyFilename` : path to CA private key to use for signing the new certificate (defaults to `%LOCALAPPDATA%waptconsolewaptconsole.ini [global] default_ca_key_path` setting) ;
— option `/CACertFilename` : path to CA certificate to use for signing the new certificate (defaults to `%LOCALAPPDATA%waptconsolewaptconsole.ini [global] default_ca_cert_path` setting) ;
— option `/CAKeyPassword` : specify the password for CA private key in clear text form to use for signing the new certificate (no default) ;
— option `/CAKeyPassword64` : specify the password for CA private key in base64 encoding form to use for signing the new certificate (no default) ;
— option `/NoCAKeyPassword` : specify that the CA private to use for signing the new certificate is unencrypted ;
— option `/EnrollNewCert` : copy the newly created certificate in `<wapt>ssl` to be taken in account as an authorized packages signer certificate ;
— option `/SetAsDefaultPersonalCert` : set `personal_certificate_path` in configuration inifile `[global]` section (default `%LOCALAPPDATA%waptconsolewaptconsole.ini`) ;

— [NEW] wapt-get : added new commands **`build-waptagent`** to compile a customized WAPT Agent in batch mode :

— copy **`waptagent.exe`** and pre-waptupgrade locally (if not `/DeployWaptAgentLocally`, upload to the WAPT Server with https) ;
— option `/DeployWaptAgentLocally` : copy the newly built **`waptagent.exe`** and prefix-waptupgrade_xxx.wapt to local WAPT Server repository directory `.\waptserver\repository\wapt\` ;

— [NEW] **`wapt-get register`** : added options for easy configuration of wapt when registering :

— `--pin-server-cert` : pin the WAPT Server certificate. (check that CN of certificate matches hostname of WAPT Server and WAPT repo) ;
— `--wapt-server-url` : set `wapt_server` setting in `wapt-get.ini` ;
— `--wapt-repo-url` : set `repo_url` setting in `wapt-get.ini`. (if not provided, and there is not `repo_url` set in `wapt-get.ini`, extrapolate `repo_url` from the WAPT Server url) ;

— [NEW] wapt-get : added check-valid-codesigning-cert / CheckPersonalCertificateIsCodeSigning action ;

Improvements and fixes

— python libraries updates

— **`cryptography from 2.3.1`** –> **`cryptography 2.5.0`** ;
— **`pyOpenSSL 18.0.0`** –> **`pyOpenSSL 19.0.0`** ;

— [FIX] Do not reset host.server_uuid in the WAPT Server database when host disconnect from websocket. Set `host.server_uuid` in the WAPT Server database when host gets a token ;
— [FIX] modify isAdminLoggedIn to try to fix cases when we are admin but function return false ;
— [FIX] ensure valid package name in package wizard (issue959) ;

— [FIX] regression when using python cryptography 2.4.2 openssl bindings for windows XP WAPT Agent (openssl bindings of the python cryptopgraphy default WHL >= 2.5 does not work on Windows XP);
— [FIX] trap exception when creating database tables from scratch fails, allowing upgrade of structure;
— [FIX] reduce the risk of *database is locked* error;
— [FIX] deprecation warning for verifier and signer when checking crl signature;
— [FIX] `persistent_dir` calculation in package's call_setup_hook when package_uuid is None in local wapt database (for clients migrated from pre 1.7 wapt, error None has no len() in audit log);
— [FIX] regression : do not try to use host_certificate / key for client side ssl authentication if they are not accessible;
— [IMP] define proxies for crl download in **`wapt-get scan-packages`**;
— [IMP] fixed bad normalization action icon;
— [IMP] paste from clipboard action available in most packages editing grid;
— [IMP] propose to define package root dev path, package prefix, the WAPT Agent or new private key / certificate when launching the WAPT Console;
— [IMP] remove the need to define waptdev directory when editing *groups* / *profiles* / *wua packages* / *self-service* packages;
— [IMP] grid columns translations in French;
— [IMP] Improved the WAPT Exit utility responsiveness improvements. Events check thread and tasks check thread are now separated.
— [NEW] added ClientAuth checkbox when building certificate in the WAPT Console;
— [NEW] added `--quiet -q` option to `postconf.py`
— [MISC] add an example of client side certificate authentication
— [ADD] added clientAuth extended usage to x509 certificates (default True) for https client authentication using personal certificate;
— [NEW] use of ssl client certificate and key in the WAPT Console for authenticating with the WAPT Server;
— [FIX] ssl client certificate authentication not taken in account for the WAPT Server api and host repository;
— [ADD] added `is_client_auth` property for certificates;
  — default *None* for `is_client_auth` certificate / CSR build;
  — do not fallback to host's client certificate authentication if it is not clientAuth capable (if so, http error 400);
— [MISC] waptcrypto : added SSLPKCS12 to encapsulate pcks#12 key / certificate in certificate store;
— [MISC] added splitter for log memo in Packages for hosts panel;
— [FIX] store fixes;
— [FIX] be tolerant when no `persistent_dir` in *waptwua* packages;

— min wapt version 1.7.3 for self service packages and *waptwua* packages,

— [FIX] WsusUpdates has no attribute `downloaded`;

58.5.19 WAPT-1.7.3.7 (2019-02-19)

(hash 373f7d92)

Bug fixes

— [FIX]] softs normalization dialog closed when typing F key (**Enterprise** only);
— [IMP] include waptwua in the WAPT **`Nginx`** Server windows locations (**Enterprise** only);
— [FIX] force option from service or websockets not being taken in account in **`install_msi_if_needed`** or **`install_exe_if_needed`**;
— [IMP] improved win updates reporting (uninstall behavior) (**Enterprise** only);
— [ADD] added uninstall action for winupdates in the WAPT Console (**Enterprise** only);
— [FIX] reporting from dmi « size type » fields with non integer content (**Enterprise** only);

Improvements

— [IMP] Improved the WAPT Exit utility : allow minimize button ;
— [IMP] Improved the WAPT Exit utility : layout changes ;
— [IMP] AD authentication : less restrictive on user name sanity check (**Enterprise** only) ;
— [IMP] handling of updates of data for winupdates with additional download urls (**Enterprise** only) ;
— [ADD] added some additional info fields to WsusUpdates table (**Enterprise** only) ;
— [ADD] added filename to Packages table for reporting and store usage (**Enterprise** only) ;
— [ADD] added uninstall win updates to the WAPT Console (**Enterprise** only) ;
— [ADD] added windows updates uninstall task capabilities (**Enterprise** only) ;
— [ADD] added filename to Packages table ;
— [IMP] increased default clockskew tolerance for client socket io ;

58.5.20 WAPT-1.7.3.5 (2019-02-13)

Bug fixes

— [FIX] regression in package filenames (missing _) ;
— [FIX] Fixed mismatch for the WAPT Console `[global] waptwua_enabled` setting ;
— [FIX] Fixed default in the WAPT Console *EnableWaptWUAFeatures* to True ;

58.5.21 WAPT-1.7.3.4 (2019-02-13)

Bug fixes

— [FIX] Fixed the WAPT Exit utility : install of and empty list of Windows Updates (**Enterprise** only) ;
— [FIX] wapt-get.exe WaptWUA commands : fixed import of waptwua client module for waptwua-scan download install (**Enterprise** only) ;
— [FIX] `install_delay` for Windows Updates stored as a time_delta in waptdb (**Enterprise** only) ;

Improvements

— [ADD] versioning on group packages filenames ;
— [ADD] button to create AD Host profiles (package automatically installed/removed based on AD Grouo memberships)
— [IMP] reduce the WAPT System Tray utility notifications occurrences. `notify_user` = `False` per default
— [FIX] Fixed the WAPT Exit utility : details panel does not show the pending packages to install ;
— [FIX] always install the missing dependencies in install (even if upgrade action should have queued dependencies installs before) for some corner known cases ;
— [FIX] get the WAPT Server certificate chain popup action when building the WAPT Agent ;
— [ADD] action to create a key / certificate in the WAPT Console conf ;
— [IMP] hide inactive / disabled WaptWUA actions in Host popup menu ;
— [ADD] checkbox to display newest only for groups ;
— [ADD] Added in the WAPT Console the config parameter `licences_directory` to specify the location (directory) of licenses (**Enterprise** only) ;
— [IMP] Improved the WAPT Agent build dialog : Removed the *Append host's profiles* option ;
— [IMP] remove waptenterprise directory if waptsetup community is deployed over a waptenterprise edition ;

58.5.22 WAPT-1.7.3.3 (2019-02-11)

— [IMP] Core :

— better support for `locales`, `maturities` and `architecture` packages filtering ;

— [NEW] Self service rule packages (**Enterprise** only) :

— Package to define which packages can be installed / remove for groups of users ;
— WAPT Windows Updates rules packages (**Enterprise** only) ;

— [NEW] package to define which Windows Updates are allowed / forbidden to be deployed by Wapt WUA Agents ;
— **WAPT Agent** build :

— [ADD] Added the option for `use_fqdn_as_uuid` when building **`waptagent.exe`** ;
— [ADD] Added the option to define the profile package to be deployed upon WAPT install on hosts ;
— [ADD] Added the options to enable WaptWUA (Windows updates with Wapt) (**Enterprise** only) ;

— Host Profile packages (**Enterprise** only) :

— [IMP] specific packages (like Group packages) which are installed or removed depending of `wapt-get.ini [global] host_profiles` ini key ;
— [NEW] if a *profile* package name matches Computer's AD Groups, it is deployed automatically ;

— Reporting (**Enterprise** only) :

— [NEW] import / export queries as json files ;
— [IMP] softwares names normalization as a separate dialog ;

— **WAPT Exit utility** :

— [IMP] reworked to make it more robust ;
— [IMP] takes in account packages to remove ;
— [IMP] takes in account Wapt WUA Updates (**Enterprise** only) :
— command line switch : /install_wua_updates ;
— wapt-get.ini setting : [waptwua] `install_at_shutdown` = `True` ;
— checkbox in the WAPT Exit utility to skip install of Windows Updates ;

— **WAPT Console** Custom commands :

— [NEW] ability to define custom popupmenu commands which are launched for the selection of hosts. Custom variables {uid} ;

— Other improvements :

— [IMP] French translations fixes ;

58.5.23 Changelog 1.7.2

— [NEW] Reporting (**Enterprise** only) :

— basic SQL reporting capability ;
— duplicate action / copy paste for reporting queries ;

— [ADD] *SetupHelpers* : added *SetupHelpers* `processes_for_file` and `get_computer_domain` ;

