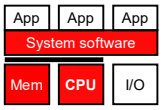
CSE 5600
Computer Systems Architecture

Security

Slides originally developed by Justin Deters (Wash U)

1

This Unit: Security



JumpSwitches for Indirect Branches

Trusted Execution Environment
- ARM
- AMD
- Intel
- RISC-V

2

Branch Target Injection Attack

- Variation on Spectre (called Spectre Variant 2)
- Train branch predictor to speculatively jump to an unauthorized memory address
- Mitigated using retpoline (short for "return trampoline")
  - Instead of indirect JMP instruction
  - Replace with fixed call, jerryrig return address stack, and issue RET call
  - I.e., "bounce" through a return instruction instead of directly executing the indirect jump instruction

3

JumpSwitches Talk

- Retpolines
  - Served as the state-of-the-art defense, effectively disabling speculative execution for indirect branches
  - 20% penalty on some workloads
- JumpSwitches
  - Enables speculative execution of indirect branches on safe targets
  - Leverages indirect call promotion, transforming indirect calls into direct calls
  - Learn targets at runtime and perform just-in-time promotion without the overhead of binary translation
- USENIX ATC '19
- JumpSwitches: Restoring the Performance of Indirect Branches In the Era of Spectre
  - https://www.usenix.org/conference/atc19/presentation/amit

4

Hardware Design Choices

- Speculative Execution → Good
- Data Prefetching → Good
- Speculative Execution + Data Prefetching → Bad
- Design choices can interact with unintended consequences
- Need to design hardware with security in mind
- Modern approach is the use of Trusted Execution Environments (TEEs)

5

Trusted Execution Environment

- Assume that the whole chip is compromised
- Add a Trusted Execution Environment (TEE) to the chip
  - Ensure hardware is not vulnerable to known threats
    - E.g., in-order execution
  - Take measures to diminish chances for software vulnerabilities
    - E.g., keep functionality as low as possible
- Run security sensitive tasks in the TEE
  - Encryption/decryption
  - Storing secure keys
  - Etc.

6

TEE Applications

- Financial services and crypto wallets
- Biometric authentication
- Digital rights management (DRM)
- AI model protection
- IoT device security
  - E.g., secure bootloaders
- Encrypted communications
- Blockchain and smart contracts
- Database operations on sensitive data
  - E.g., patient data

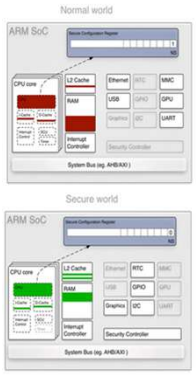
TEEs are common in industry

- ARM → TrustZone
- AMD → Platform Security Processor
- Intel → Trusted Execution Technology & SGX Software Guard Extensions
- RISC-V → Keystone
  - Open source!

ARM: TrustZone

ARM TrustZone

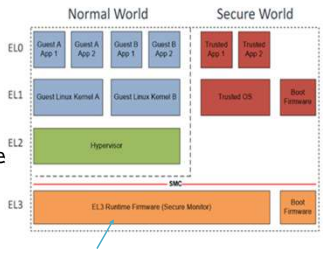
- CPU has a secure mode
  - OS cannot see this mode
  - Firmware responsible for changing modes
  - Normal World vs. Secure World
- Software has different view of hardware in secure world
- Change in security state propagates over system bus



https://genode.org/documentation/articles/trustzone

Example System

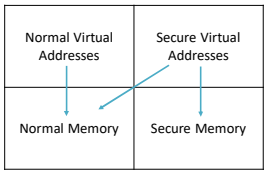
- Software must request to be moved to secure world
  - Used hardware exceptions to initiate this process
  - Firmware is in charge of preserving software state between transitions



https://elinux.org/images/0/05/Elc-tfa.pdf

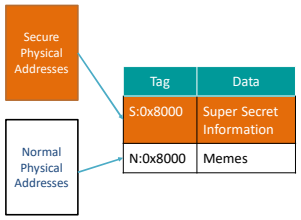
Virtual Memory

- Separate *virtual addresses* for Normal and Secure addresses
- Separate *physical addresses* for Normal and Secure memory
  - Normal Mode can only see Normal memory
  - Secure Mode can see both Normal and Secure memory



Caches

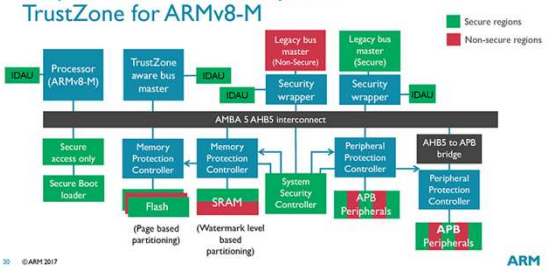
- Caches are tagged with *physical addresses*
- Only secure applications can hit a cache line tagged with a secure physical address



13

13

Simplified software security with TrustZone for ARMv8-M



OK, this is cool, but do I want to see the "complicated" version of this picture?

BTW, IDAU is hardware that knows normal vs. secure address boundaries.

