Secure Boot without UEFI: Booting VMs on Power(PC)

—

Daniel Axtens
Linux Security Engineer

IBM

Contents

Power what now?

Previously: best known for PowerPC Apple Macs.

Today: Large, enterprise-focused servers.

IBM PowerVM: firmware (Type 1) hypervisor and system management software suite.

Customer workloads run as Logical Partitions (LPARs), which are paravirt VMs.

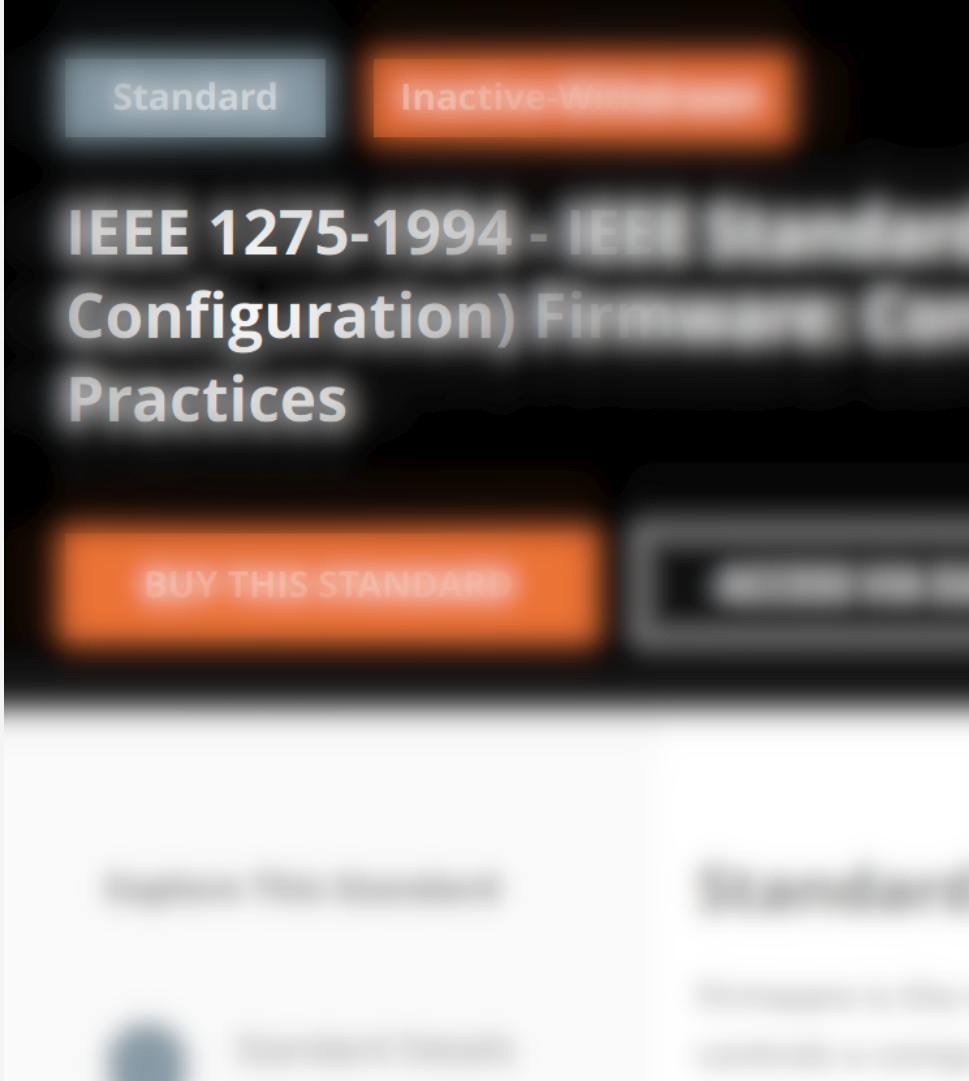
We want to **securely boot a Linux LPAR**.

Bare-metal (OpenPower) machines also exist, they boot differently.