Libraries updates

- **python 2.7.15** on Windows ;
- **openssl-1.0.2p** ;
- upgraded to **python-requests 2.20.0** (Security Fix) ;

Improvements

- [IMP] Do not refresh GridHostsForPackage if not needed (**Enterprise** only) ;
- [IMP] Do not add a newline to log text output for LogOutput ;
- [IMP] Improved handling of update_host_data hashes to reduce amount of data sent to the WAPT Server on each **update_server_status** ;
- [IMP] Set python27.dll path in wapt-get and **waptconsole.exe** (fix cases with multiple python installations) ;
- [FIX] Removal of packages when upgrading host via websockets ;
- [IMP] Do not get host capabilities if not needed when updating ;
- [IMP] Do not check package control signatures in wapt-get when loading list of packages for development tasks ;
- [IMP] Moved static WAPT Server assets to a /static root split base.html and index.html templates for blueprints ;
- [FIX] Fixed selective pending wua install or downloads (**Enterprise** only) ;
- [FIX] Fixed WUA updates filter logic (**Enterprise** only) ;
- [IMP] Improved uninstall *host* packages if `use_hostpackages` is set to false :
  - add a forced update in the task loop when host capabilities have been changed ;
  - include `use_host_packages` and `host_profiles` in host's capabilities.
- [FIX] Fixed regression not removing implicit packages.
- [IMP] More tolerant to unicode errors in **update_host_data** to avoid hiding actual exception behind an encoding exception.
- [FIX] Fixed order of columns not kept when exporting reports (**Enterprise** only)
- [IMP] Improved `install_msi_if_needed`, `install_exe_if_needed` : check if `killbefore` is not empty or None
- [IMP] Changed tasks's progress and runstatus to property
- [FIX] Fixed audit aborted due to exception : "NoneType" object is not iterable (**Enterprise** only)
- [ADD] *SetupHelpers* : Added `setuphelpers.get_app_path` and `setuphelpers.get_app_install_location` :
  - add fix_wmi procedure to re-register WMI on broken hosts ;
  - some wmi fallbacks to avoid unregistered hosts when WMI is broken on them.
- [ADD] Added online wua scans (**Enterprise** only)
- [ADD] Added random `package_uuid` when signing a package metadata which could be used later as a primary key :
  - creates a random `package_uuid` when installing in DEV mode ;
  - creates a random `package_uuid` when installing a package without `package_uuid`.
- [IMP] Moved and renamed `EnsureWUAUServRunning` to *SetupHelpers* ;
- [ADD] Added `pending_reboot_reasons` to inventory ;
- [IMP] Improved the display of WAPT package versions for missing packages ;
- [ADD] **wapt-get sign-packages** : added setting `maturity` and inc version in sign-packages action ;
- [ADD] Added *WindowsUpdates's host History* grid below *WindowsUpdate* grid (**Enterprise** only) ;
- [IMP] Improved storing of Host Windows update history in the WAPT Server database (**Enterprise** only) ;
- [IMP] keep selected or focused rows in grids ;
- [IMP] Improved updates Packages table when uploading a Package / Group. This table is meant mainly for reporting purpose ;
- [IMP] Disables indexes for some BinaryJson fields ;
- [FIX] Fixed Windows Updates `install_date` reporting (**Enterprise** only) ;
- [ADD] Added a checkbox to enable `use_fqdn_as_uuid` when building **waptagent.exe** ;
- [IMP] Changed default value for `upgrade_only_if_not_process_running` ;
- [IMP] Changed naming of organizational *unit* packages to remove ambiguity with comma in package name and comma to describe the list of WAPT packages `depends` / `conflicts` :
  - Replace "," with "_" when editing package (**Enterprise** only) ;
- [ADD] Added to the WAPT Exit utility : priorities and `only_if_not_process_running` command line switches ;

— [IMP] Improved waptupgrade : changed `windows_version` and Version ;
— [ADD] Added *SetupHelpers* `setuphelpers.windows_version` : added `setuphelpers.members_count` ;
— [IMP] Improved waptutils.Version : strip members to `members_count` if not *None* ;
— [ADD] Added control attributes editor keywords license homepage `package_uuid` to the local WAPT service database ;
— [ADD] Added short fingerprint to repr of SSLCertificate ;
— [IMP] Be sure password gui is visible even if parent window is not ;
— [ADD] Added gui for private key password dialog if `--use-gui` ;
— [ADD] Added `--use-gui` to **`wapt-get.exe`** command line argument to force the use of waptguihelper for the WAPT Server credentials when registering ;

58.5.24 WAPT-1.6.2.7 (2018-10-02)

This is a bugfix release for 1.6.2.5 :
— [FIX] Fixed the WAPT Exit utility : changed the default value of `upgrade_only_if_not_process_running` parameter to *False* instead of *True* :
if `upgrade_only_if_not_process_running` is *True*, the install tasks for packages with running processes (*impacted_process*) are skipped ;

if `upgrade_only_if_not_process_running` is *False*, the install tasks for packages with running processes may impact the user if the installer kills the running processes ;
— [FIX] *waptwua* : take in account Windows Updates *RevisionNumber* attribute to identify uniquely an Update in addition to UpdateID field (**Enterprise** only). This fixes the 404 error when downloading missing windows updates on a client.

58.5.25 WAPT-1.6.2.6 (2018-09-26)

This is a bugfix release for 1.6.2.5 :
— [FIX] Fixed the WAPT Server Enterprise on Windows : added proper upgrade path from **`PostgreSQL 9.4`** (used in WAPT 1.5) to **`PostgreSQL 9.6`** which is required for WAPT-Windows Update :
— new database binary and data directory path are suffixed with -9.6 ;
— old data is suffixed with -old after migration ;
— [FIX] upgrade script for **`MongoDB`** upgrade (WAPT 1.3) to **`PostgreSQL`** used since WAPT 1.5 ;
— [FIX] regression on WMI / DMI inventory which may be not properly sent back to the WAPT Server ;

58.5.26 WAPT-1.6.2.5 (2018-09-14)

[NEW] Main new features if you are coming from 1.5 :
— per package *Audit* feature (**Enterprise** only) ;
— *WAPT managed Windows Updates* tech preview (**Enterprise** only) ;
— wizards to guide post configuration of Windows server and first use of **`waptconsole`** ;
— **`waptconsole/`** private repo page : added a grid which shows the computers where the selected package is installed ;
It includes numerous changes over the 1.5.1.26 version.

New

— [NEW] per package audit feature :

— def audit() hook function to add into package's `setup.py`. By default, check *uninstall key* presence in registry :
— **`wapt-get audit`**;
— **`wapt-get -S audit`**;
— **`wapt-get audit <packagename>`**;
— right click in the WAPT Console on hosts or installed packages/ Audit package ;
— synthetic audit status for each host ;
— for each installed package : *last_audit_status*, *last_audit_on*, *last_audit_output*, *next_audit_on* ;
— scheduled globally with `wapt-get.ini` parameter `[global]` :

```
```

waptaudit_task_period = 4h or in package's `control` file :

```
```

audit_schedule = 1d
— audit log displayed in **`waptconsole`** below installed package grid if *Audit Status* column is focused ;

— [UPD] updated python modules
— [IMP] build with **`Lazarus 1.8.2`** instead of **`CodeTyphon 2.8`** for the Windows executables :
— better strings encoding handling and easier to setup for the development ;

Known issues

— **`PostgreSQL 9.6`** is required for WAPT WUA tech preview (Debian Jessie not supported) ;
— WAPT 1.6 includes one more security layer in the WAPT Agent to WAPT Server connection. After the WAPT Server upgrade, the client desktops will not be able to connect to the WAPT Server as long as they have not been upgraded themselves. If you require to be able to remotely manage the WAPT agent while the agent has not yet been upgraded, it is necessary to set `allow_unauthenticated_connect` to *True* in `waptserver.ini` ;

Fixes

— [FIX] add AD Groups as Hosts dependencies in **`waptconsole`** ;
— [FIX] remove image on reachable column if no status has been sent yet ;
— [FIX] Organizational Units WAPT packages not being installed when there are spaces in DN ;
— [FIX] Operational error when host are trying to reconnect but are not registered ;
— [FIX] fill in *created_on* database fields on win updates data ;
— [IMP] debian server postinst : remove old `pyc` files ;

Changes

— [IMP] Improved WAPT Console setup Wizard ;
— [ADD] *allow_unauthenticated_connect* defaults to *allow_unauthenticated_registration* if it is not explicitly set in `waptserver.ini` file (This will ease migration from 1.5 to 1.6) ;
— [IMP] `Escape` key on password edit of login moves focus to configuration combo ;
— [IMP] PackageEntry.asrequirement() : removed space between package name and version specification ;
— [IMP] missing *install_date* in *insert_many* for some updates ;
— [ADD] add force argument for WAPTUpdateServerStatus action ;
— [IMP] Do not includes `setup.py` in initial host's packages inventory, and full inventory ;
— [IMP] allow to use installed **`waptdeploy.exe`** without retry/ignore dialog ;
— [IMP] be sure error is reported properly in **`socketio`** ;
— [IMP] added *package_uuid* and homepage package attributes ;
— [IMP] added installed on columns for host wsus updates ;
— [FIX] WUA grid layout saving ;

58.5.27 WAPT-1.6.2.2 (2018-07-16)

Known issues

— **`PostgreSQL 9.6`** is required for WAPT WUA tech preview (Debian Jessie not supported) ;
— the authentication of client connections to the WAPT websockets server is not compatible with pre-1.6.2 wapt clients. During migration, if you want to keep the connection with clients, you have to disable the authentication with the parameter : `allow_unauthenticated_connect = False` in the WAPT Server's configuration file `waptserver.ini`. When all clients have migrated, this can be removed ;

New

— [NEW] wizard for the initial configuration of **`waptserver`** on Windows ;
— [ADD] wizard for the initial configuration of **`waptconsole`** connection parameters ;
— [ADD] **Enterprise only** : waptconsole/ private repo page : added a grid which shows the computers where the selected package is installed ;
— [NEW] **Enterprise only** : WAPT WUA Windows Updates management technical preview :

— activate with `waptwua_enabled = True` in `wapt-get.ini` file on the client ;
— scan of updates on Windows clients with the IUpdateSearcher Windows API and the `wsusscan2` cab file from Microsoft ;
— additional page in the WAPT Console host inventory for Windows updates status reported (HostWsus model) ;
— additional page in the WAPT Console for the consolidated view of all updates reported by hosts (WsusUpdates model) ;
— periodic task on the WAPT Server to check and download newer version of `wsusscan2` cab file from Microsoft (daemon/ service wapttasks) ;
— periodic Task on the WAPT Server to download missing windows updates files as reported by Windows client after scan :

