Flicker: Minimal TCB Code Execution

Jonathan M. McCune

Carnegie Mellon University

March 25, 2008

Bryan Parno, Arvind Seshadri

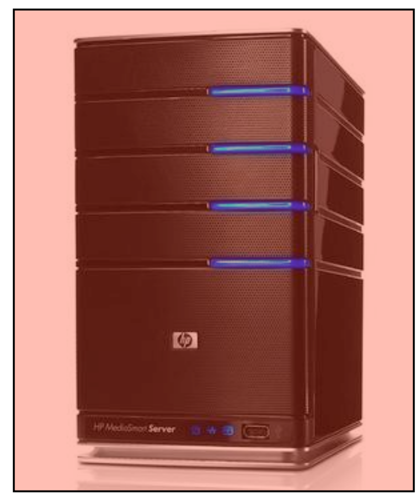
Adrian Perrig, Michael Reiter

Password Reuse

- People often use 1 password for 2+ websites
- Banking, social networking, file sharing, …

Password Exposure

- Password provided to compromised web server



www.myhobby.com is compromised!

Password Verification

- What if…
  - A compromised OS cannot learn the password
  - Only essential code can access password
    - Decrypt SSL traffic
    - Salt and hash password
    - Compare with stored hash
    - Output MATCH or FAILURE
  - Can remotely verify this is so
- Requires strong system security
- What about zero knowledge protocols?
  - A viable alternative for passwords
  - Our techniques are more general
    - Password verification is just an example

Outline

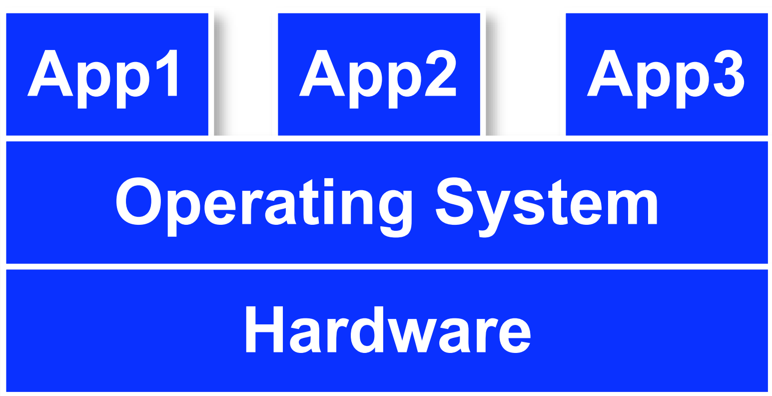
1. **Existing approaches to system security**
2. Remote attestation and verification
3. Static root of trust for measurement
4. Dynamic root of trust for measurement
5. Flicker: Minimal TCB Code Execution

- Optional
  - Example: IBM Integrity Measurement Arch.
  - Specifics of AMD SVM / Intel TXT

Some Current Approaches

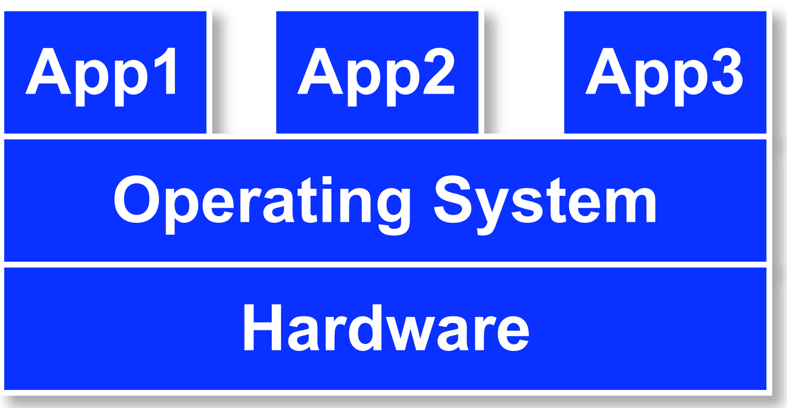
- Program code in ROM
- Secure boot
- Virtual-machine-based isolation

- Evaluation metric: size of Trusted Computing Base (TCB)



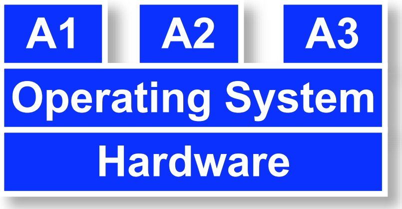
Security Properties to Consider

- How can we trust operations that our devices perform?
- How can we trust App1?
- What if App2 has a security vulnerability?
- What if Operating System has a security vulnerability?