14

14

ARM TrustZone Summary

- Primarily focused on memory partitioning
- Does not replicate processor
  - Instead, adds a new processor "mode" with more privileges
- System bus is critical enforcement mechanism
  - It won't respond to illegal address requests

15

15

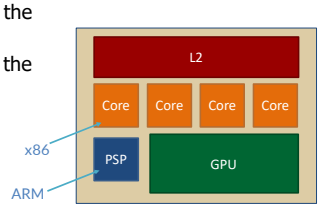
AMD: Platform Security Processor

16

16

AMD Platform Security Processor (PSP)

- Embeds a 32-Bit ARM microcontroller
  - Isolated ROM and SRAM
  - Cryptographic Processor
  - Two different ISAs on the same chip!
- Microcontroller manages the security of the processor
- Communicates with processor via interrupts



17

17

Fun Historical Note

- In March 2018 CTS Labs (a security firm) claimed to have found "major vulnerabilities" in Zen chipset
  - Others said, genuine but not "major"
  - AMD fixed them via firmware updates
- CTS publicized claims after giving AMD less than 24 hours to respond
- Many suspected timing was for stock manipulation

18

18

AMD Platform Security Processor Summary

- Totally separate fetch-execute engine on chip
  - Distinct instruction set (ARM vs. x86)
- PSP has dedicated on-chip memory (and caches)
- PSP can access main memory, but not vice versa
  - I.e., x86 processors cannot see or access PSP memory space

19

19

Intel: Software Guard Extensions

20

20

Intel Software Guard Extensions (SGX)

- More like ARM – focus on memory access
- Create memory regions – called enclaves
  - Limited access
  - Encrypted
- SGX is only one of many Intel technologies aimed at security
  - Intel Converged Security and Management Engine (CSME) evolved from previous management engine
    - Similar to AMD's PSP

21

21

RISC-V: Keystone

22

22

RISC-V Keystone

- TEE for RISC-V
- RISC-V
  - Open source ISA
  - Community gets to decide what goes in
  - Many chip implementations are also open source
  - Some industry adoption
    - Western Digital
    - SiFive
    - Google → Titan M2 in Pixel 6
- Keystone uses ISA features of RISC-V

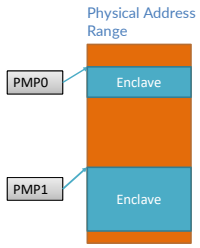
RISC-V: The Free and Open RISC Instruction Set Architecture

23

23

RISC-V Physical Memory Protection

- ISA Feature
  - Configuration of registers done through the ISA
  - Each core has its own set of PMP registers
- Controls User Mode and Supervisor Mode *physical memory* access permissions
- If the *requested physical memory* address is outside the locked range → Denied
  - Can also allow for read, write, and execute granularity of the regions
- Give permission for a *contiguous* address range of a certain size

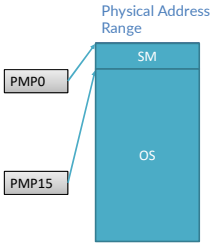


24

24

Setting up Keystone

- PMP registers are statically prioritized
  - 0 → highest priority
  - 15 → lowest priority
- Keystone creates a PMP entry for the Security Manager (SM) at the highest priority
- Keystone creates a PMP entry for the whole address range at the lowest priority
- SM launched for each core in the system

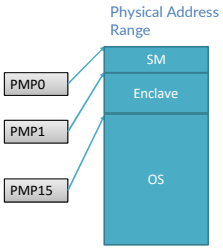


25

25

Creating an Enclave

- OS finds a free contiguous physical memory range → Calls the SM
- SM adds PMP entry
  - Higher priority than OS and user processes
- Enclave regions cannot overlap with each other or the SM

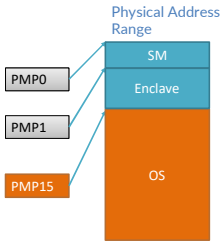


26

26

Control Transfer

- SM enables the permission bits for the enclave PMP entry
- SM removes permission for the OS PMP entry
- Enclaves can only access itself
- Nobody can access the enclave

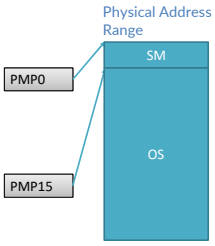


27

27

Destruction

- SM disables all permission for the enclave
- Clears the memory of the enclave
- Gives memory back to OS
- Re-enables the OS PMP entry
- PMP for the enclave is freed
- If the OS cannot interact, how is memory managed?



28

28

Memory Management

- Keystone manages the virtual memory *in the enclave*
- Each enclave has its own page table
- Only the enclave knows its own virtual-to-physical mapping
- The OS only knows what contiguous memory range used

29

29

Can We Trust the Trusted Execution Environment?

30

30

Can We Trust the Trusted Execution Environment?

- Who watches the Watchmen?
- Open source software relies on code audits and the community to find and fix security flaws
- Vast majority of hardware designs are closed source
  - Nobody can audit them except for the companies themselves
- Many security flaws make it to market
  - Many exploits aren't discovered for years
- Bad Actors
  - Zero-Day exploits are sold by private entities to governments
  - Some governments have exploits built into processors
- Perfect security is impossible
  - You can only mitigate security risks