— missing files are downloaded if one of the client should install it and has not yet a copy in its local windows update cache ;
— downloads are logged in *WsusDownloadTasks* model ;

Changes

— [ADD] field in hosts table to keep the hashes of sent host data, so that clients can send only what needs to be updated ;
— [ADD] `db_port` WAPT Serverconfig parameter if **`posgresql`** server is not running on standard port 5432 ;
— [ADD] editor optional attribute for package control, used in *register_windows_uninstall* helper if supplied ;
— [IMP] websocket authentication with a timestamped token obtained from the WAPT Server with client SSL certificate on the WAPT Server with client SSL certificate ;
— [IMP] json responses from **`waptserver`** are gzipped ;

Fixes

— [IMP] forced host uuid ;
— [IMP] forced computer AD Organizational unit ;
— [IMP] public certs dir ;
— [FIX] caching of negative result for certs chain validation ;
— [IMP] refactoring of the WAPT Server python modules (*config*, *utils*, *auth*, *app*, *common*, *decorators*, *model*, *server*) for the enterprise modularity ;
— [FIX] timezone file timestamp handling for http download ;

Python modules updates

— upgrade to **`peewee 3.4`** ;
— upgrade to **`eventlet==0.23.0`** ;
— upgrade to **`huey 1.9.1`** ;
— **`eventlet 0.20.1`** –> **`eventlet 0.22.1`** ;

0.22.1 :
— [IMP] event : Event.wait() timeout=None argument to be compatible with upstream CPython ;
— [IMP] greendns : Treat /etc/hosts entries case-insensitive. Thanks to Ralf Haferkamp ;
0.22.0 :
— [IMP] dns : reading /etc/hosts raised DeprecationWarning for universal lines on Python 3.4+. Thanks to Chris Kerr ;
— [IMP] green.openssl : Drop OpenSSL.rand support. Thanks to Haikel Guemar ;
— [IMP] green.subprocess : keep CalledProcessError identity. Thanks to Linbing@github ;
— [IMP] greendns : be explicit about expecting bytes from sock.recv. Thanks to Matt Bennett ;
— [IMP] greendns : early socket.timeout was breaking IO retry loops ;
— [IMP] GreenSocket.accept does not notify_open. Thanks to orishoshan ;
— [IMP] patcher : set locked RLocks" owner only when patching existing locks. Thanks to Quan Tian ;
— [IMP] patcher : workaround for monotonic « no suitable implementation ». Thanks to Geoffrey Thomas ;
— [IMP] queue : empty except was catching too much ;
— [IMP] socket : context manager support. Thanks to Miguel Grinberg ;
— [IMP] support : update **`monotonic 1.3`** (5c0322dc559bf) ;
— [IMP] support : upgrade bundled to **`dnspython 1.16.0`** (22e9de1d7957e) https://github.com/eventlet/eventlet/issues/427 ;
— [FIX] websocket leak when client did not close connection properly. Thanks to Konstantin Enchant ;
— [IMP] websocket : support permessage-deflate extension. Thanks to Costas Christofi and Peter Kovary ;
— [IMP] wsgi : close idle connections (also applies to websockets) ;
— [IMP] wsgi : deprecated options are one step closer to removal ;
— [IMP] wsgi : handle remote connection resets. Thanks to Stefan Nica ;
0.21.0
— [IMP] new timeout error API : .is_timeout=True on exception object. It's now easy to test if network error is transient and retry is appropriate. Please spread the word and invite other libraries to support this interface ;
— [IMP] hubs : use monotonic clock by default (bundled package) ; Thanks to Roman Podoliaka and Victor Stinner

— [IMP] dns : EVENTLET_NO_GREENDNS option is back, green is still default ;
— [IMP] dns : hosts file was consulted after nameservers ;
— [IMP] wsgi : log_output=False was not disabling startup and accepted messages ;
— [IMP] greenio : Fixed OSError : [WinError 10038] Socket operation on nonsocket ;
— [IMP] dns : EAI_NODATA was removed from RFC3493 and FreeBSD ;
— [IMP] green.select : fix mark_as_closed() wrong number of args ;
— [NEW] added zipkin tracing to eventlet ;
— [IMP] db_pool : proxy Connection.set_isolation_level() ;
— **Flask-socketio 2.9.2** –> **Flask-socketio 3.0.1** ;
— **python-engineio 2.0.1** –> **python-engineio 2.0.4** ;
— **python-socketio 1.8.3** –> **python-socketio 1.9.0** ;
— upgrade to **websocket-client 0.47** ;

58.5.28 WAPT-1.6.2.1 (2018-07-04)

New features

— [ADD] def audit() optional hook in package is called periodically to check compliance. Log and status is reported in the WAPT Server database and displayed in the WAPT Console (**Enterprise**).
— [ADD] WSUS tech preview : based on local Windows update engine and `WSUSSCAN2` cab Microsoft file. WAPT Server act as a caching proxy for updates. Scanning for, downloading and applying Windows updates can be triggered from the WAPT Console on workstations (**Enterprise**). A new wapttasks process is launched on the WAPT Server to download updates and wsusscan cab from Internet.

Changes / Improvements

— [IMP] better utf8 handling ;
— [IMP] **wapt-get make-template** from a directory creates a basic installer for portable apps ;
— [IMP] Improved wapt-get, the WAPT Exit utility : Removed ZeroMQ message queue on the client, replaced by simple http long polling to monitor tasks status ;
— [IMP] Improved the WAPT Console : Replaced blocking timer based http polling for tasks status by threaded http long polling ;
— [IMP] Improved the WAPT Console : Filter hosts on whether current personal certificate signature is authorized for remote tasks (**Enterprise**). If the same WAPT Server is used for several organizations, it allows to focus on own hosts. This supposes that different CA certificates are deployed depending on the client host's organization. In this release, the filtering is not enforced and not cryptographically authenticated ;
— [CHANGE] renamed **waptservice.py** to **service.py** and **waptserver.py** to **server.py**, activated absolute import for all python sourced absolute import for all python sources ;
— [REMOVED] *use_http_proxy_for_template* parameter (setting is now in `[wapt-templates]` repo) ;

The WAPT service
— [ADD] handling of WUA tasks (Scan, download, apply updates) (**Enterprise**) ;
— [ADD] handling of auditing tasks ;
The WAPT Server
— [ADD] tasks queue (**Huey**) for the WSUS background tasks (**Enterprise**) ;
— [IMP] gzip compression activated on the **nginx** configuration ;
The WAPT System Tray utility
— [ADD] option in `wapt-get.ini` to hide some items :
— `hidden_wapttray_actions` : comma separated list of :
LaunchWAPTConsole, *register*, *serviceenable*, *reloadconfig*, *cancelrunningtask*, *cancelalltasks*, *showtasks*, *sessionsetup*, *forceregister*, *localinfo*, *configure* ;

— [CHANGE] use long polling instead of **zmq**;
— [IMP] stop/ start/ query the WAPT service using a thread to avoid gui freeze;

Fixes

— [FIX] waptguihelper : be sure to load the proper python27.dll;
— [FIX] core : forward *force* argument from the WAPT Console to `setup.py` install() hook;
— [FIX] overwrite `psproj` package file when editing a package to fix path to WAPT python virtualenv and add new debug actions;

Modules updates

— [UPD] GUI Binaries are built with **Lazarus 1.8.2** / **fpc 3.0.4** instead of **CodeTyphon 2.8**;
— [UPD] **peewee 3.0.4**;
— [UPD] **eventlet 0.23.0**;
— [UPD] **huey 1.9.1**;
— [UPD] **pywin32** rev 223;
— [UPD] **Flask-socketio 2.9.6**;
— [UPD] **engineio.socket 2.0.4**;
— [UPD] **websocket-client 0.47**;
— [UPD] **pyOpenSSL 17.5.0**;
— [UPD] **request 2.19.1**;

Known issues

— *unit* type of packages (with AD DN style names) are not well handled by local WAPT self service, because of commas in name.

58.5.29 WAPT-1.6.1.0 (2018-06-21)

Fixes

— [FIX] Fixed av potential cause in the WAPT System Tray utility;
— [IMP] Improved buffer LogOuput;
— [FIX] Fixed wait task result loop in the WAPT Server;
— [FIX] Fixed bad acl on the WAPT service;
— [FIX] Fixed repo timeout not taken in account;
— [FIX] Fixed bad parameter for `repo_url` and `[wapt-host]` section;
— [FIX] Fixed potential cause for anti-virus flagging the WAPT Exit utility;
— [FIX] Fixed make isAdmin non blocking as a workaround for false positive checks;
— [FIX] Fixed use timeout parameter when importing external package;
— [FIX] Fixed pass timeout parameter when importing;
— [FIX] Fixed bad `repo_url` config naming;
— [FIX] Fixed calc hash when compiling if file does not exist;
— [FIX] Fixed repo timeout is float;
— [FIX] Fixed custom zip corruption when signing a package with non ascii filenames;
— [FIX] Fixed check wapt_db is assigned when rollbacking;
— [IMP] Improved logging in events;
— [FIX] Fixed installed packages section is incorrectly reported as *base* instead of *unit* or *host* in the WAPT Console;

— [IMP] ensure manual service wua is running when using command line ;
— [UPG] Python modules updates :
  — upgrade to **peewee 3.4** ;
  — upgrade to **eventlet==0.23.0** ;
  — upgrade to **huey 1.9.1**.
— [CHANGE] Replaced eventprintinfo with LogOutput ;
— [ADD] Added `waptwua_enabled` config parameter ;
— [IMP] Improved missing `ensure_list` waptwua_enabled config parameter ;
— [IMP] default *waptwua_enabled* to None to avoid wuauserv service configuration change ;
— [ADD] Added missing columns for window updates ;
— [ADD] Added action in the WAPT Console to show help on KB ;
— [IMP] Improved the WAPT System Tray utility cosmetic : hide duplicated separators in tray popup menu when some actions are hidden ;
— [ADD] Added http_proxy ini setting for the WAPT Server external download operations ;
— [IMP] Improved the WAPT System Tray utility : Start and stop the WAPT service using a thread to avoid gui freeze ;
— [IMP] Switched to pure FPC PBKDF2 password hash calc for postconf ;
— [IMP] Refactored WAPT Server code to share app and socketio instances ;
— [FIX] Fixed forward the « force » argument (command line and through the websockets) to the install() setup.py hook ;
— [FIX] Fixed to not display all missed events at tray startup in the WAPT System Tray utility ;
— [FIX] Fixed no default `audit_period` ;
— [REMOVED] **zeromq**, replaced by long http polling between the WAPT System Tray utility, wapt-get and the WAPT service ;