Program Code in ROM

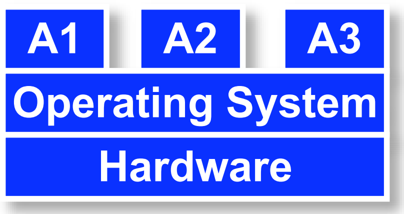
- Advantages
  - Simplicity
  - Adversary cannot inject any additional software
- Disadvantages
  - Cannot update software (without exchanging ROM)
  - Adversary can still use control-flow attack
  - Entire system is in TCB, no isolation



- Verdict
  - Impractical for current systems
  - Code updates are critical
    - Bug fixes
    - New features

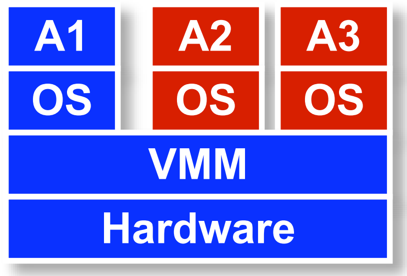
Secure Boot

- Boot process uses signature chain
  - BIOS verifies signature on boot loader
  - Boot loader verifies signature on OS, ...
- Advantages
  - Only approved software can be loaded
    - Assuming no vulnerabilities
- Disadvantages
  - Adversary only needs to compromise singe component
  - Entire system is in TCB, no isolation
  - Not all software is commercial
- Verdict
  - Entire system is still part of TCB
  - Relatively weak security guarantee



Virtual-machine-based Isolation

- Approach: Isolate applications by executing them inside different Virtual Machines
- Advantages
  - Smaller TCB
  - Isolation between applications
- Disadvantages
  - VMM is still large and part of TCB
  - Relatively complex, not suitable for average user
- Verdict: Smaller TCB, step in right direction



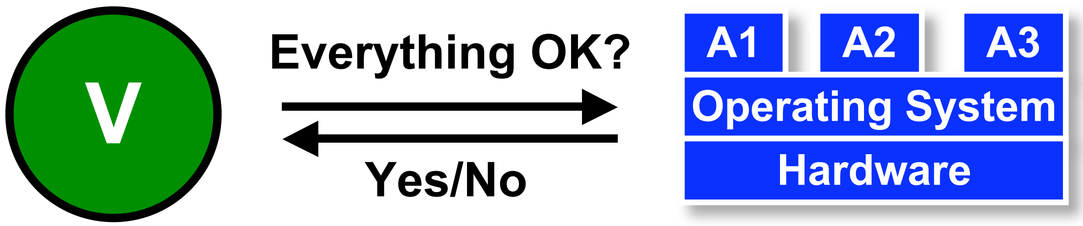
Outline

1. Existing approaches to system security
2. **Remote attestation and verification**
3. Static root of trust for measurement
4. Dynamic root of trust for measurement
5. Minimal TCB Code Execution

- Optional
  - Example: IBM Integrity Measurement Arch.
  - Specifics of AMD SVM / Intel TXT

Remote Verification?

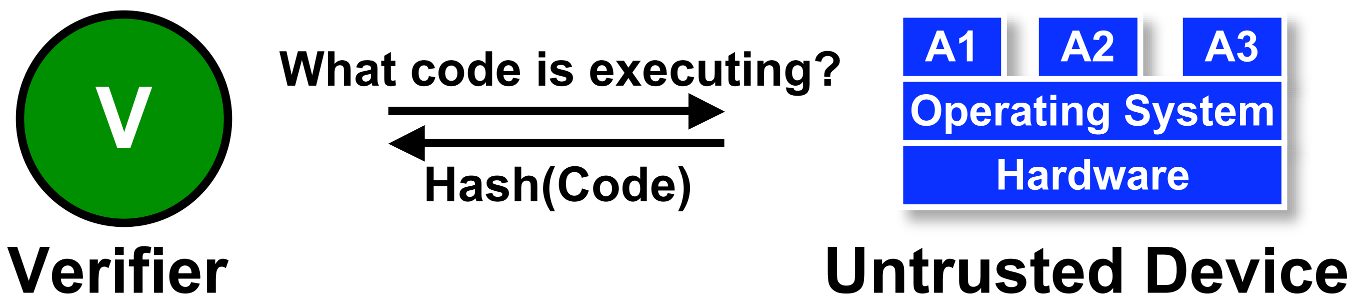
- Desirable property: Remotely verify trustworthy device operation



