Ubuntu: Gestion des paquets avec APT

Advanced Packaging Tool est un système complet et avancé de gestion de paquets, permettant :

- une recherche facile et efficace,
- une installation simple et une désinstallation propre de logiciels et utilitaires.

Il permet aussi de facilement tenir à jour une distribution Ubuntu avec les paquets en versions les plus récentes et de passer à une nouvelle version de Ubuntu, lorsque celle-ci est disponible.

APT est un ensemble d'utilitaires utilisables en ligne de commande. Il dispose aussi de nombreuses interfaces graphiques, dont Synaptic et Muon, et d'interfaces en ligne de commande, comme apt-get et Aptitude, afin d'en rendre l'utilisation plus ergonomique.

Utiliser apt derrière un proxy

Définition d'un proxy

L'installation du proxy est très facile dans /etc/apt/apt.conf, ajouter:

```
Acquire::http::Proxy "http://proxy.example.com:8080";
Acquire::https::Proxy "http://proxy.example.com:8080";
```

Chaque fois que j'utilise apt à partir de maintenant, il utilisera le proxy défini.

Remarque: il faut généralement conservé le fichier /etc/apt/sources.list original qui contient les référentiels Ubuntu standard (archive.ubuntu.com).

L'option DIRECT: Pour ne pas utiliser de proxy pour certains référentiels

Lorsqu'on a un référentiel interne apt va tenter d'en extraire les correctifs et les installations de celui-ci via le proxy.

Mais ce référentiel étant hébergé en interne, le proxy (public) ne peut pas accéder à celui-ci. Cela pose des problèmes, car apt-get update continue d'accrocher au message “En attente d'en-têtes” dès que l'URL du référentiel interne est vérifiée:

La page de manuel d'apt.conf contient les informations suivantes:

```
http::Proxy est le proxy http à utiliser par défaut. Il se présente sous la
forme standard http://[[utilisateur][:pass]@]hôte[:port]/. Les proxy peuvent
```

```
également être spécifiés en utilisant le formulaire http::Proxy:: avec le
mot-clé spécial DIRECT signifiant ne pas utiliser de proxy. La variable
d'environnement "http_proxy" remplacera tous les paramètres.
```

Cela signifie qu'en utilisant le mot-clé **DIRECT**, une URL définie (dans ce cas, l'URL du référentiel interne) peut être accédée directement sans utiliser le proxy défini:

```
cat /etc/apt/apt.conf
Acquire::http::Proxy "http://proxy.example.com:8080";
Acquire::https::Proxy "http://proxy.example.com:8080";
Acquire::http::Proxy::repo.internal.local DIRECT;
```

Une autre exécution de apt-get update fonctionne maintenant très bien:

Gestion de l'authentification d'archive

apt-secure

Lorsqu'on utilise un référentiel non signé dans Ubuntu cela retourne une erreur

```
W: GPG error: http://www.deb-multimedia.org jessie
InRelease: The following signatures couldn't be verified because the public
key is not available: NO_PUBKEY 5C808C2B65558117
E: The repository 'http://www.deb-multimedia.org jessie InRelease' is not
signed.
N: Updating from such a repository can't be done securely, and is therefore
disabled by default.
N: See apt-secure(8) manpage for repository creation and user configuration
details.
```

Depuis sa version 0.6, apt sait vérifier la signature du fichier Release de chaque archive. On s'assure ainsi que les paquets de cette archive ne peuvent pas être modifiés par quelqu'un qui ne possède pas la clé de la signature du fichier Release. À partir de la version 1.1, apt exige que les dépôts fournissent des informations récentes d'authentification pour une utilisation libre du dépôt.

Si une archive possède un fichier Release non signé ou pas de fichier Release du tout, les versions actuelles d'APT refuseront par défaut d'en télécharger des données dans les opérations update. Même si un frontal tel que apt-get(8) est forcé de télécharger, il demandera une confirmation explicite si une installation inclut un paquet d'une archive non authentifiée.

