Challenges in developing trustworthy Linux-based systems in an open-source way

Piotr Król

Kudos for review and feedback

- Ross Philipson
- Daniel P. Smith
- Andrew Cooper
- Richard Pesaud

Glossary

- LVFS - Linux Vendor Firmware Service
- HSI - Host Security ID
- HCL - Hardware Compatibility List
- FSF RYF - Free Software Foundation, Respect Your Freedom
- SMM - System Management Mode
- TEE - Trusted Execution Environment
- AMD PSP/ASP - AMD Platform Security Processor/AMD Security Processor
- Intel PPAM - Intel Platform Properties Assessment Module
- TCG DICE - Trusted Computing Group, Device Identifier Engine
- UEFI - Unified Extensible Firmware Interface
- OCP - Open Compute Project
- SNIA - Storage Networking Industry Association
- DMTF SPDM - Distributed Management Task Force, Security Protocol and Data Model

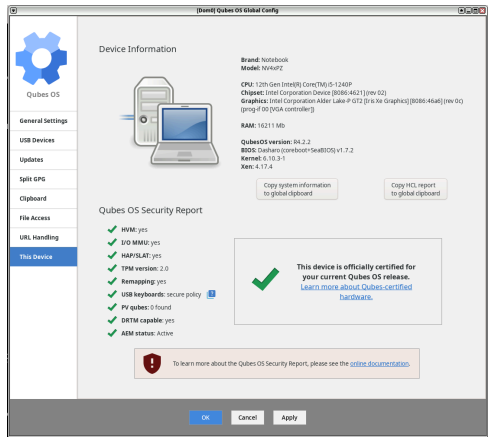
Goal

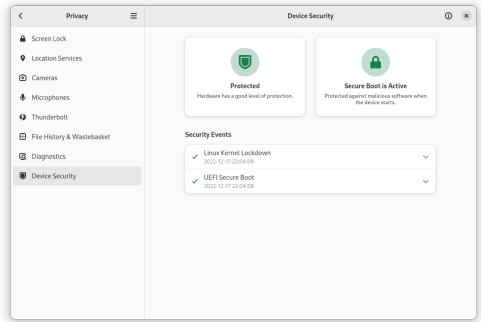
The presentation highlights five challenging areas and activities related to Linux-based system booting and security from platform vendor and user perspective.

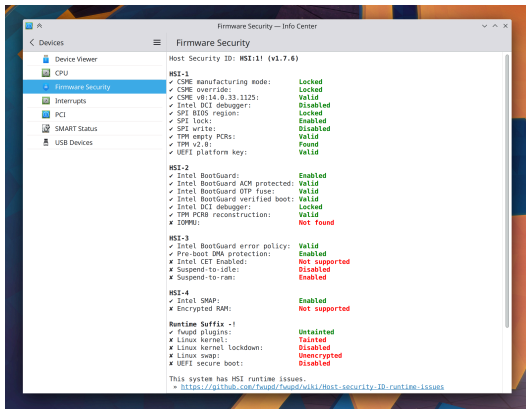
The focus is on:

- Highlighting the problems to provide inspiration and work as an anchor for future CfP processes. We see challenges and issues in the open source communities in various areas, including only sometimes being able to get domain experts in these topics.
- Make valuable references to past events.
- Get feedback on what is missing.

Challenge 1: Assessment







- Lack of OS awareness about hardware security capabilities leads to the inability to evaluate and improve system security posture.
- Over the last two years, we have seen number of various communities approaching this topic.
  - GNOME introduced the Device Security Settings page (based on LVFS HSI).
  - KDE Plasma added Firmware Security page (based on LVFS HSI).
  - Qubes OS Security Report (based on Qubes HCL).
  - FSF RYF Certification (unclear rules).