- Presented approaches not verifiable
  - Higher resilience to attacks
  - Remote verifier obtains no additional assurance

Remote Attestation

- Attestation enables verifier to establish trust in untrusted device
  - Attestation tells verifier what code is executing on device
  - If intended code is executing on untrusted device, verifier can trust its operation



Outline

1. Existing approaches to system security
2. Remote attestation and verification
3. **Static root of trust for measurement**
4. Dynamic root of trust for measurement
5. Flicker: Minimal TCB Code Execution

- Optional
  - Example: IBM Integrity Measurement Arch.
  - Specifics of AMD SVM / Intel TXT

Hardware-based Attestation

- Leverages hardware support for attestation
- Trusted Platform Module (TPM) chip
  - Already included in many platforms
  - Cost per chip less than $10

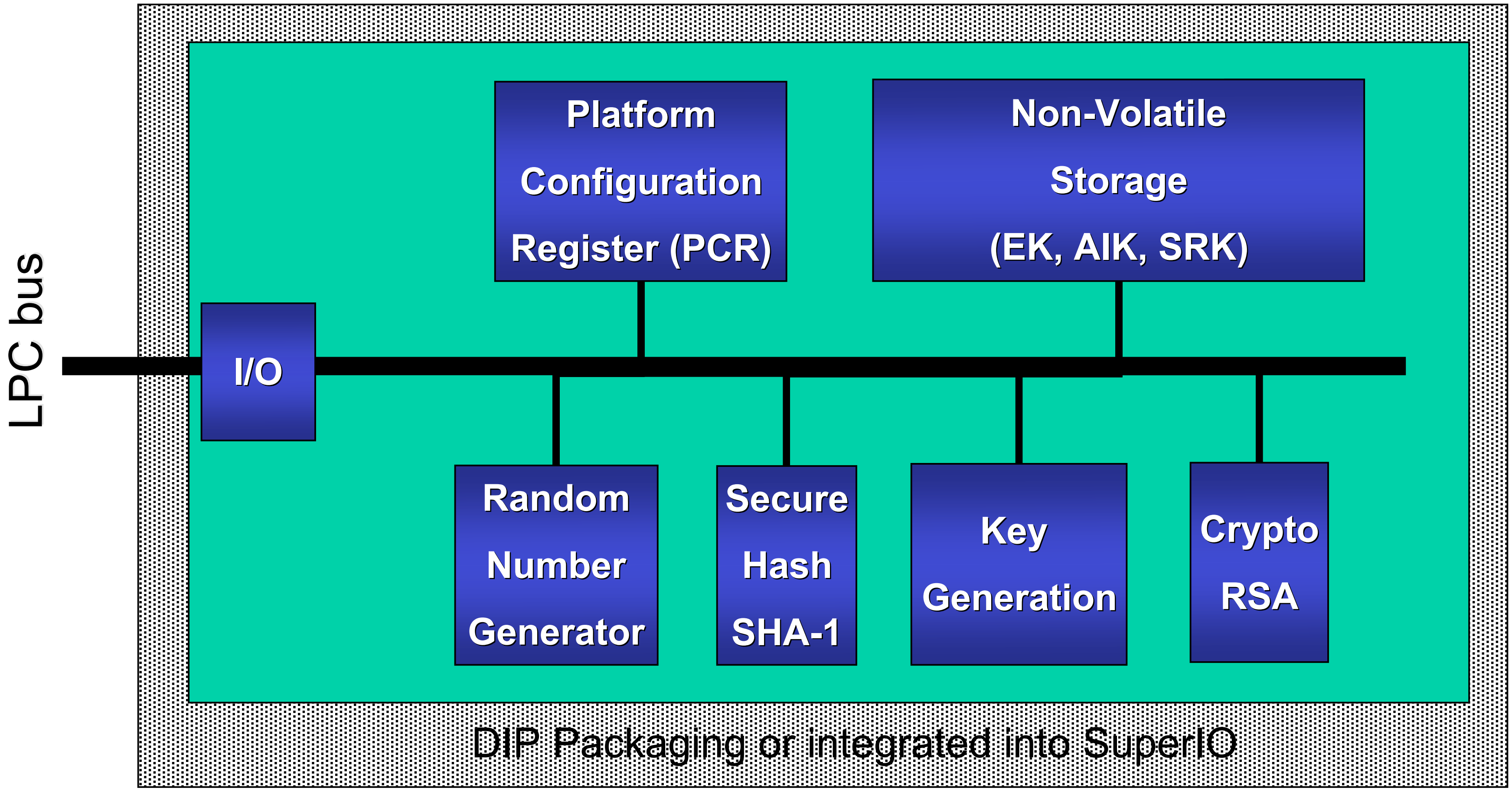
- Modern microprocessors provide special instructions that interact with TPM chip
  - AMD SVM: SKINIT instruction
  - Intel TXT/LT: GETSEC[SENTER] instruction

Trusted Computing Group (TCG)

- Open organization to “develop, define, and promote open standards for hardware-enabled trusted computing and security technologies.”

