Implementing UEFI Secure Boot in Qubes OS

Challenges and Future Steps

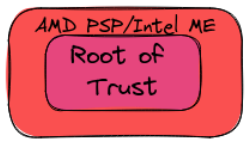
Piotr Król

Kudos

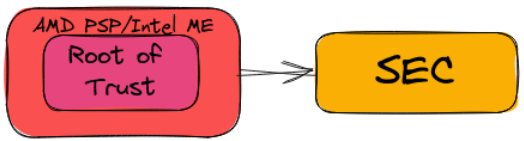
- Kamil Aronowski
- Marek Marczykowski-Górecki
- Andrew Cooper

Goals

- Briefly explain what is UEFI Secure Boot and what led to OSS dislike of it.
- Justify that only valid way of use of UEFI Secure Boot was defined by Trammel Hudson in safeboot project.
  - Anything else are just excuses, which waste resources.



CoT - Chain of Trust
IBV - Independent BIOS Vendor
IFV - Independent Firmware Vendor
SEC/PEI/DXE/BSD - UEFI boot phases

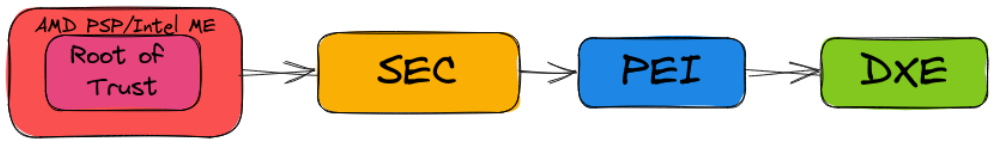


(Measure,) Verify, Execute

CoT - Chain of Trust
IBV - Independent BIOS Vendor
IFV - Independent Firmware Vendor
SEC/PEI/DXE/BSD - UEFI boot phases

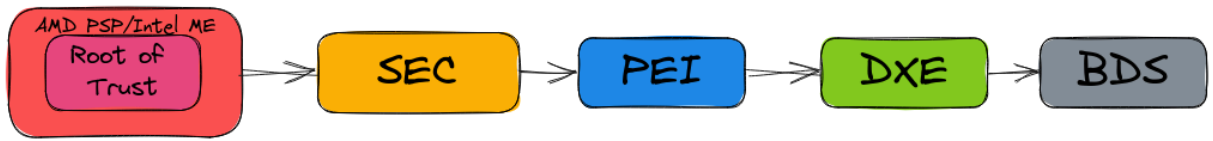
(Measure,) Verify, Execute

CoT - Chain of Trust
IBV - Independent BIOS Vendor
IFV - Independent Firmware Vendor
SEC/PEI/DXE/BSD - UEFI boot phases



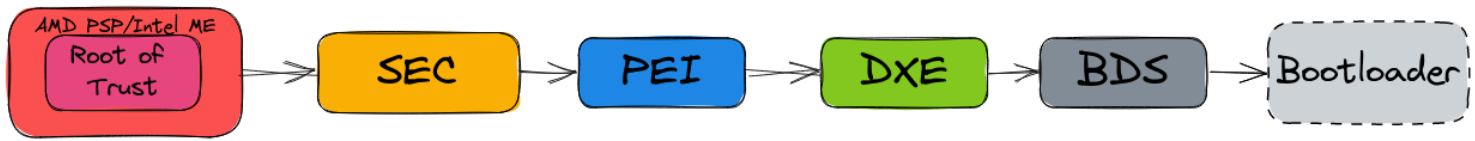
(Measure,) Verify, Execute

CoT - Chain of Trust
IBV - Independent BIOS Vendor
IFV - Independent Firmware Vendor
SEC/PEI/DXE/BSD - UEFI boot phases



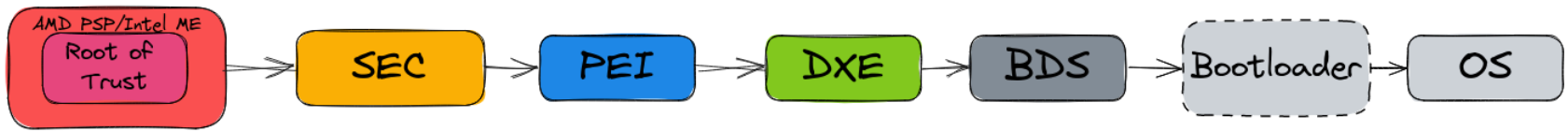
(Measure,) Verify, Execute

CoT - Chain of Trust
IBV - Independent BIOS Vendor
IFV - Independent Firmware Vendor
SEC/PEI/DXE/BSD - UEFI boot phases



