How TrenchBoot is enabling Measured Launch for Open-Source Platform Security

Daniel Smith, Chief Technologist, Apertus Solutions, LLC

Agenda

- Why TrenchBoot
- Relevance of Dynamic Launch and Xen
- Background and security of Dynamic Launch
- Background and goals of TrenchBoot
- Initial Capability under development
- What is next that TrenchBoot is building for Xen
- Wrap up

Why

- **Launch integrity is the foundation for platform security**
  - If the hypervisor was corrupt from boot, how could you trust any VM integrity/introspection capabilities that it may be running?
  - Was the correct hypervisor and VMs with the necessary security and safety features loaded at launch?
  - It deserves the attention needed to get it right and well integrated
- **Dynamic Launch has been under utilized**
  - Can be initiated many times between power-on and power-off
  - Each Dynamic Launch is an opportunity to establish the current integrity of the platform
- **Evolving hardware categories for Launch Integrity**
  - Root of Trust (discrete TPM, ME PTT, PSP fTPM)
  - Secure Coprocessors (ME, PSP, T2, Nitro)
  - Boot SPI interposers (OpenTitan, Cerberus, SureStart)
  - Hybrid (AzureSphere MCU, Arm Corstone-700 M-class Secure Enclave)

Dynamic Launch and Xen

- **TrenchBoot was born out of limitations of using tboot to launch Xen for OpenXT project**
  - Access to the TXT TPM event log is blocked
  - Conflict over access to UEFI Boot Services
  - Can only measure Multiboot modules that were loaded into memory by bootloader
  - Only one attestation action: predetermined PCR manifest verification
  - Only supports Intel TXT, no love for AMD's Secure Startup
- **Upstream Xen needs a good Dynamic Launch story for hardware-rooted integrity**
  - Google has Shielded VMs
  - Microsoft has System Guard Runtime Attestation
  - VMWare has ESXi Host Attestation Status
  - Xen has tboot (sort of … see above)

Terminology

Mapping concepts to specification and vendor terms

Description	TCG	Intel TXT	AMD-V
A component that must always behave in the expected manner because its misbehavior cannot be detected.	Root of Trust (RoT)		
Process of starting a software environment at an arbitrary time in the runtime of a system	Dynamic Launch (DL)	Late Launch	Secure Startup
Platform dependent event that triggers the DL	DL Event	GETSEC[SENTER]	SKINIT
Performs initial configuration actions that are platform specific before invoking the D-RTM CPU instruction	D-RTM Configuration Environment (DCE) Preamble		
Software/firmware that executes from the instantiation of the DL Event to the transfer of control to the DLME	D-RTM Configuration Environment (DCE)	Authenticated Code Module (ACM)	Secure Loader (SL)
Software executed after the DCE instantiated TCB is established	Dynamically Launched Measured Environment (DLME)	Measured Launch Environment (MLE)	Security Kernel (SK)

Setup of a Dynamic Launch

- **The system must be in a very specific, quiescent state to launch**
  - Intel
    - TPM with all localities closed
    - Protected mode without paging and SMX enabled
    - ACM loaded and TXT Heap configured
    - MLE loaded below 4GB and compliant page table setup
    - Machine check clear
    - APs are rendezvous and necessary state preserved
  - AMD
    - Protected mode without paging
    - Secure Loader loaded
    - Machine check clear
    - APs are rendezvous
  - Qualcomm
    - TBD: Rumored upcoming DRTM IP core
- **This enables getting to a known good state without breaking everything**

Result of a Dynamic Launch

- **Provides a very controlled and protected startup**
  - The CPU obtains Locality 4 on the TPM and clears DRTM PCRs (17-22)
  - All CPU interrupts (NMI, SMI, INIT, etc) are disabled
  - The CPU protects the DCE from DMA access
    - Intel uses Cache as RAM (CRAM)
    - AMD uses Device Exclusion Vector (DEV)
  - The DCE is measured by the CPU and stored in PCR 17 of the TPM before execution
    - On Intel the ACM is authenticated before measurement
    - On AMD the Secure Loader is owner provided
  - The DCE ensures the DLME is DMA protected, measures, and then executes
- **The results is a very high integrity assertion of the DLME**
  - Removes boot firmware from the TCB