Feature	E950
MTM	9040-MR9
System Packaging	4U
Processor Socket	2S to 4S
# of cores	32, 40, 44, or 48 cores
Memory DIMM Slots	128 DDR4 ISDIMMs
Memory – Max	16TB
Built-In IBM PowerVM	Yes
PCIe Gen4 Slots	10 Slots
Operating System	AIX, Linux

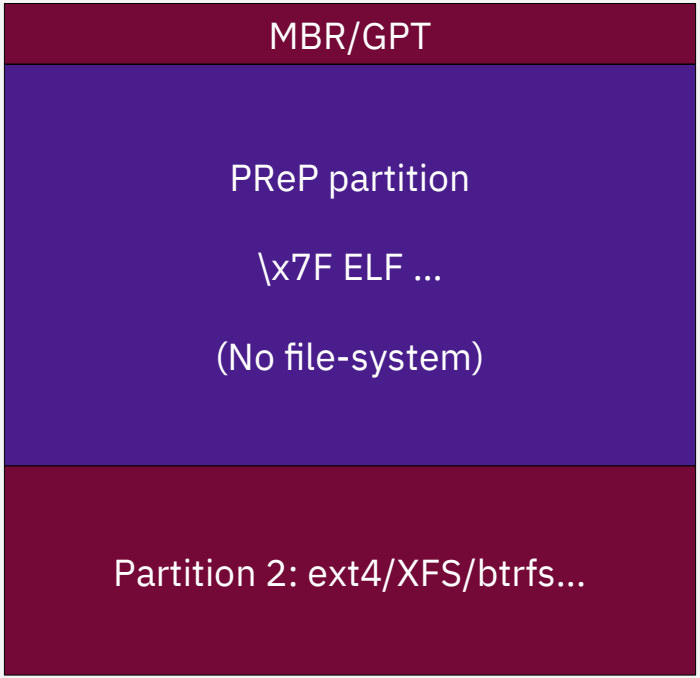
Boot environment

- OpenFirmware (IEEE1275) environment.
- Device Tree based configuration.
- 32-bit BE **ELF** bootloader. (vs UEFI with PE format binary).
- Grub platform: powerpc-ieee1275.



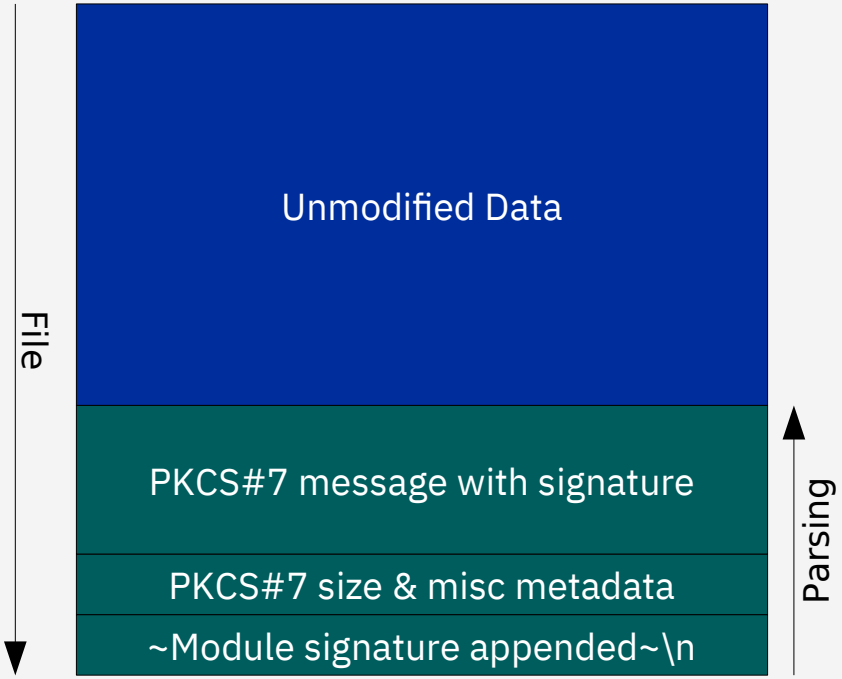
Where does the bootloader come from?

