

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

—

Daniel Axtens
Linux Security Engineer

Contents

Background

The powerpc-ieee1275 platform 03

Appended signatures 04

Verifying Linux from Grub 05

Demo 06

Verifying Grub from Firmware 07

Appended Signature note 08

Signing flow 09

Upstreaming status 10

powerpc-ieee1275

- OpenFirmware (IEEE1275) environment.
- Device Tree based configuration.
- 32-bit BE ELF bootloader. Usually loaded from the PReP partition. No file system - raw bytes on disk.
 - UEFI: PE bootloader, EFI system partition has a filesystem
- Used for VMs/Logical Partitions (LPARs) on modern Power Systems, and for old Macs.

Standard

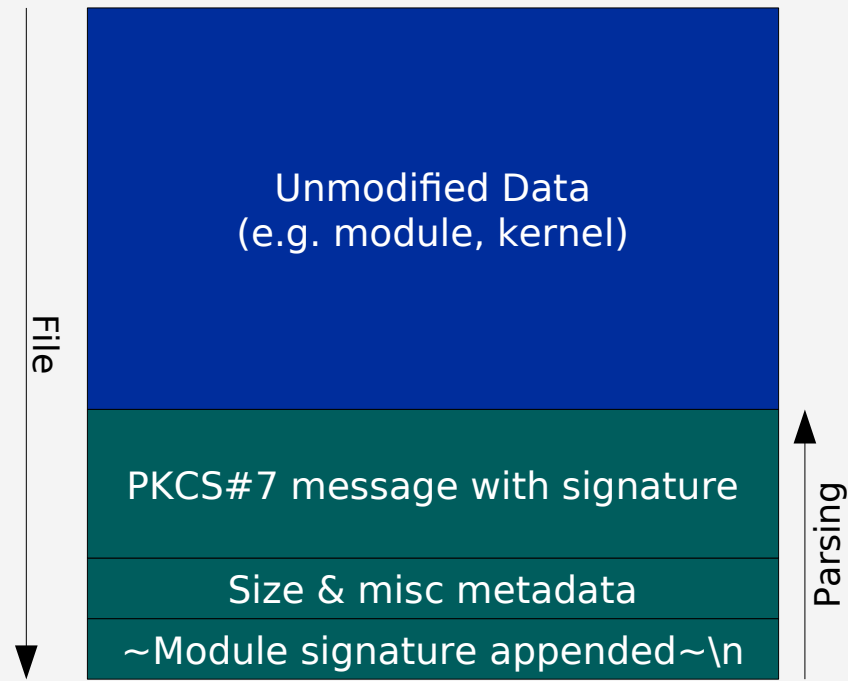
Inactive-Wind-up

IEEE 1275-1994 - IEEE Standard (Configuration) Firmware Con- Practices

BUY THIS STANDARD

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.



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.

```
70     grub_uint8_t *buf = NULL;
71     grub_off_t file_size = grub_file_size (f), total_read_size;
72     grub_ssize_t read_size;
73     grub_err_t err;
74
75     if (file_size == GRUB_FILE_SIZE_UNKNOWN)
76     {
77         return grub_error (GRUB_ERR_BAD_ARGUMENT,
78                             "Cannot parse a certificate file");
79     }
80
81     buf = grub_zalloc (file_size);
82     if (!buf)
83     {
84         return grub_error (GRUB_ERR_OUT_OF_MEMORY,
85                             "Could not allocate buffer for certificate");
86     }
87
88     while (total_read_size < file_size)
89     {
90         read_size =
91             grub_file_read (f, &buf[total_read_size],
92                             file_size - total_read_size);
93         if (read_size < 0)
94         {
95             err =
96                 grub_error (GRUB_ERR_READ_ERROR,
97                             "Error reading certificate file");
98             return err;
99         }
100         total_read_size += read_size;
101     }
102 }
```