TrenchBoot

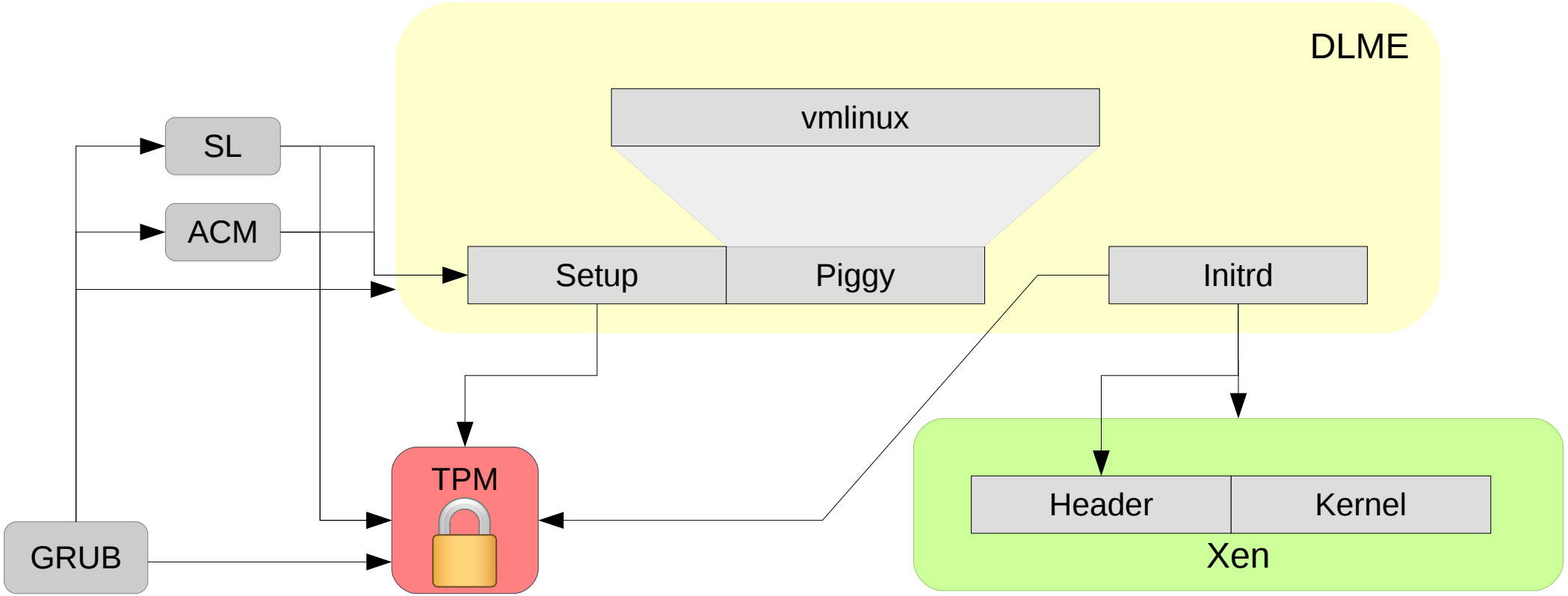
- **TrenchBoot is a cross-community integration project focused on launch integrity**
  - There is no “one thing” that is TrenchBoot
  - The purpose is to develop a common, unified approach to building trust in the platform through launch integrity
  - And to work with existing Open Source ecosystem to integrate the approach into their respective projects
    - This means there can now be a unified Dynamic Launch approach between Xen, KVM, other Open Source hypervisors, and potentially proprietary hypervisors.
- **The TrenchBoot approach provides for different strategies to build trust in the platform**
  - First Launch inspection – Establishing hardware rooted integrity during platform boot
  - Runtime inspection – Establishing hardware rooted integrity during platform runtime
    - SecureLaunch kexec for Linux
    - Runtime Xen verification
    - Re-establishing platform state after sleep or hibernate
  - Update/Shutdown inspection – Reviewing platform state before platform reboot/shutdown
    - Useful for checking integrity before persisting state to disk

Who is contributing to TrenchBoot

First Launch Inspection

- **The initial implementation being worked to demonstrate a common use case**
- **This is the traditional approach that uses Dynamic Launch to root the target kernel in hardware**
- **TrenchBoot approach expands the traditional approach**
  - Leverages Linux existing UEFI support to handle EBS hand-off
  - Provides a more flexibility means for measuring the environment
  - Leverages Linux kexec interface for launching subsequent kernel
  - On Intel platforms, SEXIT is called to close access to DRTM PCRs
- **Implications for Xen**
  - Will require Xen kexec entry to function with post EBS on UEFI
  - Will enable removal of tboot code