- **First install:** Media with a CHRP \ppc\boot-info.txt file pointing to the bootloader on disk.
- **Subsequent boots:** Loaded from the PReP partition. No file-system, raw bytes on disk.
- **Netbooting**
- **Compare to UEFI:**
  - \EFI\BOOT\BOOT*.EFI
  - EFI System Partition
  - PXE



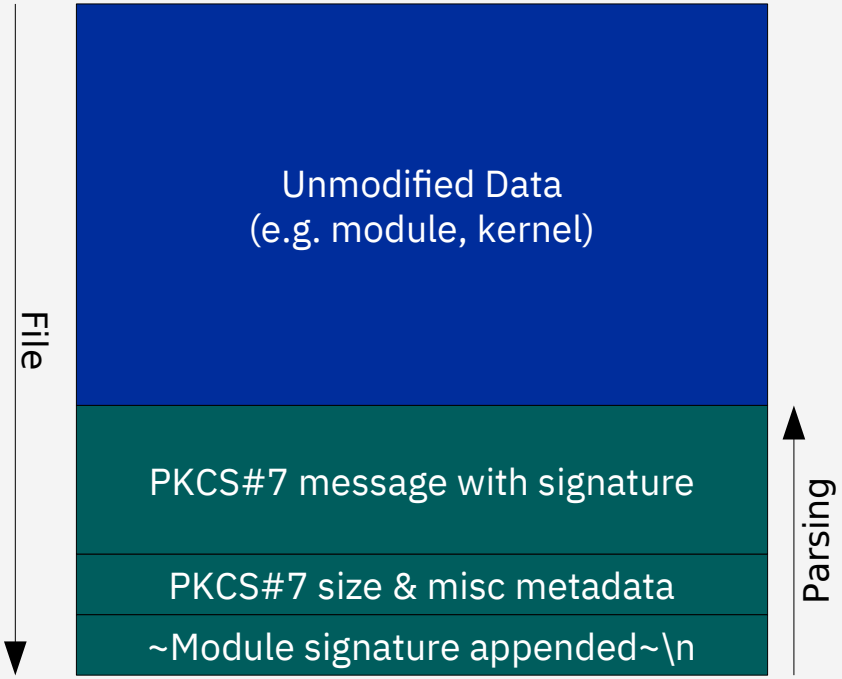
Appended Signatures

- **Simple**:
  - At the end of the data being signed.
  - Computed over all the original data.
  - Oblivious to the data being signed.
- **Existing crypto**: PKCS#7/CMS signedData message.
- **Existing tools to construct and verify:** sign-file, extract-module-sig.pl, OpenSSL.