58.5.30 WAPT 1.5.1.26 (2018-07-12)

Bug fixes

— [IMP] revert monkey_patch for the WAPT Server on windows. No reason to exclude thread ;
— [ADD] `allow_unauthenticated_connect` config (default *false*) on the WAPT Server ;
— [FIX] CRITICAL update_host failed UnboundLocalError(« local variable “result” referenced before assignment »,) ;
— [FIX] https://roundup.tranquil.it/wapt/issue951 ;
— [FIX] https://forum.tranquil.it/viewtopic.php?f=13&t=1160ix ;
— [FIX] https://forum.tranquil.it/viewtopic.php?f=13&t=1160 ;
— [FIX] `init_workdir.bat` ;
— [FIX] returns a token when updating host data for websocket authentication ;
— [IMP] rewrite package psproj when editing (to fix wapt basedir paths) ;
— [FIX] %s -> %d format string for expiration warning message ;
— [FIX] host_certificate not found for waptstarter ;
— [ADD] some dev build scripts ;

58.5.31 WAPT-1.5.1.24 (2018-07-04)

Bug fixes

— [FIX] Fixed zipfile python library bug for packages which contains files with non-ascii filenames. Signed WAPT packages were corrupted in this case ;
— [FIX] Fixed deadlocks on the WAPT Server database when simultaneous database connections is larger than 100 (default maximum connections configured by default on postgresql) ;
— [FIX] Fixed crash of the WAPT Console on warning message when license is about to expire (**Enterprise** only) ;
— [FIX] Fixed %s –> %d format string for expiration warning message ;
— [FIX] Fixed `host_certificate` not found for waptstarter ;

— [FIX] Fixed `waptserversetup.iss` to include enterprise modules (**Enterprise**) ;
— [FIX] Fixed download link to waptsetup and the WAPT Deployment utility on the WAPT Server index page for Windows ;

Modules updates

— **`requests 2.19.1`** ;
— **`Rocket 1.2.8`** - Don't try to resurrect connections that timeout. Increase the timeout ... to decrease the likelihood :
  — handle PyPi only supports HTTPS/TLS downloads now ;
  — fix the problem that when body is empty no terminating ; chunk is sent for chunked encoding.
  — avoid sending the terminating chunk in case it is a HEAD request ;
  — fix the problem that when body is empty no terminating chunk is sent for chunked encoding ;
  — explicitly set the log level to warning ;
  — fix bug « Threadpool grows by negative amount when max_threads = 0 » ;
  — do not try to resurrect connections that timeout. Increase the timeout to decrease the likelihood ;

58.5.32 WAPT-1.5.1.23 (2018-03-28)

Changes

— [IMP] Improved the WAPT Exit utility : display a custom PNG logo if one is created in `%WAPT_HOME%\templates\waptexit-logo.png` ;
— [IMP] nssm.exe is signed with Tranquil IT code signing key ;
— [ADD] Added in the WAPT Console : locale and maturity columns in packages status grid ;
— [IMP] Improved in the WAPT Console the WAPT Agent wizard ; be sure to get a relative path when checking certificate validity ;
— [ADD] Added to waptsetup `/CopyPackagesTrustedCA` and `/CopyServersTrustedCA` command line parameters to allow deployment of wapt with specific certificates with GPO for wapt without recompiling waptsetup ;

Example :

```
C:tmpwaptdeploy --hash=e17c4eddd45d34000df0cfe64af594438b0c3e1ee9791812516f116d4f4b9fa9
--minversion=1.5.1.23 --waptsetupurl=http://buildbot/~tisadmin/wapt/latest/waptsetup.exe
--setupargs=/CopyPackagesTrustedCA=c:tmptranquilit.crt --setupargs=/CopyServersTrustedCA=c:tmpsrvwapt.
mydomain.lan.crt --setupargs=/verify_cert=sslserversrvwapt.mydomain.lan.crt --setupargs=/
repo_url=https://srvwapt.mydomain.lan/wapt --setupargs=/waptserver=https://srvwapt.mydomain.lan
--setupargs=/DIR=c:wapt
```

Bug fixes

— [FIX] Fixed the WAPT Console : regression introduced in 1.5.1.22. Unable to login if the WAPT Server does not have a FQDN ;
— [FIX] *SetupHelpers* : winstartup_info fallback when `COMMON_STARTUP` folder does not exist, preventing a client to register properly ;
— [FIX] version/ revision in the WAPT System Tray utility dispkay the git hash instead of old svn revision number ;
— [FIX] Fixed the WAPT Console : update French translation for certs bundle hint ;
— [FIX] Fixed the WAPT Console : compare properly packages when number of version members differs 1.3 -<> 1.3.1 for example ;

58.5.33 WAPT-1.5.1.22 (2018-03-27)

Bug fixes

— [FIX] add Active Directory groups ;
— [FIX] newest only with `locale`, `architecture` and `maturity` ;
— [FIX] Import from external repository with mixed `locale`, `architecture` and `maturity` ;
— [ADD] `--setupargs` to **`waptdeploy`** ;
— [FIX] RPM ;
— [FIX] Enterprise build (**Enterprise** only) ;
— [IMP] different icons for WAPT Community and Enterprise editions ;
— [IMP] switch to Community features when no licence instead of aborting (**Enterprise**) ;
— some up to date Installed Packages marked as upgradable because of bad comparison `maturity` None/ maturity ;
— [IMP] `depends` and `conflicts` fields of HostsPackagesStatus table limited to 800 chars –> type changed to ArrayField to handle unlimited number of dependencies ;
— [NEW] git python module added as part of WAPT libraries ;
— [IMP] list organizational *unit* packages in group package table (**Enterprise**) ;
— [FIX] MongoDB to PostgreSQL database upgrade script ;
— [FIX] licence/ hosts count/ expiry check (**Enterprise**) ;
— [FIX] relative path for *verify_cert* ;

Known issues

— When the WAPT Server is searched with DNS SRV query (dnsdomain param), kerberos register authentication is not working.

58.5.34 WAPT-1.5.1.21 (2018-03-13)

Global architecture

— [IMP] multiple languages for description of packages. English, French, German, Spanish, Polish are handled as a start point. More to be added in the future ;
— [IMP] the description columns in the WAPT Console displays either languages depending on `language` setting in `waptconsole.ini`. In packages, `description_fr`, `description_en`, etc… have been added ;
— [IMP] when renaming hosts, old host package (matching previous host uuid) is now « removed » instead of forgotten ;
— [NEW] Handle AD organizational unit packages (**Enterprise** only ;)
— [NEW] package attributes :
— `locale` attribute : A computer can be configured to accept only packages with a specific locale ;
— `maturity` attribute : stores status like *DEV*, *PREPROD*, *PROD* to describe the level of completion of the package. Computers can be configured to accept packages with specified maturities. Default packages maturity of computer is both the empty one and *PROD* ;
— `impacted_process` attribute : csv list of process names which would be killed before install (**`install_msi_if_needed`**, **`install_exe_if_needed`**) and uninstall (by the mean of uninstallkey list). Could be used too in the future for « soft » upgrade remote action which upgrade softwares while they are not running ;

Setup/ WAPT upgrades

WAPTupgrade package

— [IMP] increased lifetime for upgrade task windows scheduler trigger for computers which are down for many days when upgrading ;
— [ADD] trigger at start of the computer ;

The WAPT Console

— [IMP] display of the list of embedded trusted packages certificates when building the custom WAPT Agent installer ;

Bug fixes

— [FIX] handle unicode filepaths for Packages Wizard ;
— [IMP] work in progress improvement of unicode handling globally in the WAPT Console ;
— [FIX] use proxy if needed for « download and edit » from external repo ;

SetupHelpers

— [FIX] Fixed bug in **`create_programs_menu_shortcut`** and **`create_user_programs_menu_shortcut`**. Shortcuts were created in `startup` and not `startup/programs`.

58.5.35 WAPT-1.5.1.19 rc1 (2018-03-08)

Global architecture

There is now some additional support for packages localization.

In Package `control` file, the *description_fr*, *description_en*, *description_de*, *description_pl*, *description_es* can be used to give description in respective french, english, german, polish languages.

If not set, the base description is used.

WAPT Console

58.5.36 WAPT-1.5.1.18 rc1 (2018-02-27)

Global architecture

There is a significant internal change on how python libraries are managed inside WAPT. This has implications on the way python scripts are launched. This change is only relevant for peoples launching WAPT processes manually.

We have removed the (not clean) sys.path manipulations inside wapt python scripts sources. The consequence is that all python scripts **MUST** be run with prior setting `PYTHONHOME` and `PYTHONPATH` pointing to WAPT home directory (`/opt/wapt` on Linux).

Failing to do so results in scripts claiming that libraries are missing.

On the WAPT Server running on Linux, libs are now in the default `/opt/wapt/lib/python2.7` location instead of using non standard former one.

— [IMP] WAPT has its own full python environment for libraries, even when debugging. Before, system wide python27 installation was needed for **`PyScripter`** to run.

Now, **`PyScripter`** can be started with a special batch file `waptpyscripter.bat` which sets the environment variables for python (`PYTHONHOME` and `PYTHONPATH`) and run **`PyScripter`** with python dll path set to wapt own copy.

