NAME

apt-secure - Archive authentication support for APT

DESCRIPTION

Starting with version 0.6, **apt** contains code that does signature checking of the Release file for all archives. This ensures that packages in the archive cant be modified by people who have no access to the Release file signing key.

If a package comes from a archive without a signature, or with a signature that apt does not have a key for, that package is considered untrusted, and installing it will result in a big warning. **apt-get** will currently only warn for unsigned archives; future releases might force all sources to be verified before downloading packages from them.

The package frontends **apt-get(8)**, **aptitude(8)** and **synaptic(8)** support this new authentication feature.

TRUSTED ARCHIVES

The chain of trust from an apt archive to the end user is made up of several steps. **apt-secure** is the last step in this chain; trusting an archive does not mean that you trust its packages not to contain malicious code, but means that you trust the archive maintainer. Its the archive maintainers responsibility to ensure that the archives integrity is preserved.

apt-secure does not review signatures at a package level. If you require tools to do this you should look at **debsig-verify** and **debsign** (provided in the debsig-verify and devscripts packages respectively).

The chain of trust in Debian starts when a maintainer uploads a new package or a new version of a package to the Debian archive. In order to become effective, this upload needs to be signed by a key contained in the Debian Maintainers keyring (available in the debian-keyring package). Maintainers keys are signed by other maintainers following pre-established procedures to ensure the identity of the key holder.

Once the uploaded package is verified and included in the archive, the maintainer signature is stripped off, and checksums of the package are computed and put in the Packages file. The checksums of all of the Packages files are then computed and put into the Release file. The Release file is then signed by the archive key for this Debian release, and distributed alongside the packages and the Packages files on Debian mirrors. The keys are in the Debian archive keyring available in the debian-archive-keyring package.

End users can check the signature of the Release file, extract a checksum of a package from it and compare it with the checksum of the package they downloaded by hand - or rely on APT doing this automatically.

Notice that this is distinct from checking signatures on a per package basis. It is designed to prevent two possible attacks:

- Network man in the middle attacks. Without signature checking, malicious agents can introduce themselves into the package download process and provide malicious software either by controlling a network element (router, switch, etc.) or by redirecting traffic to a rogue server (through ARP or DNS spoofing attacks).
- Mirror network compromise. Without signature checking, a malicious agent can compromise a mirror host and modify the files in it to propagate malicious software to all users downloading packages from that host.

However, it does not defend against a compromise of the Debian master server itself (which signs the packages) or against a compromise of the key used to sign the Release files. In any case, this mechanism can complement a per-package signature.

USER CONFIGURATION

apt-key is the program that manages the list of keys used by apt. It can be used to add or remove keys, although an installation of this release will automatically contain the default Debian archive signing keys used in the Debian package repositories.

In order to add a new key you need to first download it (you should make sure you are using a trusted communication channel when retrieving it), add it with **apt-key** and then run **apt-get update** so that apt can download and verify the InRelease or Release.gpg files from the archives you have configured.

ARCHIVE CONFIGURATION

If you want to provide archive signatures in an archive under your maintenance you have to:

- *Create a toplevel Release file*, if it does not exist already. You can do this by running **apt-ftparchive release**(pro vided in apt-utils).
- *Sign it*. You can do this by running **gpg --clearsign -o InRelease Release** and **gpg -abs -o Release.gpg Release**.
- *Publish the key fingerprint*, that way your users will know what key they need to import in order to authenticate the files in the archive.

Whenever the contents of the archive change (new packages are added or removed) the archive maintainer has to follow the first two steps outlined above.

SEE ALSO

apt.conf(5), **apt-get(8)**, **sources.list**(5), **apt-key(8)**, **apt-ftparchive(1)**, **debsign(1)**, **debsig-verify(1)**, **gpg(1)**

For more background information you might want to review the **Debian Security Infrastructure**[1] chapter of the Securing Debian Manual (available also in the harden-doc package) and the **Strong Distribution HOWTO**[2] by V. Alex Brennen.

BUGS

APT bug page[3]. If you wish to report a bug in APT, please see /usr/share/doc/debian/bug-reporting.txt or the **reportbug(1)** command.

AUTHOR

APT was written by the APT team <apt@packages.debian.org>.

MANPAGE AUTHORS

This man-page is based on the work of Javier Fernndez-Sanguino Pea, Isaac Jones, Colin Walters, Florian Weimer and Michael Vogt.

AUTHORS

Jason Gunthorpe

APT team

NOTES

1. Debian Security Infrastructure
   http://www.debian.org/doc/manuals/securing-debian-howto/ch7
2. Strong Distribution HOWTO
   http://www.cryptnet.net/fdp/crypto/strong_distro.html
3. APT bug page
   http://bugs.debian.org/src:apt