Appended Signatures

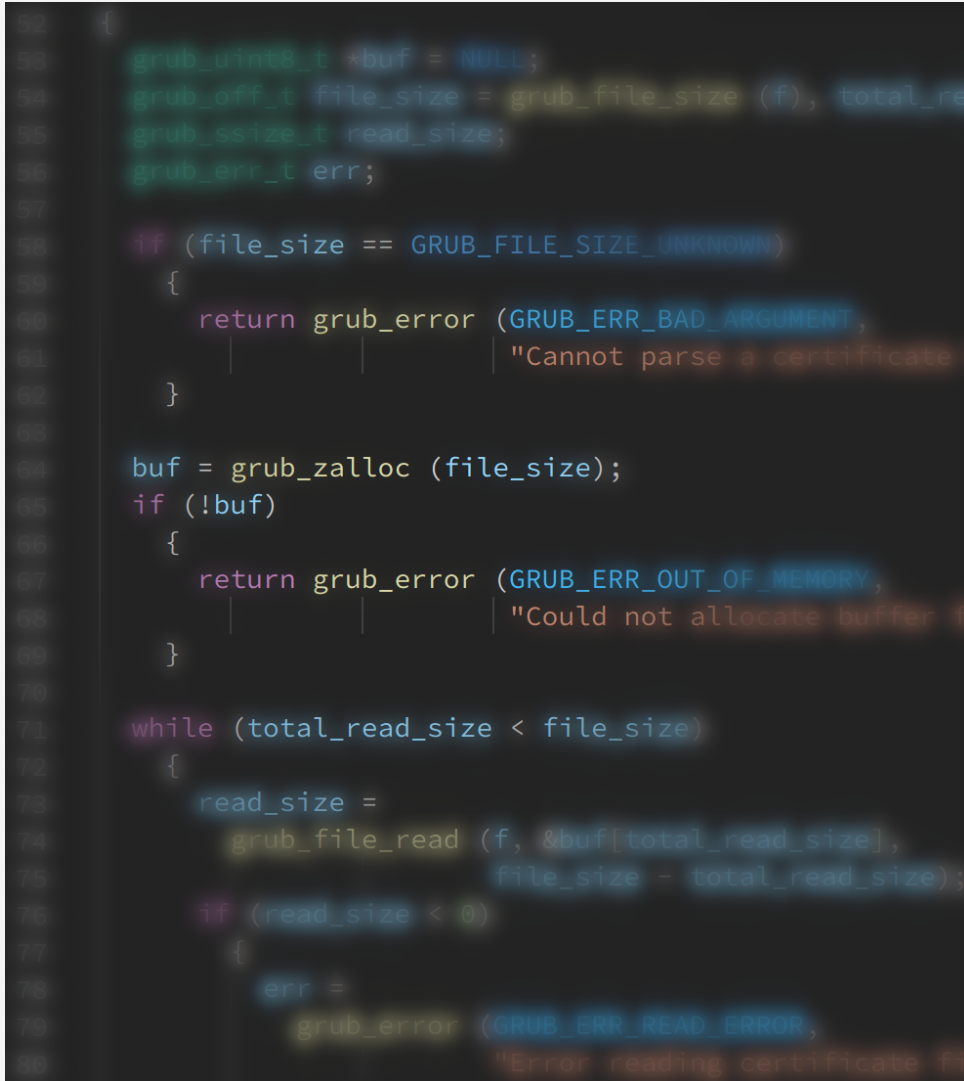
- Originally used to sign Linux kernel modules.
- Can be used to sign kernels, verified by IMA on kexec.
- Distros sign ppc64le kernels like this already for OpenPower secure boot.




PFW
grub
Linux

Verifying Linux from Grub

- **Teach grub to verify appended signatures with an x509 certificate.**
- Use existing grub features and concepts: e.g. verifier interface.
- Write as little crypto-adjacent code as possible:
  - PKCS#7 and x509 are based on ASN.1: import libtasn1.
  - Borrow PKCS#7 and x509 definitions from GnuTLS.
  - Use existing gcrypt code to do the actual maths.




