Towards a more secure operating system without sacrificing usability

The GNOME challenge
Philosophy behind GNOME
Inclusiveness...
... end user experience
Accessible & usable by everyone

绰号 l10n
绰号 i18n
绰号 Accessibility
绰号 Usability
Freedom
“Filtering out extraneous information is one of the basic functions of consciousness”
— Barry Schwarz
IF YOU FORCE THE USER TO BE A PART OF A SECURITY SYSTEM

YOU'RE GONNA HAVE A BAD TIME
Prompts are dubious
Security prompts are wrong
Interrupting the user to make a permanent security decision is EVIL.
Untrusted connection

This connection is untrusted. Would you like to continue anyway?

The identity provided by the chat server cannot be verified.

The certificate is self-signed.

Certificate Details

☐ Remember this choice for future connections

[Buttons: Cancel, Continue]
The software is not signed by a trusted provider.

Do not update this package unless you are sure it is safe to do so.

Malicious software can damage your computer or cause other harm. Are you sure you want to update this package?

Close  Force install
Abrt found a new update which fix your problem. Please run
before submitting bug: pkcon update --repo-enable=fedora --
repo-repo=updates-testing tracker-0.14.1-1.fc17. Do you
want to continue with reporting bug?

No

Yes
Ellisons Law:
For every keystroke or click required to use a security feature the userbase declines by half.
NOT SURE IF

B IS 8
# .caffrc -- vim:ft=perl:
# This file is in perl(1) format - see caff(1) for details.

$CONFIG{'owner'} = 'Username';
#$CONFIG{'email'} = '[user]@[domain]';
#$CONFIG{'reply-to'} = 'foo@bla.org';

# You can get your long keyid from
#   gpg --with-colons --list-key <yourkeyid|name|emailaddress..>
#
# If you have a v4 key, it will simply be the last 16 digits of
# your fingerprint.
# Example:
#   $CONFIG{'keyid'} = [ qw{FEDCBA9876543210} ];
# or, if you have more than one key:
#   $CONFIG{'keyid'} = [ qw{0123456789abcdef 89abcdef76543210} ];
#   $CONFIG{'keyid'} = [ qw{0123456789abcdef 89abcdef76543210} ];

# Select this/these keys to sign with
#$CONFIG{'local-user'} = [ qw{0123456789abcdef 89abcdef76543210} ];

# Additionally encrypt messages for these keyids
LET'S MAKE THEM USE BASE16, OCAML, AND PERL FOR THEIR CRYPTO
I DON'T ALWAYS TARGET USERS

BUT WHEN I DO, IT'S ME
Wähle einen Schlüssel

610CB25237B370E9EB2108E89CEE1B6B059B598E
Tobias Mueller   muelli@cryptobitch.de   None
Expires: 2019-07-06 11:11:31

F289F7BA977DF4143AE9FDFBF70A02906C301813
Tobias Mueller   tobiasmue@gnome.org   None
Expires: 2018-02-28 12:00:00
Um den Key signiert zu bekommen, muss eine andere Person den Sicherheitscode oder den Barcode scannen.

**Key Details**

Fingerprint: F289 F7BA 977D F414 3AE9  FDFB F70A 0290 6C30 1813

UIDs: Tobias Mueller <tobiasmue@gnome.org>

**Sicherheitscode**

F289 F7BA 977D F414 3AE9  
FDFB F70A 0290 6C30 1813
To sign the key, confirm that you want to sign the following key.
This will generate an email that must be sent in order to complete the signing process.

Key
A0FF 4590 BB61 22ED EF6E 3C54 2D72 7C7 6869 7734

UIDs
Alfa Test <alfa@example.net>
Alpha Test <alpha@example.net>
Alice <unknown>
Containerise all the Apps!
Challenges for containerised Apps

- Access to X, DRI
- DBus, other Apps
- File-IO
- Sound, Video, Printing, ...
- Grant access temporarily rather than wholesale
Flatpak
A new way of distributing applications in GNU/Linux

- Cross-distribution deployment
- Runtimes and applications (OSTree)
- Sandboxing (bubblewrap)
- Invisible to the user
- Directly connect users and app developers
Bubblewrap
Namespaces, cgroups, seccomp

🔍 Sandbox apps in chroot-like environments as an unprivileged user
🔍 Implements a subset of the Kernel’s user namespaces feature to isolate processes
🔍 Allows passing a list of seccomp filters to limit syscalls
The Sandbox – classic security

- Limited access to the host system by default:
  - No access to processes outside the sandbox (namespaces)
  - No access to the network, session bus and devices
  - Controlled execution of certain syscalls (seccomp filters)
  - Read-only access to the runtime and app (bind mounts)
  - Read-write access to $HOME/.var/app/$APPID
  - Controlled access to resources (cgroups)
  - No access to host services (e.g. X/Wayland, system bus...)

very limiting by default, but there are ways of dealing with that to run real-word applications...
Punching holes

- Grant access to UNIX domain sockets: X.org, Wayland, PulseAudio, System and Session D-Bus...
- Grant access to specific devices: dri, kvm
- Grant access to see, use and/or own specific D-Bus names
- Share specific subsystems with the host (network, IPC)
- Fine-grained permissions for filesystem access
- Define extensions for runtimes or applications (e.g. l10n)
Escaping the Sandbox through Portals – modern security through interactivity

- Grant access to UNIX domain sockets: X.org, Wayland, PulseAudio, System and Session D-Bus...
- Grant access to specific devices: dri, kvm
- Grant access to see, use and/or own specific D-Bus names
- Share specific subsystems with the host (network, IPC)
- Fine-grained permissions for filesystem access
USB Security
When do you use USB?
And when not?
And who else uses your USB when you’re not aware..?
CVE - CVE-2016-0133
The USB Mass Storage Class driver in Microsoft Windows Vista SP2, Windows Server 2008 SP2 and R2 SP1, Windows 7 SP1, Windows 8.1, Windows Server ...
https://www.cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2016...

CVE - CVE-2013-3200
The USB drivers in the kernel-mode drivers in Microsoft Windows XP SP2 and SP3, Windows Server 2003 SP2, Windows Vista SP2, Windows Server 2008 SP2 ...
www.cve.mitre.org/cgi-bin/cvename.cgi?name=cve-2013-3200

CVE - CVE-2010-1083
The processcompl_compat function in drivers/usb/core/devio.c in Linux kernel 2.6.x through 2.6.32, and possibly other
This thumbdrive hacks computers. “BadUSB” exploit makes devices turn “evil”

Researchers devise stealthy attack that reprograms USB device firmware.
>>sudo python usb_inhibit.py
sudo python usb_inhibit.py -- allow 0x