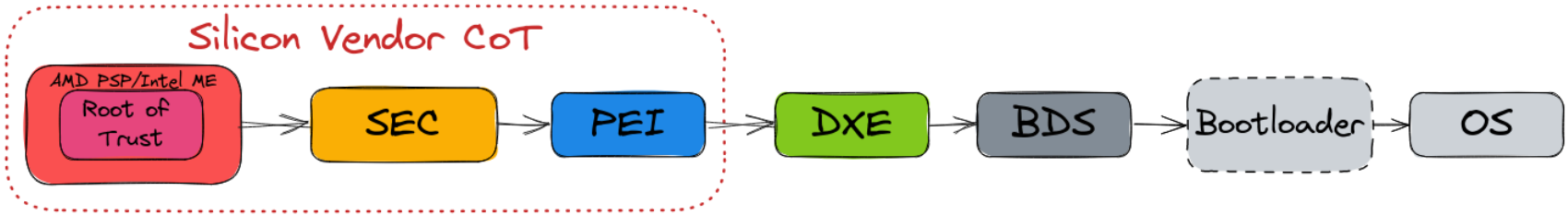
(Measure,) Verify, Execute

CoT - Chain of Trust
IBV - Independent BIOS Vendor
IFV - Independent Firmware Vendor
SEC/PEI/DXE/BSD - UEFI boot phases



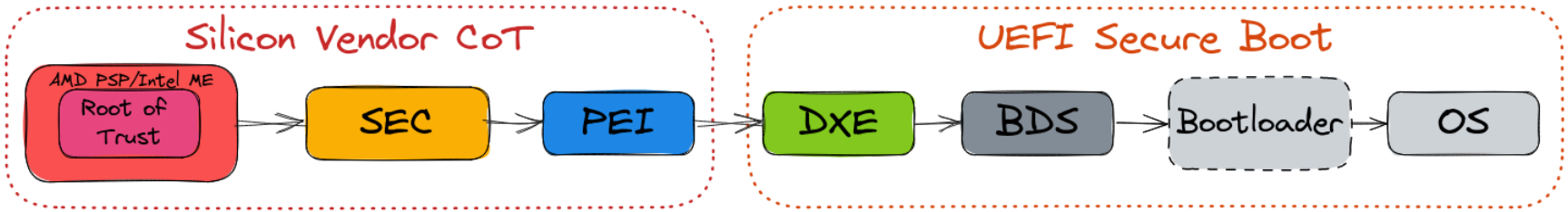
(Measure,) Verify, Execute

CoT - Chain of Trust
IBV - Independent BIOS Vendor
IFV - Independent Firmware Vendor
SEC/PEI/DXE/BSD - UEFI boot phases



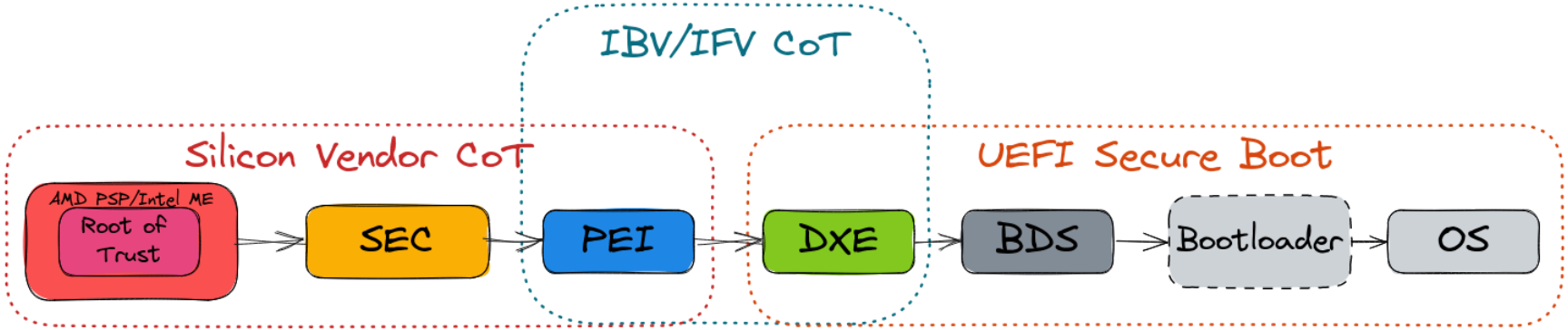
(Measure,) Verify, Execute

CoT - Chain of Trust
IBV - Independent BIOS Vendor
IFV - Independent Firmware Vendor
SEC/PEI/DXE/BSD - UEFI boot phases