PFW
grub
Linux

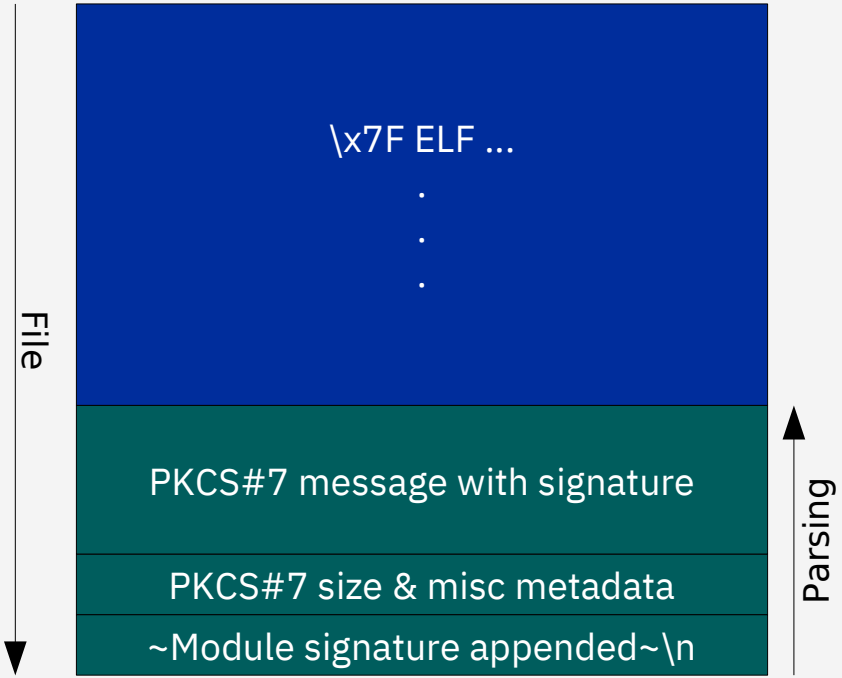
Signing Grub: design constraints

- **Be backwards compatible.**
  - Not changing ELF & PReP partitions.
  - Not moving to UEFI.
- Re-use as many tools and concepts as possible.
  - Especially don’t invent new crypto.
  - One set of key formats and management processes is enough.
- Support multiple signers.

Verifying Grub from Firmware

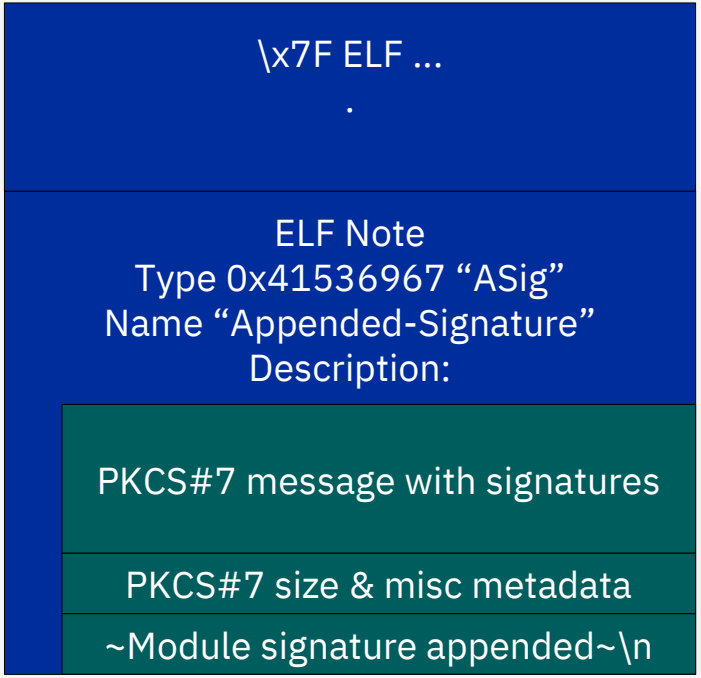
If we use an appended signature, **how does FW know if it is present**?

- Grub often loaded from PReP partition: no file system, no file size, no clear EOF.
- Length of message varies depends on number of signers, signing keys, and hash types: can't skip forward a fixed amount.
- Would prefer not to 'keep scanning': if you install a new unsigned grub you might find an old, now invalid signature.



Appended Signature Note

- We want:
  - all data to be within ELF structures
  - an ordinary appended signature that usual, unmodified tools understand
- **Add a new SHT_NOTE section at the end of the binary to contain the signature.**
  - descsz = ALIGN(signature size, 4)
  - Padding at the start of description, so appended signature magic is always at the end of the file.