Basic Flow of First Launch



And then comes runtime inspection

- **That's right, I want to relaunch a running Xen without rebooting!**
- **Why would I want to do something this crazy?**
  - It is actually quite logical
  - A lot of work was done to make sure the right kernel is launched but that guarantee really ended after communication with the outside world began
  - Consider how often a system reboot occurs to establish the integrity of your system?
    - Servers --> rarely
    - Desktop --> occasionally
    - Laptops --> regularly
  - There are actually a few use cases of interest
    - A System Owner/Administrator may want to check a system
    - An OpenXT-like platform may want to check integrity before launching a critical VM
- **Xen could be the first hypervisor that can at any point securely re-establish the integrity of itself!**

How will this work

- **Conceptually the approach will be to,**
  - Bring the system to a quiescent state either by pausing or sleeping (S3) all domains
  - Xen will DL into an integrity kernel
    - This may be setup at boot time or as a special type of domain
    - A protocol will be defined to pass necessary information such as the return address
  - Integrity kernel will inspect/verify in-memory Xen
  - Integrity kernel will record measurements taken and optionally create a signed quote
  - Integrity kernel will then jump back to address passed to Xen
  - Xen will bring the system back to a running state
- **The result will be a hardware rooted runtime inspection of Xen**

Timeline

- **Announced at PSEC'18**
- **Provided an initial briefing of the work on September 2018 Xen Community Call**
- **First working demonstration on Intel desktop system in February 2019**
- **In March began engaging Linux Kernel Mailing List on boot protocol changes**
  - Resulted in the setup_header2 RFC submitted in June
- **Remainder of 19Q3 is the completion of First Launch inspection for Intel/AMD and upstreaming to respective projects**
  - Will be engaging Xen mailing list on the launching Xen via kexec post EBS
- **Upon completion, will begin work on Runtime inspection**
  - Ideate on how to enable Xen to function as a DCE Preamble
  - Architect the hand-over protocol from host to inspection kernel
  - Architect the structure for quote signed evidence
  - Ideally will have a Xen Design Document in 20Q1

Opportunity

- **Improvements that hardware and system manufacturers might want to take to heart**
  - Would like to see AMD added an SKEXIT instruction
    - Intent would be to close access to DRTM PCRs
  - Would like to see AMD provide support for STMs or an STM-like capability
    - The intent is to obtain hardware rooted measurements of SMM
    - This is likely not easy ask and open to alternatives that achieve the intent
  - Would like to see improvements in IOMMUs
    - The intent is to have true and complete device isolation
  - Would like to see OEMs incorporate an STM or an equivalent that enables a hardware rooted measurement of SMM as part of DL
  - Would like to see Device manufactures adopt Intel's PCIe Device Security Enhancements
  - Would like to see ARM and RISC-V provide a late launch instruction
  - Would like to see TPMs become common an ARM boards
    - And leveraging them in the BootROM

References

- **Trust Computing Group Architecture Overview**
  - https://trustedcomputinggroup.org/wp-content/uploads/TCG_1_4_Architecture_Overview.pdf
- **Trusted Computing Goub D-RTM Architecture**
  - https://trustedcomputinggroup.org/wp-content/uploads/TCG_D-RTM_Architecture_v1-0_Published_06172013.pdf
- **Intel TXT Software Developers Guide**
  - https://www.intel.com/content/www/us/en/software-developers/intel-txt-software-development-guide.html
- **AMD64 Architecture Programmer's Manual Volume 2: System Programming**
  - https://www.amd.com/system/files/TechDocs/24593.pdf
- **Inside the Octagon**
  - http://alex-ionescu.com/Publications/OPCDE/octagon.pdf
- **PSEC 2018: TrenchBoot: Unified Approach to Harness Boot Integrity Technologies**
  - https://www.platformsecuritysummit.com/2018/speaker/smith/
- **PCI Express Device Security Enhancements**
  - https://www.intel.com/content/dam/www/public/us/en/documents/reference-guides/pcie-device-security-enhancements.p
- **Arm® Platform Security Architecture Firmware Framework**
  - https://pages.arm.com/rs/312-SAX-488/images/DEN0063-PSA_Firmware_Framework-1.0.0.pdf


Xen
Project
DEVELOPER AND DESIGN
SUMMIT