— [NEW] Command line scripts with proper environment :
— *wapt-serverpostconf* on Linux server to start the WAPT Server postconf.py
— *wapt-scanpackages*
— *wapt-signpackages*

— [NEW] debugging commandline tools which setup python environment properly before running the python script.py before running the python script :
— to debug the WAPT service, launch in cmd as admin : **runwaptservice.bat** ;
— to debug the WAPT Server, launch in cmd : **runwaptservice.bat** or under linux : **runwaptserver.sh** ;
— to launch **PyScripter** without the need for local system wide python27 install, run **waptpyscripter.bat** ;

WAPT client

— [IMP] Add local wapt-get.ini settings *packages_whitelist* and *packages_blacklist* to restrict accepted packages from repository based on their package's name ;
— [IMP] More detailed reporting off host's repositories configuration (now includes dnsdomain, proxy, and list of trusted certificates) ;
— [FIX] fixed display in the Windows task bar of the login window (to allow in particular the autofill of the password by password managers) ; the WAPT Agent failing to compile if keys/ certificates already exist but the certificate had been removed from `C:\wapt\ssl` ;
— [NEW] Handle AD organizational unit packages (Enterprise edition)
— [IMP] Fallback to basic authentication when a host is registering on the WAPT Server if kerberos is enabled but authentication fails.
— [IMP] Improved **wapt-get.exe**, allow to designate configuration `wapt-get.ini` file with *–config* option with base name of user `waptconsole.ini` file (without ini extension) instead of full path. Handy when switching between several configurations. Same behavior as for the WAPT Console. Example :

```
wapt-get -c site3 build-upload c:\waptdev\test-7zip-wapt ;
```

— [FIX] Be sure to not loop for ever in websockets retry loop if something is wrong in the WAPT Server or websocket configuration.
— [FIX] Update PyScripter project template to use project directory as parameter for debug actions, and use relative paths for filenames.
— [FIX] incorrect package version comparison. Return True when comparing 1.2-1 to 1.2.1-3 (note : this is not homogeneous with the Version() class behavior. todo : merge both) ;
— [FIX] waptsetup : register and update **MUST** be launched with elevated privileges. So remove *runasoriginaluser* option.
— [NEW] Introduced attributes target_os and impacted_process for package's `control` file. They are not yet taken in account.
— [NEW] Introduced method to handle X509 client certificates authentication for repositories and the WAPT Server (specially for public WAPT Servers) ;
— [NEW] Introduced classes to generate X509 CRL ;

Setuphelpers

— [UPD] `setuphelpers.removetree` : Try to remove readonly flag when `remove_tree` reaches an Access Denied error ;
— [FIX] Fixed unicode handling in shell startup shortcuts ;
— [IMP] `waptutils.wget` can check sha1 or sh256 hashes in addition to md5, and can cache and resume partial downloads ;

WAPT Console

— [NEW] action in the WAPT Console to plan in near future a restart of the WAPT service on selected hosts ;
— [IMP] mass host update/upgrade in the WAPT Console actions are now launched in single shot instead of one host at a time ;
— [NEW] allow to force a host_dn in `wapt-get.ini` when host is not in a domain (**Enterprise** only) ;
— [NEW] *SetupHelpers* : added timeout parameter for `setuphelpers.service_start`, `setuphelpers.service_stop` and `setuphelpers.service_restart` ;
— [IMP] *group filter list* box is now editable, and one can type a partial group match and press enter to filter on all matching groups. Separator is comma (,). Handle * at the end of search to find all occurrences even if one group matches exactly ;

WAPT Server

— [ADD] bat script migrate-hosts.bat to set environment for `migrate-hosts.py`;
— [ADD] trigger_action.py script to trigger action on pre 1.5 hosts with reachable 8088 port from 1.5 WAPT Server;
— [FIX] `registration_auth_user` reset to None when reusing host certificate for re-register;
— [IMP] removed unnecessary dependencies krb5-user, msktutil, python-psutil for the WAPT Server package;
— [IMP] increase client_max_body_size for http post on nginx for large update/ upgrade trigger :
  — fix `signature_clockskew` parameter not taken in account in the WAPT Server configuration;
  — unified loggers for the WAPT Server;
  — have the WAPT Server ask WAPT clients to update status using websockets if websocket connection is up but database is not aware of given SID (case where the WAPT Server is restarted but **`nginx`** is kept up, and restart of the WAPT Server service is fast enough to not trigger a reconnection of the clients);
— [FIX] disable proxy for migrate-hosts;

Known issues

— WAPT service : if a system account level http proxy is defined in registry on the windows host, websocket client library tries to use it and fails to connect to the WAPT Server. Workaround : make an exception for the WAPT Server;
— In the WAPT Console : if a http proxy is defined in `waptconsole.ini`, section `[global]`, key *http_proxy*, it is used by the WAPT Console even if setting `use_proxy_for_xxx` is False Workround : set `http_proxy` to an empty string in `waptconsole.ini`;
— when using a not self-signed personal certificate, depending of th issuer, the certificate file <`private_dir`>`mine_cert.crt` can contain the full chain (own certificate, intermediate CA, and root CA). When the WAPT Console asks if the certificate should be put in authorized client certificate directory (<`wapt-dir`>`ssl`), the full `crt` file is copied as this. This means that all certificates in `crt` file are authorized, and not only the personal one. This is perhaps not desired;
Workaround : check if the personal pem encoded `crt` file contains the full certificates chain. If this is the case, copy in <`wapt-dir`>`ssl` only the parts of the PEM file matching the certificates you want to trust;
— SNI is not properly handled by the WAPT Console code, leading to incorrect error about certificate validation on WAPT https Server with virtual hosts;
— Certificates CSR updates (periodical signature, ...) must be managed manually using tools like easy-rsa. Only CSR accessible by a URL are supported;
— proxies are not supported on the WAPT Server, so CRL can not be updated properly (as far as Distribution Point is defined in certificates) if the WAPT Server has no direct http access to the distribution points;
— https certificates are verified on the clients using the bundle defined by the `verify_cert` ini settings. If this setting is simply *True*, the bundle supplied with python libraries is used to check issuers. This bundle is not updated unless WAPT is upgraded, so new issuers or no more trusted issuers are taken in account only at this point. So it is better to deploy your own CA bundle along with wapt and define the `verify_cert` path.
— for 1.5.1.18 rc1, on the linux server, there are broken symbolic links in `lib/python2.7` folder. Next RC does not exhibit this problem;

58.5.37 WAPT-1.5.1.14 (2018-01-09)

— [NEW] Historize in *wapt_localstatus* PostgreSQL table the dependencies and conflicts of installed packages (to provide an easy way to warn when conflicting package will be installed or should be removed);
— [FIX] load fill certificate chain from host packages to check `control` (as it is the case for other types of packages);
— [SEC] regression : check host package control signature right after downloading (it is checked too when starting install);
— [FIX] regression : do not install host package if version is lower than installed one;
— [FIX] Do not raise an exception during session-setup if package has no `setup.py`;

The WAPT Agent

— [FIX] intermediate CA pinning : Allow to deploy intermediate CA as authorized package CA without root CA (segragation of rules between entities) ;
— [FIX] old style print statement (without parentheses) raising an error in *setup-session* or *uninstall* **setup.py** functions ;

SetupHelpers

— [IMP] Added `setuphelpers.cache_dir` parameter to **wget** function ;
— [IMP] renamed *cabundle* parameter to *trusted_bundle* ;
— [NEW] Add python methods to create certificate from CSR ;

The WAPT Console

— [ADD] Added a checkbox in the WAPT Agent builder to sign with sha1 in addition to sha256 for old wapt client upgrades ;
— [IMP] force host package version to be at least equal to already installed host package (when host package is deleted, version was starting again at 0) ;
— [FIX] regression : check existing host package signature before editing it ;

The WAPT Server

— [FIX] Force the WAPT Server database structure upgrade at each WAPT Server startup ;
— [ADD] `db_connect_timeout` parameter for pool of the WAPT Server database connections ;
— [NEW] Store `depends` and `conflicts` attributes in the WAPT Server *HostPackagesStatus* PotsgreSQL table ;

Known issues

— SNI is not properly handled by the WAPT Console code, leading to incorrect error about certificate validation on the WAPT https Server with virtual hosts ;
— certificates CSR updates (periodical signature, ...) must be managed manually using tools like easy-rsa. Only CSR accessible by a URL are supported ;

58.5.38 WAPT-1.5.1.13 (2018-01-03)

— Quelques fallback pour permettre l'utilisation de la Console WAPT sous Wine.
— Ebauche architecture plugins dans la Console WAPT.
— Interface GUI pour entrer les mots de passe dans **PyScripter**.
— Action **wapt-get make-template** dans installeur crée un paquet vide.
— Inclusion de la chaine de certificats du signataire dans le paquet au lieu du seul certificat final.
— IMPROVE : gestion des certificats signés par une autorité intermédiaire pour les actions de la Console WAPT.
— Ajout option pour spécifier fichier de configuration pour la Console WAPT.
— [FIX] SNI pour la récupération de la chaine de certificats dans la Console WAPT.
— [ADD] added actions to launch mass updates/ upgrades, offer updates to the users (WAPT Enterprise).
— `F5` rafraîchit la liste des paquets.
— Changement à distance de la description de l'ordinateur.
— Possibilité de configurer plusieurs instances de serveurs Wapt sur un serveur/ VM.
— chunked http upload pour pouvoir uploader des gros paquets sans passer par un **scp**.
— Ajout installation forcée d'un paquet sur un poste dans la la Console WAPT.
— Ajout option pour masquer les actions avancées (simplication affichage de la Console WAPT).
— CN du Certificat / clé host sont nommés comme l'UUID.
— Si une ou plusieurs dépendances d'un paquet ne peuvent pas être installées, le paquet parent n'est pas installé et est marqué en erreur.
— Memory leak sur le serveur ?
— Gestion timezone pour validité de certificats.
— [SECURITY] prend tous les fichiers en compte dans la vérification des hashes, pas seulement ceux dans le répertoire racine (régression apparue en 1.5 mais non présente en 1.3).