Assessment

- Some interfaces for CPU assessment already exist. We have tools like:
  - `lscpu`, which shows `/sys/devices/system/cpu/vulnerabilities`.
  - `/proc/cpuinfo` bugs.
- We could only secure our ecosystem if we knew what security mechanisms are available on a given piece of hardware.
- Shipping hardware with Linux would be more challenging without exposing the interface, which will help assess security configuration and help protect against misconfiguration of security features.
- Achieving results requires broader cooperation between communities and vendors.
- Followed presentations and communities discussed this topic:
  - Embedded Recipes: LVFS: The next 50 million firmware updates.
  - TrenchBoot Summit: LVFS Host Security ID (HSI) and Silicon-Based Core Security.
  - Arm SystemArchAC Security Sub-team.

Challenge 2: SMM (TEE?) as Chain of Trust Gap

- For various reasons, vendors add secure areas that try to guarantee the confidentiality and integrity of the code and data inside it.
- This creates an exciting attack target that has been abused many times in the past.
- Solutions:
  - SMI Transfer Monitor - Linux driver out of tree, probably far from making it first-class citizen.
  - Project Mu - has an SMM Supervisor integrated on top of STM API. Microsoft Secured Core PC makes it a requirement, as does other SMM threat-mitigating technology.
  - Ron Minnich: Let's move SMM out of firmware and into the kernel.
- Followed presentations and communities discussed this topic:
  - BSides Portland: ABC to XYZ of Writing System Management Mode (SMM) Drivers.
  - Dasharo Dev vPub: Overview of the Intel SMI Transfer Monitor on Dasharo Firmware.
- SMM (TEE?), by definition, should not give direct access to the user, but it needs to communicate the platform security state to the user. Is there a place for it in the Linux kernel?

Challenge 3: All your hardware belong to us

- The growth of hardware and firmware components like AMD PSP features, Intel PPAM and ME features, or MS Pluton, and how those effectively block/enable building trustworthy systems by the open-source community.
- In parallel, this creates an ecosystem in which we cannot leverage the the full potential of hardware and firmware in our machines.
- On one side, this is an attack vector that we cannot do anything about. On the other side, our hardware no longer belongs to us if anyone is surprised by that fact.
- Followed presentations and communities discussed this topic:
  - BlackHat: Breaking Firmware Trust From Pre-EFI: Exploiting Early Boot Phases
  - SMM isolation - SMI deprivileging (ISRD)
- Why did it all happen?
  - USENIX ATC '21/OSDI '21 Joint Keynote Address-It's Time for Operating Systems to Rediscover Hardware

Challenge 4: Root of Trust

- There are promising plans for defeating the lack of consistent assessment, implementation, and provisioning of Root of Trust on very different hardware configurations through Caliptra, DICE, SPDM, and more, as well as what impact it may have on the OS.
- Some SPDM-related patches are slowly landing in Linux.
- Followed presentations and communities discussed this topic:
  - UEFI Forum: Using SPDM in UEFI for Device Attestation.
  - OCP: OCP Attestation using SPDM and DICE.
  - OCP: TPM Transport Security: Defeating Active Interposers with DICE.
  - OCP: And Update on Caliptra.
  - SNIA: TCG DICE & DMTF SPDM Binding Overview.

Challenge 5: What we can do better with DRTM for AMD

- Lessons learned from the 11-th series of making 20+ year-old security tech for Intel CPUs first-class citizens in the Linux kernel.
- Make sure to consider platform-specific challenges and architectural alignment (Intel vs AMD, ARM inclusion). Design a robust ABI framework up front to reflect this.
- Consider configuration impacts and document them upfront (e.g., KASLR, IOMMU in the Linux kernel).
- Check various compilers and build environments from the beginning. Ensure successful builds with the new features turned on and off in the configuration.

Challenge 5: What we can do better with DRTM for AMD

- Plan for all entry points and boot protocols (32/64 bit, legacy/UEFI). Note that the upstream work assumes 64b environments to run in, but it should build in all environments
- Be willing to accommodate feedback and suggestions where possible and attempt to get assistance from the community (e.g., WAIT/MONITOR, linker-based `kernel_info` placement).
- Avoid tight coupling to specific hardware behaviors (e.g., TPM access).
- Is there anything missing?

Q&A

piotr.krol@3mdeb.com