- These secure platform primitives include
  - Platform integrity measurements
  - Measurement attestation
  - Sealed storage
- Can enable
  - Trusted boot (not secure boot)
  - Attestation
- Goals:
  - Ensure absence of malware
  - Detect spyware, viruses, worms, …

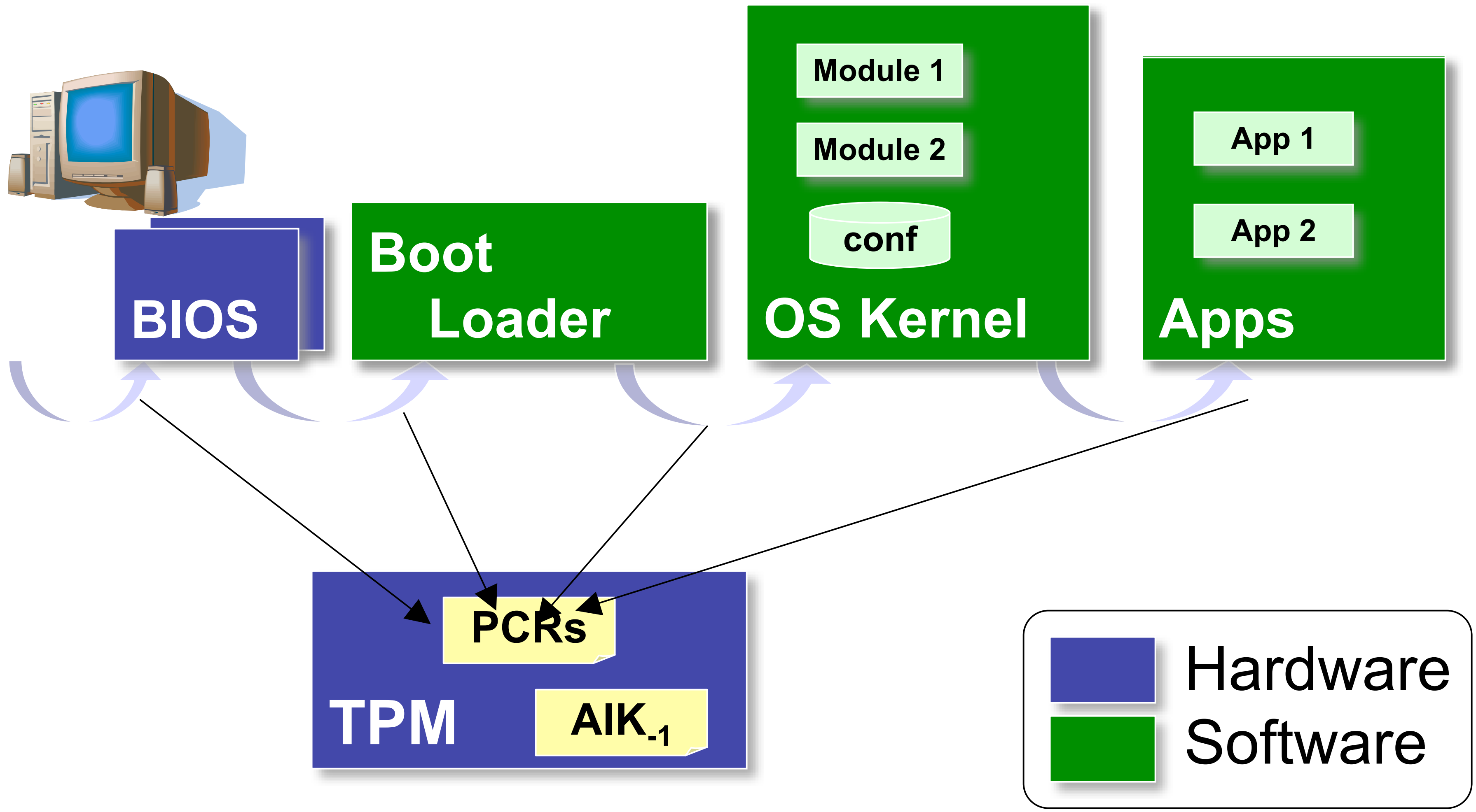
TCG Trusted Platform Module (TPM)