Multiple signatures

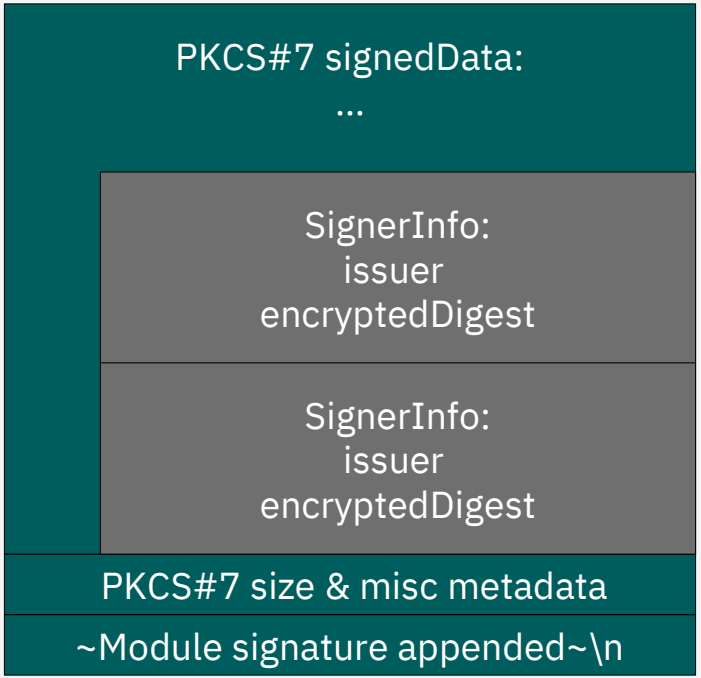
Use cases:

1) Key rotation: distro signs with 2 keys, one for old FW, one for new FW.

2) Production infrastructure with own keys, while maintaining existing distro signature.

   Signatures made at different points in time. Second signer doesn't have access to first signer's key material.

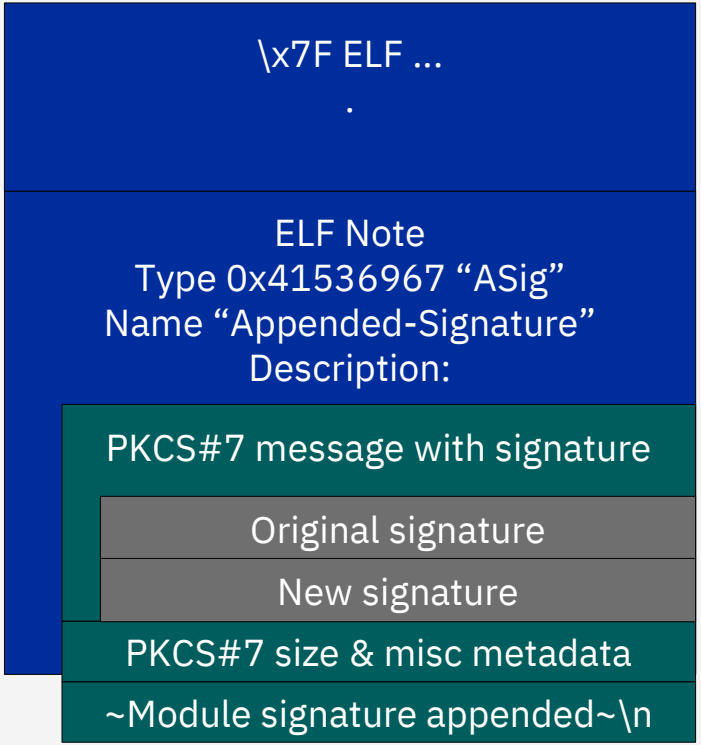
In theory, appended sigs/PKCS#7 work for these use cases, but tools are lacking for (2).



Multiple signatures

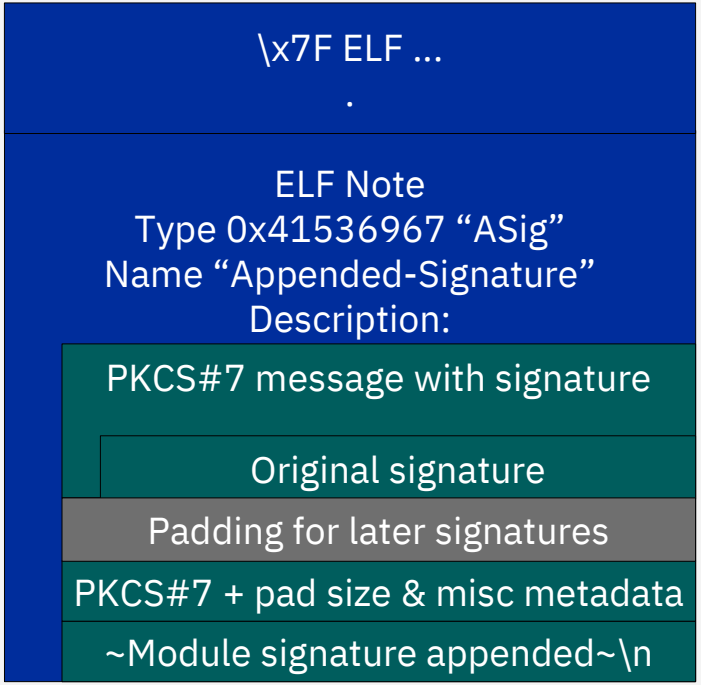
The ELF note contains the size of the appended signature, and is itself signed.