(Measure,) Verify, Execute

CoT - Chain of Trust
IBV - Independent BIOS Vendor
IFV - Independent Firmware Vendor
SEC/PEI/DXE/BSD - UEFI boot phases



(Measure,) Verify, Execute

CoT - Chain of Trust
IBV - Independent BIOS Vendor
IFV - Independent Firmware Vendor
SEC/PEI/DXE/BSD - UEFI boot phases

```
switch (GetImageType (File)) {
  case IMAGE_FROM_FV:
    Policy = ALWAYS_EXECUTE;
    break;

  case IMAGE_FROM_OPTION_ROM:
    Policy = PcdGet32 (PcdOptionRomImageVerificationPolicy);
    break;

  case IMAGE_FROM_REMOVABLE_MEDIA:
    Policy = PcdGet32 (PcdRemovableMediaImageVerificationPolicy);
    break;

  case IMAGE_FROM_FIXED_MEDIA:
    Policy = PcdGet32 (PcdFixedMediaImageVerificationPolicy);
    break;

  default:
    Policy = DENY_EXECUTE_ON_SECURITY_VIOLATION;
    break;
}
```

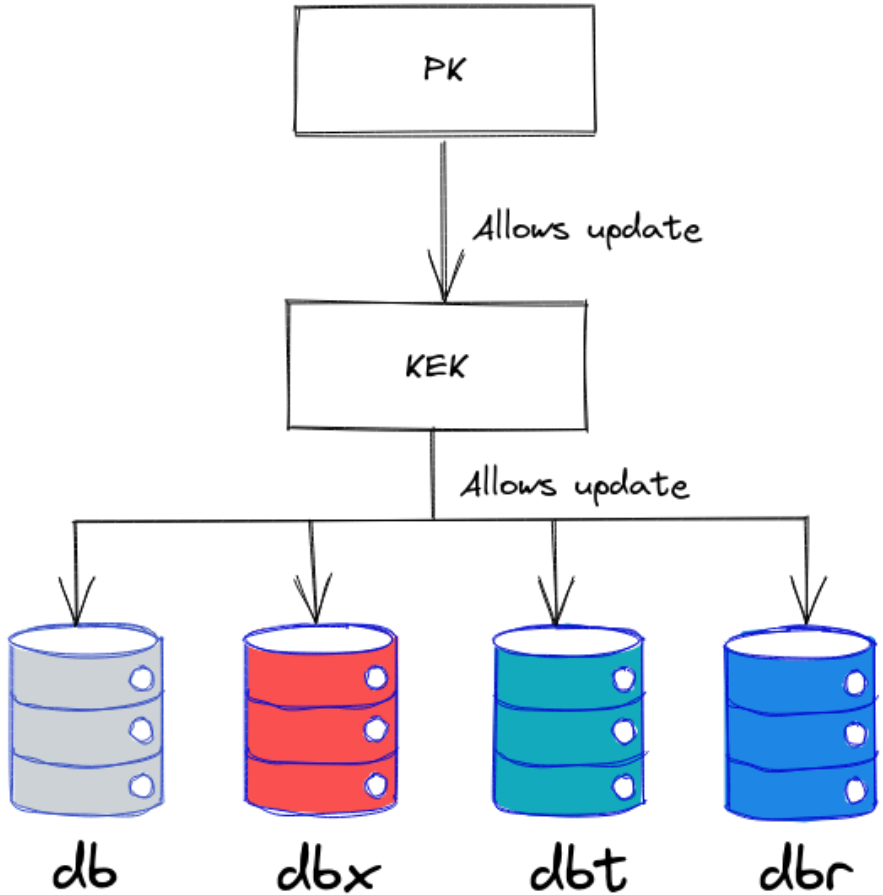
Policies:
`ALWAYS_EXECUTE`, `NEVER_EXECUTE`, `ALLOW_EXECUTE_ON_SECURITY_VIOLATION`, `DEFER_EXECUTE_ON_SECURITY`
`QUERY_USER_EXECUTE_ON_SECURITY_VIOLATION`.

UEFI Secure Boot

UEFI Secure Boot has emerged as a critical feature to protect systems against persistent firmware attacks and unauthorized code execution. While Qubes OS is renowned for its security-centric approach, the official support for UEFI Secure Boot remains a significant milestone yet to be fully realized.

UEFI Secure Boot is really Chain of Trust technology, which covers very specific part of x86 UEFI compliant boot process.

By default it covers only transition between UEFI BIOS and OS Loader/OS Kernel. Verification of components coming from Firmware Volume, OptionROMs, drivers from external media or internally connected are verified only if correct policy is set by UEFI BIOS vendor.