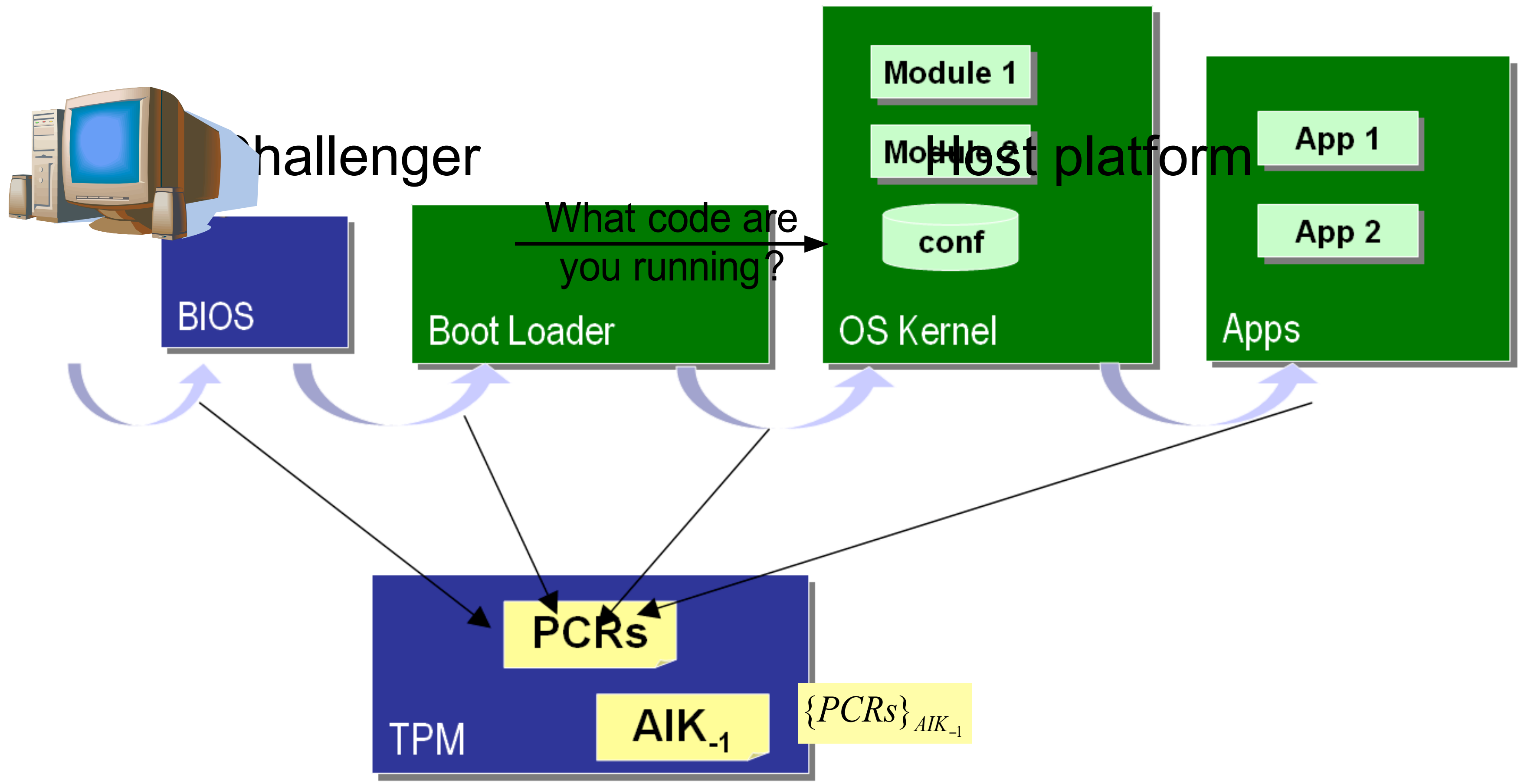
Basic TPM Functions

- PCRs store integrity measurement chain
  - PCR_{new} = SHA-1(PCR_{old}||measurement)
- Remote attestation (PCRs + AIK)
  - Attestation Identity Keys (AIKs) for signing PCRs
  - Attest to value of integrity measurements to remote party
- Sealed storage (PCRs + SRK)
  - Protected storage + unlock state under a particular integrity measurement (data portability concern)

