Confined Applications on the Ubuntu Desktop

Perth Linux Users Group, April 2018

James Henstridge
Canonical Ltd.

Traditional Linux Packaging

Distribution provides an archive of a large amount of software

Usually curated by a fairly small set of maintainers

Software updates are usually confined to security fixes

Major version updates usually only come with new distro releases

Package Archive Security

Distributor’s public key installed locally

Archive index covering current versions of packages plus hashes

Archive index is cryptographically signed with distributor’s private key

Package Installation

Files are unpacked to arbitrary locations on file system

Scripts may be run before or after file installation, removal or upgrade

- Scripts run as root, so can perform arbitrary system modifcations
- Badly written scripts might not make reversible changes on removal

Additional Third Party Software

Add an additional archive to the package manager:

- A URL to find the archive index
- A public key to verify updates

Problems

Once a user places their trust in a third party archive, it can do anything the distribution maintainers can do

- Including replacing packages provided by the distro

No guarantee that a combination of third party archives will work together

Third party may not support all distribution releases

Changing use of UNIX systems

Modern Linux desktops are not used the same way as traditional multi-user UNIX systems.

Traditional:

- Ordinary users are potential adversaries
- Protect root account at all costs

Modern:

- All important data owned by the single user on system

Malicious Software doesn’t need root

MOTHERBOARD

VICE

INTERNET INSECURITY

Flight Simulator Add-On Tried to Catch Pirates By Installing Password-Stealing Malware on Their Computers

Malware included in the installer for pirated versions of the game steals usernames and passwords from Chrome: 'This is by far one of the most extreme, and bizarre, methods of DRM we've ever seen.'

Snap packaging

Atomic updates

Run applications inside a confined sandbox

One package to run over multiple distributions/releases

Third party developers can self-publish to the store

Snapd installed by default on Ubuntu, Solus. Available for most major Linux distros.

Atomic Updates

Snap packages are squashfs file system images

- Read only, with each file individually compressed

Packages are mounted rather than being extracted into the main file system

- No partial installs, and easy rollbacks

Since packages are retained and unmodified, can be used as basis for delta based updates

Application Confinement

Multiple technologies used to construct the sandbox:

- Mount namespaces
- Seccomp filters
- AppArmor mandatory access control

Applications expected to conform to sandbox rather than sandbox conform to applications

Separates apps from each other as much as apps from the host system

Mount Namespaces

Like chroots on steroids

After “unsharing” the mount namespace, you can perform mount operations that are not visible to the parent process

Rearrange file system with bind mounts (possibly read only)

Seccomp filters

Seccomp is a kernel feature that restricts the system calls a process can execute based on a policy written in the Berkeley Packet Filter language

Originally developped to support Chromium’s multi-process sandbox

Forbidden syscalls can kill the process or just return an error

If an app shouldn’t have network access, we can block the network system calls

AppArmor

A mandatory access control system

Used to control file access independent of file permissions

Also used to mediate access to D-Bus services

AppArmor protection not available on all distros

Common package runtimes

To make a package work across multiple distributions and releases, we don’t want to use the host system’s libraries

Using mount namespaces, we can provide a different root file system to apps providing the same libraries everywhere

Base file system is read only and also distributed as a snap

Currently most packages rely on an Ubuntu 16.04 base (“core”), but different snaps can use different bases

The Store

Developers can register a package name and then upload releases

- Can publish packages to multiple “channels” (edge, beta, candidate, stable)

Uploads are subject to automated review

- On success, package can be published to a channel immediately
- On failure, the package either needs to be resubmitted or manually reviewed

Snap packages appear in GNOME Software

Featured snaps

 Spotify

 Hiri

 WebStorm

 Firefox

 Ao

 SDLPoP

 Tusk

 gravit-designer

 onlyoffice-desktopeditors

 goxel

 Noson

 Ora

 Shattered Pixel Dungeon

 BunqDesktop

 Slack

 Skype

Future Improvements

Better security

- Encourage use of Wayland for confinement
  - Closes off some attack vectors present in X11
- Add support for xdg-desktop-portal
  - Services to support common operations blocked by confinement
  - Allow user to grant access to files outside of a snap’s data directory, reducing the need for the home interface

Improved support for exposing desktop themes to apps

References

More information about snaps:

https://snapcraft.io

Building snaps:

https://docs.snapcraft.io