58.5.39 WAPT-1.5.1.5 (2017-11-16)

Architecture globale

— [NEW] the host packages are now named with the BIOS *UUID* of the host instead of the *FQDN* (it is possible to use the FQDN as the UUID with the parameter *use_fqdn_as_uuid* but it may create duplicates in the WAPT Console) ;
— le service **the WAPT service** écoute sur l'adresse de loopback, port 8088 et non plus sur toutes les interfaces. Cela réduit la surface d'attaque potentielle si un attaquant spoofe l'adresse IP du serveur WAPT ;
— le service **the WAPT service** crée au démarrage une connexion Websockets (Socket.IO) vers le serveur pour permettre à la Console WAPT de déclencher les Update/ Upgrade / Install/ Remove ; On ne pass plus par le port 8088 du service ;
— [NEW] the Websocket requests from the WAPT Console to the WAPT agents are now signed with the key of the *Administrator*. Before, security relied on source IP restriction and the validation of the Administrator's login/ password ;
— la base de données d'inventaire est maintenant une base PostgreSQL en remplacement de MongoDB. Cela facilite le requêtage pour un reporting personnalisé, le langage SQL étant mieux connu des administrateurs système ;
— l'affichage dans la Console WAPT d'un grand nombre de machines a été amélioré. L'affichage de plusieurs milliers de machines n'est plus un problème ;
— modifier la configuration d'un grand nombre de machines a été rendu largement plus performant ;
— la reprise d'un téléchargement partiel de paquet est maintenant possible (interruption lors de l'arrêt …) ;
— les clés privées doivent maintenant obligatoirement être protégées avec un mot de passe ;

Console WAPT

— passage en Websockets ;
— gestion des écrans de haute résolution (ex : écrans 4k) ;
— modernisation des jeux d'icônes dans la Console WAPT ;
— changement à la volée de la description du poste ;
— option pour changer le mot de passe d'une clé ;

Format des paquets

— la présence du fichier `setup.py` est optionnelle (plus particulièrement, il n'est pas nécessaire pour les paquets groupes et machines qui ne contiennent que des dépendances) ;
— [NEW] if the package contains a `setup.py` file, it MUST be signed with a **Code Signing** certificate, otherwise the package WILL NOT be installed. The roles are now differenciated between the role of the *Package Deployer* (allowed to sign group and host packages) and the role of *Package Developer* (allowed to sign group, host AND base packages) ;
— lors de la signature du paquet, le certificat du signataire est ajouté dans le paquet (`WAPT/certificate.crt`) ;
— le fichier `manifest` est renommé `manifest.sha256` au lieu de `manifest.sha1` et `signature.sha256` au lieu de `signature` ;
— ajout des attributs suivants au fichier `control` :
— `signed_attributes` : pour la fiabilité de la vérification ;
— `min_wapt_version` : le paquet est ignoré (et ne s'installe pas) si wapt n'est pas au moins à cette version ;
— `installed_size` : le paquet ne s'installe pas s'il n'y a pas au moins cet espace disponible sur le disque système ;
— `max_os_version` : le paquet est ignoré si Windows a une version supérieure à cet attribut ;
— `min_os_version` : le paquet est ignoré si Windows a une version inférieure à cet attribut ;
— `maturity` : `PROD`, `PREPROD`, `TEST` ;
— `locale` : `fr`, `en`, etc ;

Configuration générale des agents

- section explicite `[wapt-host]` pour le dépôt des paquets machines sinon l'url est déduite de <repo_url>+"-host" ;
- section explicite `[wapt]` pour le dépôt principal, sinon <repo_url> est pris en compte ;
- vérification des certificats activée par défaut pour toutes les connexions https ;
- signature avec du sha256 au lieu de sha1 ;
- prise en compte de paquets signés avec des certificats délivrés par une autorité, déploiement uniquement du certificat de l'autorité ;
- utilisation de l'UUID du client pour le nom des paquets host au lieu du FQDN ;
- possibilité d'utiliser le FQDN comme UUID au lieu de l'UUID du Bios. (paramètre `use_fqdn_as_uuid`) (ou uuid forcé : paramètre `forced_uuid`) ;
- lorsqu'on signe, on désigne le signataire par son certificat et non sa clé privée. La clé privée est recherchée par wapt dans le même répertoire que le certificat personnel. On incite à avoir un certificat par personne agissant sur WAPT ;
- possibilité de prendre en compte la révocation de certificats (la CSR est fournie aux poste lors de l'update, dans le fichier Packages) ;
- re-signature possible sous Linux avec la commande **`wapt-signpackage.py`** ;
- installation dans `Program Files(x86)` par défaut ;

SetupHelpers

- `setuphelpers.running_as_admin`, `setuphelpers.running_as_system` ;
- correctif sur `add_shutdown_script` ;
- ajout paramètre *remove_old_version* pour `setuphelpers.install_msi_if_needed` et `setuphelpers.install_exe_if_needed` ;

wapt-get

- ajout fonction **`update-package-sources`** qui lance la fonction optionnelle **`update_package()`** du paquet ;
- remplacement de l'option *–private-key* par l'option *–certificate* pour désigner le certificat à utiliser pour signer le paquet. La clé privée est recherchée dans le même répertoire que le certificat ;
- remplacement du fichier `WAPT/wapt.psproj` à chaque édition d'un paquet (pour mettre à jour le chemin vers les modules WAPT suivant l'installation dans `C:\wapt` ou `C:\Program Files (x86)\wapt`) ;
- vérification du certificat serveur lors du **`enable-check-certificate`** pour éviter de mauvaises configurations ;

wapt-signpackages

- ajout options

```
--if-needed
--message-digest
--scan-packages
--message-digest
```

```
Usage: wapt-signpackages -c crtfile package1 package2

Re-sign a list of packages

Options:
-h, --help            show this help message and exit
-c PUBLIC_KEY, --certificate=PUBLIC_KEY
Path to the PEM RSA certificate to embed identitiy in
```

(suite sur la page suivante)

(suite de la page précédente)

```
control. (default: )
-k PRIVATE_KEY, --private-key=PRIVATE_KEY
Path to the PEM RSA private key to sign packages.
(default: )
-l LOGLEVEL, --loglevel=LOGLEVEL
Loglevel (default: warning)
-i, --if-needed       Re-sign package only if needed (default: warning)
-m MD, --message-digest=MD
Message digest type for signatures.  (default: sha256)
-s, --scan-packages   Rescan packages and update local Packages index after
signing.  (default: False)
```

Console WAPT

- [NEW] all actions sent to the hosts are signed with the Administrator's key ;
- [NEW] generation of a key / certificate pair signed by a Certificate Authority (WAPT Enterprise) ;
- option de créer un certificat **Code Signing** ou non (version Enterprise) ;
- option pour changer le mot de passe d'une clé RSA ;
- option de vérification des certificats lors de la création du **waptagent** ;
- lancement TISHelp (version Enterprise) ;
- limitation du nombre de machines retournées dans la Console WAPT ;
- ajout filtre *reachable* = poste connecté au serveur WAPT ;
- possibilité de changer la description du poste

The WAPT Server

- authentification sur une base LDAP (version Enterprise) ;
- utilisation des Websockets pour les actions ;

The WAPT service

- le Webservice http de **waptservice** écoute uniquement sur la loopback 127.0.0.1 (donc plus de vérification si port 8088 ouvert sur firewall..) ;
- le **waptservice** se connecte en websocket au serveur WAPT si le paramètre `waptserver` est présent dans `wapt-get.ini` ;
- le paramètre *websockets_verify_cert* active la vérification SSL du certificat pour la connexion websockets ;
- affichage de liste des certificats / CA autorisés pour les paquets ;
- affichage signataire paquet ;
- [NEW] *allow_user_service_restart* parameter allows a standard user to restart the WAPT service on her computer ;
- lancement de **tishelp** en mode service par URL /tishelp ;

Installeur waptagent

- suppression installation **msvcrt** ;
- restent uniquement 2 options : installer le service et lancer *wapttray* ;
- options pour une installation silencieuse :
- *dnsdomain* pour la recherche auto wapt et the Serveur WAPT
- *wapt_server*
- *repo_url*
- **waptupgrade** fait systématiquement une installation complète (pas d'installation incrémentale) ;

Improvements 1.5.0.12-amo –> 1.5.0.16

— `setup.py` pas obligatoire pour uninstall ;
— chemin unicode pour édition de paquets ;
— corrigé la recherche de dépots en s'appuyant sur les DNS ;
— corrigé \0000 pour PostgreSQL ;
— introduit une option pour avoir une double signature sha1 et sha256 ;
— vérification https pour upload **`waptagent`** ;
— option *–if-needed* dans **`wapt-signpackages`** ;
— fix proxy dans import paquets ;
— gestion des révocations de certificats (CSR) ;
— fix attributs requis dans signature actions ;
— *max_clients* ;
— fix option sans serveur (**`waptstarter`**) ;
— ajout lancement **`tishelp`** ;
— force update à l'installation ;

58.5.40 WAPT-1.4.0 (2017-05-05)

— pas de release officielle ;
— [NEW] migration sur la base PostgreSQL à la place de MongoDB ;

58.5.41 WAPT-1.3.13 (2017-07-25)

Security fix

— régression : Package files content check was skipped if signature of `manifest` and `Packages` index file checksum was ok. This regression affects all 1.3.12 releases, but not WAPT <= 1.3.9 and >= upcoming 1.5. In order to exploit this bug, one would need to tamper the `Packages` files either through a MITM (if you do not have valid https certificate check) or a root access on the WAPT Server.

Other changes

— compatibility with packages signed with upcoming WAPT 1.5. With WAPT 1.5, package are signed with sha256 hashes. An option allows to sign them with sha1 too so that they can be used with WAPT 1.3 without signing them again.
— new package certificate for Tranquil IT packages. previous certificate for package on store.wapt.fr has expired. all packages on store.wapt.fr has been signed again with new key / certificate with both sha1 and sha256 hashes, and WAPT 1.5 signature style (control data is signed as well as files)
— fix for local GPO add_shutdown_script() function (thanks jf-guillou !)
— fix for **`waptsetup.exe`** postinstall actions (**`update`** / **`register`**) when running **`waptsetup.exe`** installer without elevated privileges : added *runascurrentuser* flag
— remove needless python libraries to make install package slimmer

58.5.42 WAPT 1.3.12.13 (2017-06-26)

Console WAPT

— [NEW] Assistant de création de paquets à partir d'un fichier `MSI` ou d'un `Exe` ;
— [NEW] Option dans le menu *Outils* ou par drag drop dans l'onglet dépôt privé ;
— [NEW] Découverte des options silencieuses ;
— [NEW] Utilisation des fonctions **`install_exe_if_needed`** et **`install_msi_if_needed`** au lieu d'un simple **`run()`** pour les exes et les MSI (plusieurs templates de `setup.py` dans `C:\wapt\templates`) ;
— [NEW] Amélioration significative de la vitesse de modification en masse des paquets machines ;
— [NEW] Vérification optionnelle de la signature des paquets que l'on importe d'un dépôt extérieur. La liste des certificats autorisés se trouve par défaut dans `%APPDATA%\waptconsole\ssl` et peut-être précisée dans les paramètres de la **`waptconsole`**. Le paramètre ini se nomme `authorized_certs_dir`. Sinon, les certificats autorisés sont ceux dans `C:\wapt\ssl` ;
— [NEW] Vérification optionnelle du certificat https pour les dépôts extérieurs dans la Console WAPT ;
— [NEW] Vérification de la signature des paquets machines, groupes et logiciels avant leur modification dans la Console WAPT ou dans **`PyScripter`** ;
— [NEW] Lors de l'import d'un dépôt extérieur, possibilité d'éditer le paquet pour inspection plutôt que de le charger directement sur le dépôt de production ;
— [NEW] Changement des URL relatives à la documentation. https://www.wapt.fr/en/doc/ ;
— [NEW] Possibilité d'actualiser le certificat sans recréer la paire de clés RSA (en particulier pour préciser un Common Name correct, qui apparaît comme le signataire des paquets) ;
— [NEW] HTTPS par défaut pour les URL de dépot.

