Secure boot implementation status
A report from the Debian EFI team

Debian EFI team

DebConf 2018

July 31, 2018

Slide build version: Thursday 19th July, 2018. 37b8857f53bbdcbf70523bdc100803d279314f01

Overview

1. Context
   - Secure boot explained in short
   - SB goal for Debian
   - What is Shim
2. Signing Infrastructure
   - Package generation
   - Template binary package
3. Current status
   - Signing service
   - Dak
   - Packages

Overview

Boot sequence

- Firmware → Boot loader → System
- UEFI → Grub → Linux Kernel

Secure boot goal

Prevent a **REMOTE** attacker to temper with the boot sequence

How?

Firmware has a set of embedded certificates; **chain of trust**

$$\begin{tikzcd} \boxed{\textbf{Firmware}} \arrow[r, bend right, "\text{signature verification}"'] & \boxed{\textbf{Boot loader}} \arrow[r, bend right, "\text{signature verification}"'] & \boxed{\textbf{System}} \end{tikzcd}$$

Only against remote attacks?

- UEFI allows changing certificates with physical access to the machine through **boot services**
- "secure boot" != "trusted/measured boot"

General goal

- Boot only binaries **signed by Debian** when SB is enabled
- Generic infrastructure signing any whitelisted package

Inconvenients

- Machines for end users doesn't contain Debian certificates from the shelfs
- Current process to install Debian:
  - Disable secure boot; or
  - Install Debian certificates by ourselves
- Scary to newcomers
  (Do I need to disable **Secure** B..? Doesn't sound right)
- Inconvenient to the cloud
  (or any place with no easy access to the physical machine)

Microsoft

- Machines certificated by Microsoft:
    - Embedded MS certificate
    - Users are allowed to install their own certificates (x86_64)
    - MS has a signing service that allows organizations to get their blob signed by them
- Get Grub signed by MS?

Inconvenients to get Grub signed by MS

- Grub's code is too big
- Frequent bug fixes
- Frequent new features
- Frequent updates
- Frequent new versions
- Every Grub version signed by MS $\rightarrow$ not viable
- Workaround: **Shim**

Shim

- Shim is a **simple** bootloader with the only goal to load the next boot loader (Grub)
- Small code and non frequent new versions
- Shim allows embedding a certificate in its code
  - Shim $\rightarrow$ sigined by MS
  - Grub $\rightarrow$ signed by Debian

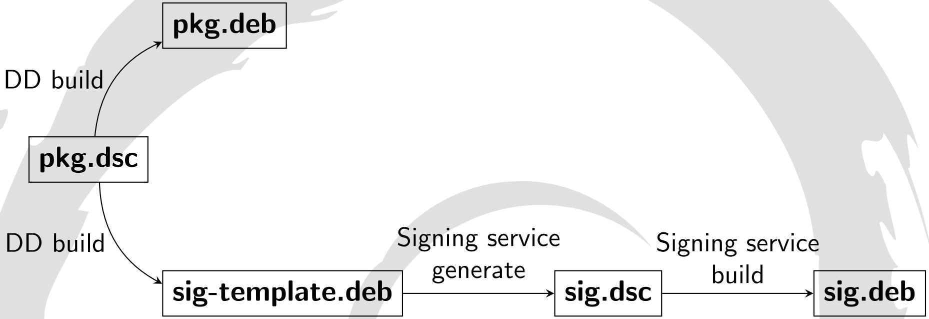
Debian boot sequence for SB

$$\begin{tikzcd}
\boxed{\textbf{UEFI}} \arrow[r, bend right, "\text{MS sig verif}"'] & \boxed{\textbf{Shim}} \arrow[r, bend left, "\text{Debian sig verif}"] & \boxed{\textbf{Grub}} \arrow[r, bend right, "\text{Debian sig verif}"'] & \boxed{\textbf{Kernel}} \arrow[r, bend left, "\text{Debian sig verif}"] & \boxed{\textbf{Modules}}
\end{tikzcd}$$

- Debian out-of-the-box:
  - SB doesn't need to be disabled
  - Less scary to users
- "Assurance" that boot sequence was not tempered by a remote attack

Overview

Package generation



*.dsc represents the source package

sig.dsc: the source package of the signed version

- Sig source package (sig.dsc) is generated **automatically** by the signing service
- It contains dettached signatures
- Its build depends on the unsigned pkg.deb
- Build is simple: attaches signatures to the files from pkg.deb
- **Build is reproducible**

Package flow

$$\begin{tikzcd}
\boxed{\textbf{DD}} \arrow[r, "\text{dput}"] & \boxed{\textbf{Dak}} \arrow[r, "\text{trigger}"] & \boxed{\textbf{Signing service}} \arrow[r, "\text{dput}"] & \boxed{\textbf{Dak}}
\end{tikzcd}$$

- Signing service
  - Maintains an audit log of every file that got signed
- Dak: Debian archive kit
  - Trigger post-accept event when a template is detected (whilelist is verified)
  - Embargoed: wait signed package from signing service before publishing both packages

Template binary package structure

- dpkg -x template.deb
- /usr/share/code-signing/<template-bin-pkg-name>/
  - files.json:

    contains a list of files to be signed
  - source-template/:

    folder with the structure to generated the new source package
- Signing service copies dettached signatures to debian/signatures/
- Then it executes: dpkg-genchanges ... && debsign ... && dput ...

Overview

Signing service

- Code available at https://salsa.debian.org/ftp-team/code-signing (SB Sprint 2018)
- State: functional and deployed in experimental suite
  signing packages with a fake Debian key
- Audit log kept in a sqlite DB
- TODO: notify maintainers in case of failures to process the package
- TODO: Backup system for the audit log
- TODO: Think about key and signature revocation process
- TODO: Deploy to stable / testing / unstable

DAK

- TODO

Packages to be signed

- shim boot services
- fwupdate
- grub2
- kernel

shim

- TODO

fwupdate

- TODO

grub2

- TODO

kernel

- TODO

Thanks

Questions?