Demo time

Booting

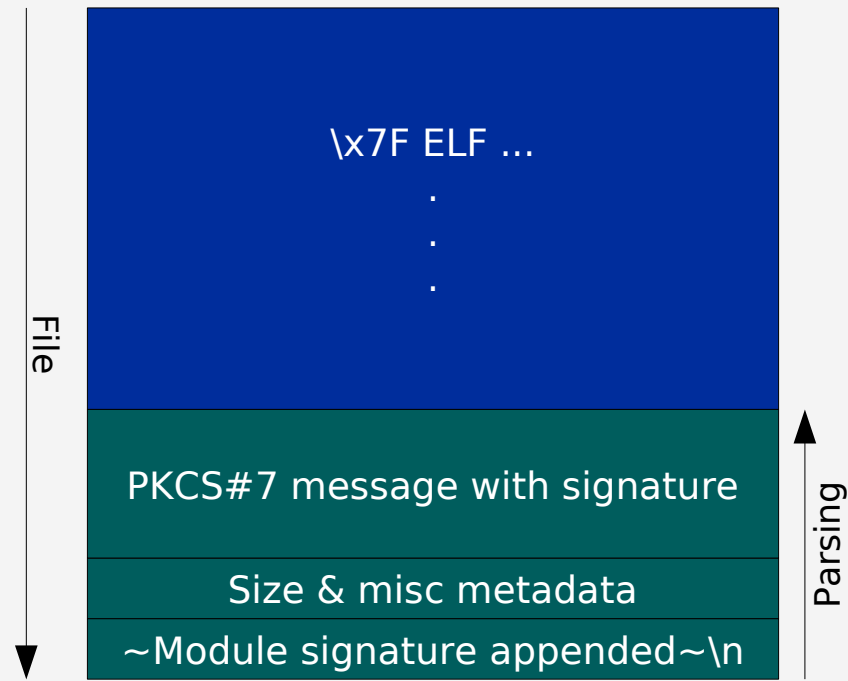
<https://asciinema.org/a/kgDMBoZx9pNqGzkhZ70kkosVz>

Commands

<https://asciinema.org/a/kqhuUA70CuETzI4RDkkL1hXf7>

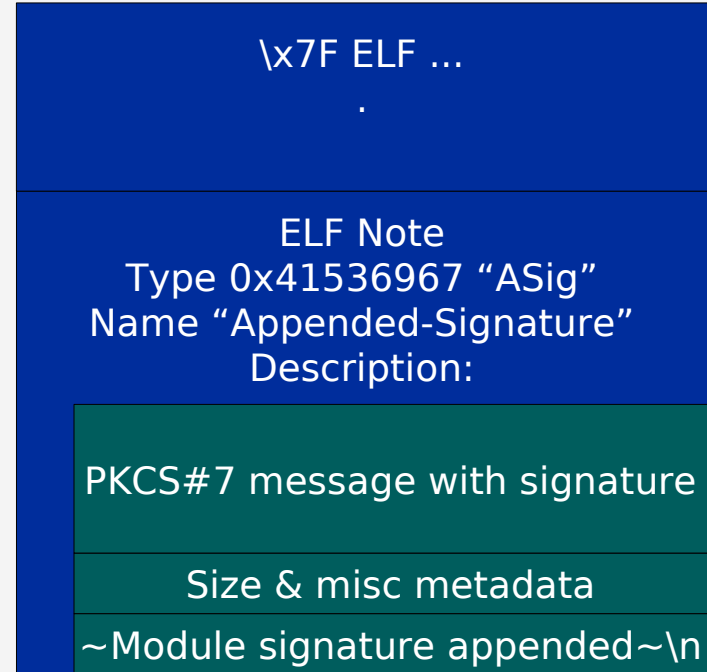
Verifying Grub from Firmware

- **Must be backwards compatible!**
- If we use an appended signature, how does FW know if it is present?
 - Grub loaded from PReP partition: no file system, no file size, no clear EOF.
 - Cannot 'keep scanning': you could hit an old signature.
 - Length of signature varies depends on signing key and hash type.



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.



Signing flow

Sign empty file to determine signature size.

Use new grub-mkimage parameter to reserve space for the signature.

Generates a valid ELF file with a note but the description is all zeros.

Truncate the excess zeros away.

Sign with usual tools.

Upstream status

Verification of Grub from Firmware with Appended Signature ELF note:

Already on list:

<https://lists.gnu.org/archive/html/grub-devel/2020-08/msg00037.html>

Note specification will be submitted for possible inclusion in forthcoming Power Architecture Platform Reference.

Verification of Linux from Grub:

Internal reviews and cleanup underway

Thank you

Daniel Axtens
Linux Security Engineer

—

daniel.axtens1@ibm.com
dja@axtens.net

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

© Copyright IBM Corporation 2020. All rights reserved. The information contained in these materials is provided for informational purposes only, and is provided AS IS without warranty of any kind, express or implied. Any statement of direction represents IBM's current intent, is subject to change or withdrawal, and represent only goals and objectives. IBM, the IBM logo, and ibm.com are trademarks of IBM Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available at [Copyright and trademark information](#).