Autres correctifs

— [FIX] Paramètre `AppNoConsole` : `1` pour NSSM (**`waptservice`** / **`waptserver`**) pour permettre le fonctionnement sur Windows 10 Creators Updates ;
— [FIX] Problème de fichier Zip qui restent verrouillés si une erreur est déclenchée ;
— [FIX] Suppression répertoire temporaire lors de l'annulation d'édition d'un groupe ;
— [FIX] Gestion espace dans les fichiers de projet PyScripter ;
— [FIX] Gestion utf8 / unicode pour certaines fonctions ;
— [FIX] Fix gestion encoding quand **`run_not_fatal()`** renvoie une errreur ;
— [FIX] remplacement librairie mongo.bson par json natif de python ,
— [FIX] bug dans la synchro des groupes AD avec les paquets WAPT ;
— [FIX] bug « La clé privée n'existe pas » la première fois qu'elle est renseignée si on ne redémarre pas la Console WAPT ;
— [FIX] bug « redémarrage service wapt » (merci à QGull) ;
— [FIX] possibilité d'avoir des majuscules dans les noms de paquet (toutefois pas recommandé, les noms des paquets sont sensibles à la casse) ;
— [FIX] quelques actualisation des exemples de configuration `wapt-get.ini.tmpl`
— [FIX] la compilation du **`waptagent`** échoue si les clés / certificats existent déjà mais que le certificat a été supprimé de `C:\wapt\ssl` ;
— [FIX] affichage dans la barre des tâches de la fenêtre de login (pour permettre en particulier l'autofill par des gestionnaires de mot de passe) ;

58.5.43 WAPT 1.3.9.3 (2017-04-11)

— [FIX] Argument *shell* = *True* was not explicitly passed to the underlying function as it occurred on previous versions.

58.5.44 WAPT 1.3.9 (2017-03-03)

Fixes

— [FIX] update code to follow more PEP8 recommandations ;
— [FIX] upgradedb locks sqlite database issue ;
— [FIX] Fix broken DNS SRV record discovery ;
— [FIX] Fix unicode handling of signer / CN / organization in certificates ;
— [FIX] Unzipped netifaces module ;

wapt-get

— [NEW] Expands wildcards args for **wapt-get install**, **wapt-get show**, **wapt-get build-package**, **wapt-get sign-package** ;
— [FIX] Fix **wapt-get show-params** ;
— [FIX] Fix **wapt-get register** with description not working on some computers ;
— [FIX] Fix broken *-c –config* option ;

SetupHelpers

— [NEW] `setuphelpers.reg_key_exists` ;
— [NEW] `setuphelpers.reg_value_exists` ;
— [NEW] `setuphelpers.run_powershell` ;
— [NEW] `setuphelpers.remove_metroapp` ;
— [NEW] `setuphelpers.local_users_profiles` ;
— [NEW] `setuphelpers.get_profiles_users` ;
— [NEW] `setuphelpers.get_last_logged_on_user` ;
— [NEW] `setuphelpers.get_user_from_sid` ;
— [NEW] `setuphelpers.get_profile_path` ;
— [NEW] `setuphelpers.wua_agent_version` ;
— [NEW] `setuphelpers.local_admins` ;
— [NEW] `setuphelpers.local_group_memberships` ;
— [NEW] `setuphelpers.local_group_members` ;
— [IMP] command :*run* : explicit default values for `setuphelpers.run` command help in **PyScripter**. Added *return_stderr argument* (overloaded str object) ;
— [FIX] `setuphelpers.run_notfatal` : fix unicode issue in use wmi module for `setuphelpers.wmi_info_basic` instead of `setuphelpers.wmic` shell command ;
— [IMP] `setuphelpers.make_path` : improved when first argument is a drive. Be smart if an argument is a callable ;
— [FIX] `setuphelpers.CalledProcessError` : restored code :*setuphelpers.CalledProcessError* alias.
— [ADD] `setuphelpers.host_infos` : added *profiles_users*, *last_logged_on_user*, *local_administrators*, *wua_agent_version* attributes ;
— [IMP] `setuphelpers.ensure_unicode` : return None if None, for bytes strings, try utf8 decoding before system locale decoding ;

The WAPT Console

— [FIX] restore allowed lowercase/uppercase package naming ;
— [ADD] 4 host popup menu actions :
— *Computer Mgmt* ;

— *Computer Users* ;
— *Computer Services* ;
— *RemoteAssist* ;
— [FIX] fixed other issues in the WAPT Console :
— Don't search host while typing ;
— utf8 search (accents…) ;
— utf8 compare ;
— try to get localized versions of special folders ;

Setup

— [ADD] **waptpythonw.exe** binary in distribution for the WAPT Console less python scripts (to avoid having **cmd.exe** windows poping up when invoking a python script) ;
— [FIX] change default wapt templates URL to https://store.wapt.fr/wapt ;
— [FIX] when upgrading, (full **waptagent.exe** install) remove stalled **waptagent.exe** installs ;

58.5.45 WAPT 1.3.8.2 (2016-11-18)

Security

— [SEC] Fix inheritance of rights on wapt root folder for Windows 10 during setup when installed in `C:\wapt`. On Windows 10, **cacls.exe** does not work and does not remove « Authenticated Users » from `C:\wapt`. **cacls.exe** has been replaced by **icacls.exe** :
— on pre-wapt 1.3.7 systems, you can fix this by running the following command, or upgrade to wapt 1.3.8 (you may check `icacls.exe c:wapt /inheritance:r`)
— This can be achieved with a GPO, or a wapt package
— [IMP] in next versions of WAPT, the default install path of wapt will be changed from root folder `C:\wapt` to a more standard `C:\Program Files (x86)\wapt`.
— [IMP] By default, **waptsetup.exe** / **waptsetup-tis.exe** do not distribute certificates to avoid to deploy directly packages from Tranquil IT. **waptagent.exe** by default distributes the certificates that are installed on the mangement desktop creating the **waptagent**.

Core changes

— [IMP] The database structure has changed between 1.3.8 and 1.3.8.2 to include additional attributes from packages : *signer*, *signer_fingerprint*, *locale*, and *maturity*. *signer* and *signer_fingerprint* are populated when signing the package to identify the origin. This means local WAPT database is upgraded when first starting WAPT 1.3.8.2 and this is not backward compatible ;
— [IMP] Installers have a limited set of options, the most common use of WAPT is privileged ;
— [ADD] 3 new parameters for the **waptexit** policy behavior : *hiberboot_enabled*, *max_gpo_script_wait*, *pre_shutdown_timeout*. These parameters are not set by default and should be added to `wapt-get.ini` *[global]* section if needed ;
— [IMP] Use user's `waptconsole.ini` configuration file instead of `wapt-get.ini` for the commands targeted to package development (*sources*, *make-template*, *make-host-template*, *make-group-template*, *build-package*, *sign-package*, *build-upload*, *duplicate*, *edit*, *edit-host*, *upload-package*, *update-packages*. This avoids the need to write these parameters in `wapt-get.ini` on the development workstation. These parameters are not shared across multiple users on same host. One use case is to allow multiple profiles (key, upload location) depending on the maturity of package (development, test, production…) ;

SetupHelpers

— [ADD] helper functions `setuphelpers.dir_is_empty`, `setuphelpers.file_is_locked`, `setuphelpers.service_restart` and `setuphelpers.WindowsVersions` class

— [IMP] Added referer and *user_agent* in `setuphelpers.wget` and `setuphelpers.wgets`
— [IMP] run function : define stdin as PIPE to avoid lockup process waiting for input or error like unable to duplicate handle when using for example powershell
— [IMP] Version class : try to compare version using at least Version.members_count
— [FIX] encoding fixes for registry functions, fix encoding for registry_setstring key name
— [FIX] `setuphelpers.install_exe_if_needed` : do not check uninstall_key or min_version if not provided
— [FIX] `setuphelpers.install_exe_if_needed` and `setuphelpers.install_msi_if_needed` version check if *–force*
— [UPD] Check version and uninstall key after install with **`setuphelpers.install_exe_if_needed`** and `setuphelpers.install_msi_if_needed`
— [UPD] inventory includes informations from WMI.Win32_OperatingSystem
— [ADD] `setuphelpers.get_disk_free_space` helper function
— [UPD] check free disk space when downloading with `setuphelpers.wget`. Check http status before.
— [UPD] Version class : Version("7")<Version("7.1") should return True

wapt-get

— [ADD] Added 2 commands to get the WAPT Server SSL certificate and activate the certificate checking when using https with the WAPT Server
— [FIX] Fixed **`get_sources`** to allow svn checkout of a new package project
— [FIX] Fixed **`register`** problems with some BIOS with bitmaps
— [UPD] Check uninstall key after package install if uninstallkey is provided
— [FIX] Fixed OS compatibility in `manifest` file for **`wapt-get`** and **`waptconsole`** version windows
— [FIX] Fixed erroneous error messages for **`session-setup`** in the WAPT Console
— [UPD] Added « pattern » parameter to all_files function
— [FIX] Install Date incorrectly registered by **`register_uninstall`**
— [ADD] Added the **`user_local_appdata`** function
— [ADD] Added the `signer` CN and `signer_fingerprint` to `control` file when building a WAPT package
— [ADD] Added the control attribute `min_wapt_version` to trigger an exception if `Package` requires a minimum level of libraries. The version is checked againts **`setuphelpers.py`** "s __version__ attribute.
— [ADD] Added the `authorized_certificates` attribute that is sent to the WAPT Server. It contains the list of host's signer certificates distributed on the host
— [FIX] Fixed that when signing, check if WAPT zip file has already a `signature` file (python zipfile can not replace the file inline).

The WAPT service
— [ADD] Show *All Versions* checkbox in *Available Packages* page
— [UPD] Skin updated
— [ADD] Added *Filter* searchbox for available packages

The WAPT Console
— [ADD] Added *NOT* checkbox for keywords search in **`waptconsole`** to search for hosts NOT having a specific package or software…
— [FIX] Fixed integer limit for grid display of package size, use int64 for size of packages in **`waptconsole`**.
— [UPD] Do not list packages of section « restricted » in local webservice available packages list
— [UPD] Updated : *Common Name* attribute should be populated now, so that signer identity is not None in package `control` file.
— [ADD] Added signer's identity column in packages grid
— [FIX] Fixed escape quotes in WAPT package's description
— [ADD] Check **`waptagent.exe`** version against **`waptsetup-tis`** version at **`waptconsole`** startup.
— [UPD] try to display a *progress* dialog at **`waptconsole`** startup
— [FIX] company not set when building customized **`waptagent.exe`**
— [ADD] initialize Organization in **`waptagent.exe`** build with CN from certificate.