- Everything works if both signatures are added at once (key rotation use case).
- Adding a signature later would either spill past the end of the ELF binary, or require changing the size of the note, breaking the first signature.
- Not hashing the size would break some properties of appended signatures we want to maintain: oblivious to data being signed, made using standard tools.



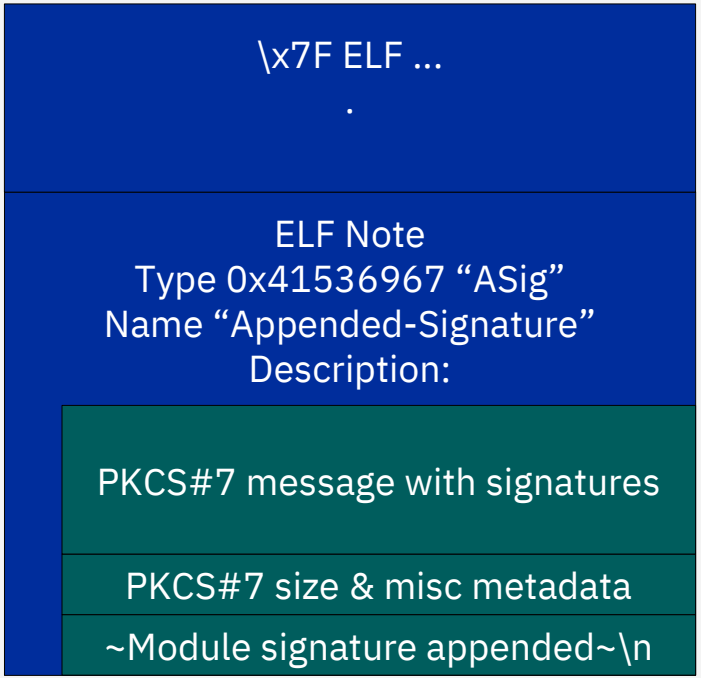
Fixed size signature block

- Allocate extra size for the PKCS#7 message to grow into. (Padding is discarded by parsers.)
- Pros:
  - Maintains appended-signature properties we like.
  - One place to implement in FW for all sources of boot material.
- Cons:
  - Limit to the number of subsequent signatures that can be added.
  - Abuse of the size field in metadata.



Summary of Design

- Appended signatures everywhere.
- Grub is signed with an appended signature.
  - An ELF note is added to the grub core image to state the presence and size of the appended signature on Grub.
- X.509 certificates are embedded in grub.
- Grub can parse the appended signature on the Linux kernel and verify it against the embedded certificates.
- Both parts should be portable to other non-UEFI architectures/platforms.




PFW
grub
Linux

Upstream status

Sign grub with an appended signature:
https://lists.gnu.org/archive/html/grub-devel/2020-08/msg00037.html

Verify appended signatures from grub:
https://lists.gnu.org/archive/html/grub-devel/2020-10/msg00152.html

Link enforcement to secure-boot property advertised by firmware (RFC):
https://lists.gnu.org/archive/html/grub-devel/2020-10/msg00048.html

Thank you

Daniel Axtens
Linux Security Engineer
—
daniel.axtens1@ibm.com

The views expressed in this presentation are those of the author, not necessarily those of IBM.