Côté client : installer une clé

Une installation par défaut possède toutes les clés pour obtenir en toute sécurité des paquets des dépôts par défaut, cependant lorsqu'on souhaite ajouter des dépôts tiers il est possible d'ajouter une

clé:

1. télécharger la clé en utilisant un canal fiable pour ce téléchargement.
2. ajouter la clé avec la commande apt-key
3. lancer la commande apt-get update pour télécharger et vérifier le fichier InRelease ou Release.gpg de l'archive ainsi configurée.

Côté fournisseur: signer des paquets

Lorsqu'on dispose d'un référentiel local et que l'on veut signer les archives dont on a la responsabilité, il faut :

1. créer un fichier Release à la racine de l'archive, s'il n'existe pas déjà. On peut le créer avec la commande `apt-ftparchive release` (fournie dans le paquet `apt-utils`)
2. le signer, avec les commandes `gpg -- clearsign -o InRelease Release` et `gpg -abs -o Release.gpg Release`
3. publier l'empreinte de la clé. Ainsi les utilisateurs de l'archive connaîtront la clé qu'ils doivent importer pour authentifier les fichiers de l'archive. Le mieux est de diffuser sa clé dans son propre paquet de trousseau comme le fait Debian avec `debian-archive-keyring` pour ensuite distribuer automatiquement les mises à jour et les transitions de clés
4. fournir les instructions pour ajouter l'archive et la clé. Si les utilisateurs ne peuvent récupérer de façon sûre la clé, la chaîne de confiance est rompue. La façon d'aider les utilisateurs à ajouter la clé de l'archive dépend de l'archive et de l'audience cible : cela va d'un paquet de trousseau inclus dans une autre archive que des utilisateurs ont déjà configurée (comme les dépôts par défaut de leur distribution) à la mobilisation du web de confiance.

Chaque fois que le contenu de l'archive change, (suppression ou ajout de nouveaux paquets) le responsable doit refaire les deux premières étapes.

Forcer la mise à jour à partir d'un référentiel non signé

On voit combien cela peut devenir fastidieux surtout lorqu'on veut utiliser un dépôt purement interne, mais on peut contourner certaines mesures de protection importantes en utilisant les options suivantes:

utilisation de l'option AllowUnauthenticated

Le manuel d'apt-get indique :

```
--allow-unauthenticated Ignore if packages can't be authenticated and don't
prompt about it.
This can be useful while working with local repositories, but is a huge
security risk if
data authenticity isn't ensured in another way by the user itself. The usage
of the Trusted
option for sources.list(5) entries should usually be preferred over this
```

```
global override.
Configuration Item: APT::Get::AllowUnauthenticated.
```

On peut donc bypasser le contrôle en utilisant l'option –allow-unauthenticated à apt-get ainsi:

```
sudo apt-get --allow-unauthenticated upgrade
```

Pour rendre ce paramètre permanent utiliser un fichier de configuration dans /etc/apt/apt.conf.d/. Le nom du fichier peut être 99myown et peut contenir cette ligne:

```
 APT::Get::AllowUnauthenticated "true";
```

De cette manière, il ne sera plus nécessaire d'utiliser cette option à chaque fois que l'on souhaite installer un logiciel.

Mais il faut être prudent sur l'utilisation de cette option plus largement, les garanties sont en place pour protéger. De plus cette méthode ne fonctionne pas pour les mises à jour et apt retourne un message du type:

```
N: Les mises à jour depuis un tel dépôt ne peuvent s'effectuer de manière
sécurisée,
et sont donc désactivées par défaut.
N: Voir les pages de manuel d'apt-secure(8) pour la création des dépôts et
les
détails de configuration d'un utilisateur.
```

Utilisation de options de confiance

L'option de confiance est ce qui désactive la vérification de GPG. On peut définir ces options dans sources.list :

```
deb [trusted=yes] http://www.deb-multimedia.org jessie main
```

Une autre solution générique serait d'utiliser allow-insecure:

```
deb [allow-insecure=yes] http://www.deb-multimedia.org jessie main
```

Ceci est très utile lorsqu'on souhaite affiner les paramètres de sécurité pour des entrées individuelles.