Making Kexec/Kdump Work With Secureboot

Vivek Goyal
Red Hat

Problem

- Secureboot says no unsigned code at privileged level 0
- Kexec runs unsigned code at privilege level 0
  - New Kernel
  - Binary blob called purgatory

Proposed Solution

- Pull /sbin/kexec into kernel
  - It is almost like re-writing /sbin/kexec
  - Maintain old interface and deprecate over a period of time.
- Extend secureboot trust chain to user space
  - Sign /sbin/kexec and verify signature at exec() time
  - Let /sbin/kexec verify signature of bzImage
  - Purgatory is part of /sbin/kexec. /sbin/kexec signature verification allows us to trust purgatory.

Requirements

- Not all of the user space is signed
- Protect signed /sbin/kexec from other applications
  - Disable ptrace()
  - Lock down executable in memory (No swap)
- Statically linked /sbin/kexec
  - Shared libraries are not signed
  - Mapped shared library can be overwritten

Signature Verification

- Use IMA signatures
- Signatures stored in security.ima xattr
- Put memory locking information in security.ima
- During exec() in ELF loader
  - Verify signature using bprm hook
  - Map elf sections with memory locked
  - Verify signature again using a new post load hook
  - Store signed process info in cred. Used to deny ptrace() from unsigned processes.

A new option for keyctl()

- A kernel mechanism to verify signature of new bzImage to be loaded
  - Can't trust user space unsigned crypto libraries
  - Trusted system keys are in kernel keyring
  - New option KEYCTL_VERIFY_SIGNATURE
  - Takes user buffer and signature as input
  - Calls IMA functions to verify signature of user passed buffers.

Kexec and EFI run time

- Kexec has been booting new kernel with EFI disabled.
- Kdump need to make EFI run time calls to import db keys into system_keyring
- Other kexec users want proper support on EFI
- Borislav Petkov (Suse) is working on using fixed virtual addresses (-4G) for EFI mappings.

That's It