Securing Secure Boot on Xen

Ross Lagerwall

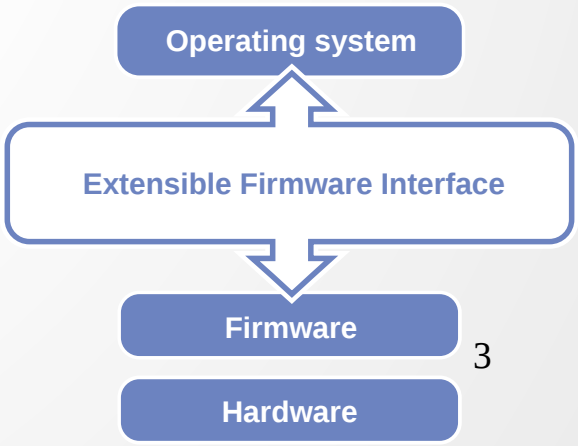
Software Engineer, Citrix Systems

Why Secure Boot?

- How can we prevent running malware at boot?
- With Secure Boot!
- What if the machine is a VM in the cloud?
- Xen doesn‘t support Secure Boot yet – let´s see how it can be done.

Background

- UEFI is a replacement for the BIOS
- It defines how operating systems interact with firmware including how the OS is started
- Secure Boot is part of the UEFI specification from 2.3.1 Errata C
- Using Secure Boot on Xen requires booting the guest as a UEFI guest
- Support for this exists using OVMF



Images: Tiancocore (https://github.com/tianocore/edk2/blob/master/MdeModulePkg/Logo/Logo.bmp)
UEFI (https://en.wikipedia.org/wiki/Unified_Extensible_Firmware_Interface)

Background

- Starting an OS under UEFI works differently compared with BIOS
- The firmware knows how to load a specified file on a FAT-formatted filesystem – could be a bootloader or an OS kernel.
- With Secure Boot, there is an extra step:

Before executing the file its signature is verified

Background

- How is it verified?
- Hardware has NVRAM which stores UEFI variables and these include certificates databases (called PK, KEK, db)
- The kernel to be booted needs to be signed by one of the trusted certificates
- If malware patches or replaces the kernel the firmware will refuse to start it because it fails the signature check

Background

- How are the certificate databases populated?
- Prepopulated at the factory
- They are UEFI authenticated variables which require updates to be signed by the certificates already stored there
- Alternatively a platform-specific reset method can be used to clear the databases

Implementation on real hardware

- The code which implements SetVariable() (i.e. variable services) needs to be protected from malware otherwise Secure Boot is compromised
- But malware could be running at the highest privilege level!
- An extra level of privilege is needed
- System Management Mode!
- Put the code in SMRAM so that it cannot be accessed outside of SMM
- Configure the NVRAM so that it can only be written from SMM
- Setting variables requires trapping into SMM

Implementation on KVM

- KVM virtualizes what real hardware does
- QEMU emulates a block of flash memory for the NVRAM
- KVM emulates SMM for guests
- Reuses existing code for the security-critical parts of variables services

See Securing secure boot with System Management Mode, Paolo Bonzini, KVM Forum 2015 for more details.

Implementation on Xen

- There are numerous vulnerabilities and attacks against SMM
- Xen has no support for emulating SMM
- Using emulated flash limits flexibility of variable storage
- How can we do better?

Implementation on Xen

- With virtualization, there are already two distinct privilege levels (guest and hypervisor) so SMM is not needed
- Run a daemon that implements variable services outside of the guest execution context
- Add a new module to OVMF that implements variables services by proxying to the daemon running outside the guest