UEFI Secure Boot and Linux

- Microsoft became a holder of of keys for UEFI CA
  - nobody else had infrastructure (Microsoft already signed OS drivers),
  - nobody else had/wanted to put resources in the process,
- Microsoft to provide certification mandates among OEM inclusion of one their certificates in KEK database and two in DB (MSFT PCA, UEFI CA).
- UEFI CA is the certificate which signs non-Microsoft software.

UEFI Secure Boot has bad press

- PKfail (CVE-2024-8105)
  - Use of insecure PK generated by AMI for reference implmentation needs
- UEFI Shell considered harmful (CVE-2024-7756)
  - If UEFI BIOS has UEFI Shell and attacker would be able to trigger it, then it can be leveraged for UEFI Secure Boot bypass thanks to `mm` tool (CWE-489: Active Debug Code).
- BootHole
  - Multiple vulnerabilities in GRUB2 allowing to bypass UEFI Secure Boot.
  - Lack of revocation in shim lead to escalation of revocation to certificate from Microsoft to prevent shims that allow running vulnerable GRUB2.
- Black Lotus
  - Bootkit using CVE-2022-21894 ("Baton Drop") to bypass UEFI Secure Boot.
  - > Windows Boot Applications allow the `truncatememory` setting to remove blocks of memory containing "persistent" ranges of serialised data from the memory map, leading to Secure Boot bypass.

The cure happen to be worse than the disease

- Microsoft effort for maintaining security properties of UEFI Secure Boot on certified hardware lead to requirements, which leads to many issues in OSS ecosystem. If it all was not bad design from the beginning.
  - SBAT (*Secure Boot Advanced Targeting*) was created to create revocation mechanism in shim,
  - All processes around signing are very bureaucratic.
  - shim community put together dedicated review repo and guidelines to simplify process of OSS community.
- Need for complex and non-standarized integration with HSMs.
- But we can see almost no one attack UEFI Secure Boot mechanism, not even implementation, but rather configuration or already authorized code.
  - Maybe we will get there.

UEFI Secure Boot and Qubes OS

- Those are requirements for being signed by Microsoft, not for UEFI Secure Boot a la'safeboot.
- First production-related question coming from Wim Vervoorn (Eltan), Aug 2017.
- Effort started to be tracked in #4371 since Oct 2018.
- Trammell reported booting Qubes OS 4.1 with secure boot enable leveraginging unified `xen.efi` in Aug, 2020.
  - initial concerns: xen.efi reliability while loading other boot files, complexity of boot parameters modification
- Effort for signing boot images was requested by Marek in May 2023 in #8206.
- Qubes OS Summit 2023 we discussed inclusion of Qubes OS CA in Dasharo firmware.
- Frederic works on improving Qubes Builder v2 to include `pesign` for signing needs PR #77.
- There is no easy way to use current installer (R4.2.3) and modify it for UEFI Secure Boot enabled booting.
- Current approach is to build unified `xen.efi` and sign it. Pre-built `xen.efi` binaries will be provided in R4.3.

UEFI Secure Boot and Xen

- There is a lot in this can of worms.
- From discussion with Andrew:
  - Xen needs support for SBAT, which is not present yet.
  - NX_COMPAT is required for SBAT support.
  - kexec have to leverage SHA256 for checking integrity of executed images.
  - kexec purgatory code should be built-in Xen.
  - Livepatching have to check patches singatures.
  - Command line have to be correctly handled (some options may be not safe).
  - New hypercalls ABI for checking all passed pointers.
  - Some debug options should have ability to be disabled at compilations time: GDB stub.
- In summary "unsigned code can’t make any unaudited reads/writes".
- Not everything affect Qubes OS.
  - But ensuring Linux kernel lockdown mode is forced will belong to this tasks.

What we can do right now?

- Choose hardware which has better UEFI Secure Boot Support
  - HP and some ACER BIOS implement ability to trust given file for execution,
  - not ideal but could be improved, especially by Dasharo Team, of course if there is consensus about security of such solution with Qubes OS Team,
- Wait for R4.3.
- Compile unified kernel yourself and try safeboot-like approach - this is what we will try.
- Disable UEFI Secure Boot

It would not be end

- What are the capabilities which my system support?
- UEFI specificaition not defines what is supported,
- every implementation may do things little bit different,
- how to test that?
  - try
  - trust spec version compliance, which say something but not everything,

Hackathon Challange

- Build unified xen.efi using qubes-builderv2
  - fix documentation?
- Test in OVMF

Q&A

Backup

What could be done differently?

- Improve reference UEFI BIOS implementation to make dealing with UEFI Secure Boot manageable by mare mortals.
- Provide system level tooling as part of OS installation process.
  - Many distro doing that since those cannot afford fullfilling requirements for review process.