The WAPT Exit utility

— [UPD] some text introduction changes

The WAPT Tray utility

— [NEW] Limit trayicon balloon popup when Windows version is above Windows 7 or if `notify_user` = `False` in `wapt-get.ini`

The WAPT Server

— [UPD] Use broadcast address on interface for wakeonlan call
— [FIX] Removed the check of the WAPT Server password which prevents the proper registration of **waptserver** on Windows.
— [UPD] When upgrading, reuse existing `waptserver.ini` file if it already exists, do not overwrite the `server_uuid` and ask for password reset if it already exists

The WAPT Deployment utility / waptupgrade

— [FIX] Fixed **waptdeploy** not working on WinXP removed DisableWow64FileSystemRedir on **runtask**.
— [FIX] Fixed **waptupgrade** : Missing quotes for system account on Windows XP

Libraries

— [ADD] Added BeautifulSoup for wapt packages auto updates tasks
— [UPD] Updated **winsys** library update to "1.0b1"

58.5.46 WAPT 1.2.3.2 (2015-05-05)

— [ADD] *UUID* parameter for direct requests to hosts from the WAPT Server ;
— [ADD] allow host to refuse request if not right target (if ip has changed since last **update_status** for example)
— [ADD] fallback on the WAPT Server usage_statictics if mongodb lacks aggregate support
— [IMP] register host on the WAPT Server in postconf using **waptservice** http instead of command line **wapt-get**

58.5.47 WAPT 1.2.2 (2015-04-22)

— [ADD] **reset-uuid** and **generate-uuid** for https://roundup.tranquil.it/wapt/issue421 duplicated *UUID* issues
— [IMP] mass hosts delete, added delete hosts package action. WAPT Server >=1.2.2 only : https://roundup.tranquil.it/wapt/issue433
— [ADD] read the docs theme for sphinx *SetupHelpers* API documentation. WIP https://roundup.tranquil.it/wapt/issue427
— [IMP] doc updates
— [ADD] api/v1/hosts_delete method
— [ADD] **need_install**, **install_exe_if_needed**, **install_msi_if_needed** functions to *SetupHelpers*
— [ADD] parameters for **waptdeploy**.

58.5.48 WAPT 1.2.1 (2015-03-26)

WAPT Console

— [ADD] combobox for filtering on groups in **waptconsole**.
— [ADD] *Add ADS Groups as packages* action to WAPT host selection popup menu
— [ADD] **cleancache** action to clean local packages cache in the WAPT Console
— [ADD] added **notify_server** on network reconfiguration if **waptserver** is available ;
— [IMP] column *groups* shows only host's direct dependencies with package's section == « group » instead of all direct dependencies.
— [ADD] optional anonymous statistics (nb of hosts, nb of packages, age of updates…) sent to Tranquil IT to document the communication around WAPT (sent by **waptconsole** at most every 24h)
— [IMP] improved mass hosts delete,
— [ADD] delete hosts package action. WAPT Server >=1.2.2 only : https://roundup.tranquil.it/wapt/issue433
— [IMP] big packages uploads (write uploaded packages by chunk) (but still some issues on 32bits WAPT Servers due to **uwsgi**)
— [IMP] display version of mismatch when editing package

— [FIX] host's packages not saved when some dependencies do not exist anymore
— [FIX] restore working *Cancel running task* button
— [FIX] canceling subprocesses not working in freepascal apps (when waiting for **InnoSetup** compile for example)

wapt-get / WAPT service

— [ADD] **reset-uuid** and **generate-uuid** for https://roundup.tranquil.it/wapt/issue421 duplicated *UUID* issues
— [IMP] **find_wapt_repo_url** processus to avoid waiting for all repos if one repo is ok (improved response time in buggy networks)
— [IMP] windows DNS resolver in wapt client (python part) instead of pure python resolver. Should reduce issues when multiple network cards or inactive network connections.
— [IMP] changed priority of WAPT Server discovery using SRV dns records. –> first priority ascending and weight descending. –> comply with standards.
— [FIX] solved some issues with **SQLite** and threads in local **waptservice**
— [IMP] explicit transaction handling and *isolation_level* = *None* for local waptDB (to try to avoid locks)
— [IMP] teardown handler for **waptservice** to commit or rollback thread local connections
— [FIX] for waptrepo detection in freepascal parts : same processus as python part.
— [FIX] for **edit_package** when supplying a wapt filename instead of package request

SetupHelpers

— [ADD] read the docs theme for sphinx *SetupHelpers* API documentation. WIP https://roundup.tranquil.it/wapt/issue427
— [ADD] _all_ list to avoid importing unecessary names in **setup.py** modules. Now only functions defined in *SetupHelpers* are available when importing *SetupHelpers*. This can break some WAPT packages if names were indirectly imported through *SetupHelpers* module.
— [ADD] **need_install**, **install_exe_if_needed**, **install_msi_if_needed** functions to *SetupHelpers*.
— [ADD] **local_desktops** function
— [FIX] version class instances accept to be compared to str
— [REM] `setuphelpers.processnames_list` which is unused in *SetupHelpers*
— [ADD] `setuphelpers.add_ads_groups` and `setuphelpers.get_computer_groups` to **waptdevutils.py**
— [FIX] `setuphelpers.run` helper
— [FIX] on_write callback not working
— [FIX] TimeoutExpired not formatted properly
— [FIX] use closure for registry keys

The WAPT Deployment utility

— [IMP] Improved the WAPT Deployment utility with more command line options (in particular tasks to merge to default inno-setup selected tasks)
— [FIX] waptrepo detection using dns records

Install

— [FIX] **waptagent** upload error on windows
— [FIX] debian packages should work for Jessie
— [IMP] **copytree2** for **waptupgrade**
— [FIX] trap exception for version check on copy of *.exe* and *.dll*
— [FIX] **mongodb-server** version should be >= 2.4

58.5.49 WAPT-1.1.1 (2015-02-26)

WAPT Console

— [IMP] Improved the loading of the main grid has been optimized ; only configured columns are displayed ;
— [IMP] Improved the WAPT Server : detects the hosts whose **waptservice** is listening. Their *Reachable* status is shown with a green / grey indicator ;
— [IMP] Improved the WAPT package to upgrade WAPT on hosts (? ? ?-waptupgrade.wapt) is generated by the WAPT Console at the same time as the WAPT agent installer (**waptagent.exe**), the two files are then uploaded on the WAPT Server ;
— [ADD] Added the package dependencies of each host are displayed in the grid. This allows to see what hosts have no package ;
— [ADD] Added possibility to trigger available package upgrades on hosts that are listening from the WAPT Console. In that case, the host sends its status to the WAPT Server after the upgrade ;
— [ADD] Added possibility to filter hosts in the WAPT Console according to their upgrade status or whether they are « reachable » or not,
— [ADD] When packages are flagged for install but are not yet installed on a host, they appear with a blue « + » indicator. It is then possible to force the immediate install of the package with a right-click ;

The WAPT service

— [ADD] cleaning of the cache on the hosts after each successful upgrade ;

The WAPT Server

— [ADD] the versions of the WAPT agent, WAPT Server are shown in the main web page of the WAPT Server (with a red indicator if there is a problem) ;

SetupHelpers

— [ADD] Added functions to *SetupHelpers* to manage shortcuts :
  — `setuphelpers.remove_desktop_shortcut` ;
  — `setuphelpers.remove_user_desktop_shortcut` ;
  — `setuphelpers.remove_programs_menu_shortcut` ;
  — `setuphelpers.remove_user_programs_menu_shortcut`.

Installation

— [IMP] verification of used ports during the post-configuration of WAPT Server on a Windows host ;

Webservices

— [IMP] the **waptserver** no longer listen on 8080 port by default.

The Apache frontal web server listens in HTTP and HTTPS and relays action calls to the python **waptservice** that only listens locally.

It is therefore necessary to update `wapt-get.ini` files on WAPT agents and to replace *wapt_server* = http://srvwapt.mydomain.lan:8080 with *wapt_server* = https://srvwapt.mydomain.lan

If you can not make that change to your WAPT agents, it is possible to return to the previous behavior.

On Debian, edit the file `/opt/wapt/waptserver/waptserver.ini`, and in the `[uwsgi]` section, put :

```
```

http-socket = 0.0.0.0 :8080

On Windows, edit `C:\waptwaptserver\waptserver.ini` and replace :

```
```

server = Rocket((“127.0.0.1”, port), “wsgi”, {« wsgi_app » :app})

with :

```
```

server = Rocket((“0.0.0.0”, port), “wsgi”, {« wsgi_app » :app})

The repository may stay in HTTP on port 80.

The calls to the WAPT Server are authenticated, but it is advized to restrict access to authorized sub-networks with a firewall.

— [IMP] json calls to the webservice of the WAPT Server are now standardized ;
— [IMP] when launching **`update`** / **`upgrade`** / **`remove`** / **`forget`** / **`tasks_status`** actions from the WAPT Console, the IP address of the host is no longer sent, but instead its *UUID*, and it is the WAPT Server that finds the IP address and the port to use ; et c’est le serveur wapt qui s’occupe de déterminer quelle IP / port utiliser ;
— [ADD] verification in the WAPT Console that the version of the WAPT Server is sufficient ;
— [ADD] the timeout to connect to WAPT agents and read the data are configurable in `waptserver.ini` ;

58.5.50 WAPT-1.0 (2015-01-31)

— [ADD] first public version of WAPT

CHAPITRE 59

WAPT SOFTWARE LICENSE AGREEMENT

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.

59.1 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.

59.2 2. GRANT OF RIGHTS

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.

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

59.3 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.

59.4 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.

59.5 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.

59.6 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.

59.7 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.

59.8 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.

59.9 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.

59.10 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.

CHAPITRE 60

External component licenses used in WAPT

Le développement du logiciel WAPT a commencé en mars 2012 ; il est porté en très grande partie par l'équipe de Tranquil IT.

With WAPT >= 1.9, developments done within WAPT are licensed under a *proprietary license*.

TABLEAU 1 – 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