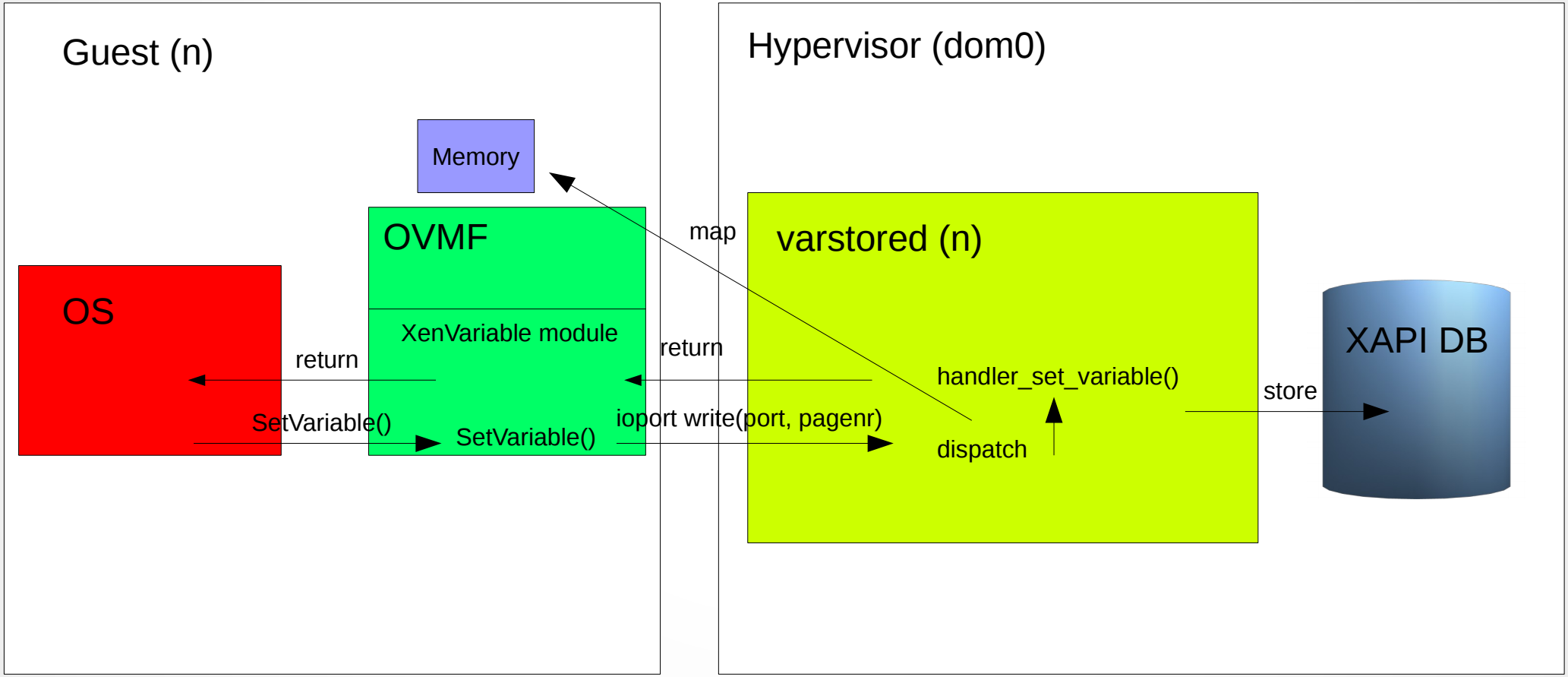
Implementation on Xen

- The guest has no direct access to the code that executes the variables services
- Nor does it have direct access to the variable storage
- Variable storage is abstracted so that different backends can be used:

Berkeley DB, SQLite, MySQL,

XAPI database, flat files, etc.

Implementation on Xen



Implementation on Xen

- varstored: A daemon that implements all the required variable services using the XAPI database for storage
- XenVariable: An OVMF module that proxies variable service calls to varstored
- It works with Xen to securely implement Secure Boot and boots Linux and Windows guests
- Could be used with KVM without much difficulty
- It is not a platform specific implementation unlike the other approaches

Demo

- Video at https://rossl.org/fosdem2019.mkv

When can I use it?

- Not yet publicly available – will be released shortly and announced on the Xen mailing list