TCG-Style Attestation



TCG-Style Attestation

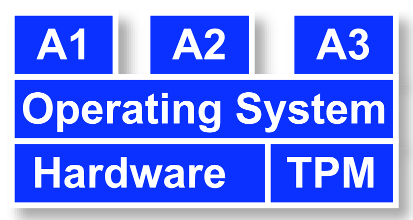


Optional

- IBM's Integrity Measurement Architecture
- Works for Linux

Shortcomings of TCG-style Attestation

- **Static** root of trust for measurement (**reboot**)
- Coarse-grained, measures entire system
  - Requires hundreds of integrity measurements just to boot
  - Every host is different
    - firmware versions, drivers, patches, apps, spyware, …
  - What does a PCR mean in this context?
  - TCB includes entire system!
- Integrity measurements are done at load-time not at run-time
  - Time-of-check-time-of-use (TOCTOU) problem
  - Cannot detect any dynamic attacks!
  - No guarantee of execution



Outline

1. Existing approaches to system security
2. Remote attestation and verification
3. Static root of trust for measurement
4. **Dynamic root of trust for measurement**
5. Flicker: Minimal TCB Code Execution

- Optional
  - Example: IBM Integrity Measurement Arch.
  - Specifics of AMD SVM / Intel TXT

Dynamic Root of Trust for Measurement aka: Late Launch

- Involves both CPU and TPM v1.2
- Security properties similar to reboot
  - Without a reboot!
  - Removes many things from TCB
    - BIOS, boot loader, DMA-enabled devices, …
    - Long-running OS and Apps if done right
- When combined with virtualization
  - VMM can be measured (MVMM)
    - Uptimes measured in *years*
  - Integrity of loaded code can be attested
  - Untrusted legacy OS can coexist with trusted software
- Allows introduction of new, higher-assurance software without breaking existing systems

AMD/Intel Late Launch Extensions

- AMD: Secure Virtual Machine (SVM)
- Intel: Trusted eXecution Technology (TXT)
  - Formerly LaGrande Technology (LT)
- Similarities:
  - Late launch of a measured block of code
  - Hardware support for virtualization
- Differences:
  - AMD provides measured environment only
  - Intel adds authenticated code capabilities
    - The system's chipset contains a public key to verify signed code

AMD Secure Virtual Machine

- Virtualization support
  - DMA protection for memory
  - Intercept selected guest instructions / events
  - Much more…

- Late launch with support for attestation
  - New instruction: SKINIT (Secure Kernel Init)
  - Requires appropriate platform support (e.g., TPM 1.2)
  - Allows verifiable startup of trusted software
    - Such as a VMM
    - Based on hash comparison

SKINIT (Secure Kernel Init)

- Accepts address of Secure Loader Block (SLB)
  - Memory region up to 64 KB

- SKINIT executes atomically
  - Sets CPU state similar to INIT (soft reset)
  - Disables interrupts
  - Enables DMA protection for entire 64 KB SLB
  - Causes TPM to *reset dynamic PCRs* to 0
  - Sends SLB contents to TPM
  - TPM hashes SLB contents and extends PCR 17
  - Begins executing SLB

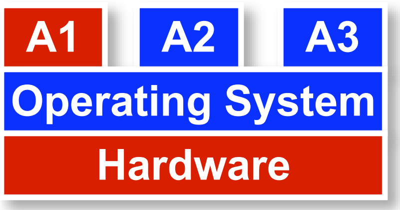
SKINIT Security Properties

- Verifier receives attestation after SKINIT
  - Knows SKINIT was used
  - Knows software TCB includes only the SLB
  - Knows exactly what SLB was executed

- SLB can be written to provide add’l props.
  - Knows any inputs to SLB
  - Knows any outputs from SLB
  - Knows exactly when SLB finished executing

AMD SVM Security Discussion

- Property: Verifiable untampered code execution
- SKINIT + TCG 1.2 provide very strong security properties
- Minimal TCB: Only hardware and application need to be trusted



Optional

- Detail on specific AMD/Intel Extensions
  - AMD Secure Virtual Machine (SVM)
  - Intel Trusted eXecution Technology (TXT)

Outline

1. Existing approaches to system security
2. Remote attestation and verification
3. Static root of trust for measurement
4. Dynamic root of trust for measurement
5. **Flicker: Minimal TCB Code Execution**

- Optional
  - Example: IBM Integrity Measurement Arch.
  - Specifics of AMD SVM / Intel TXT

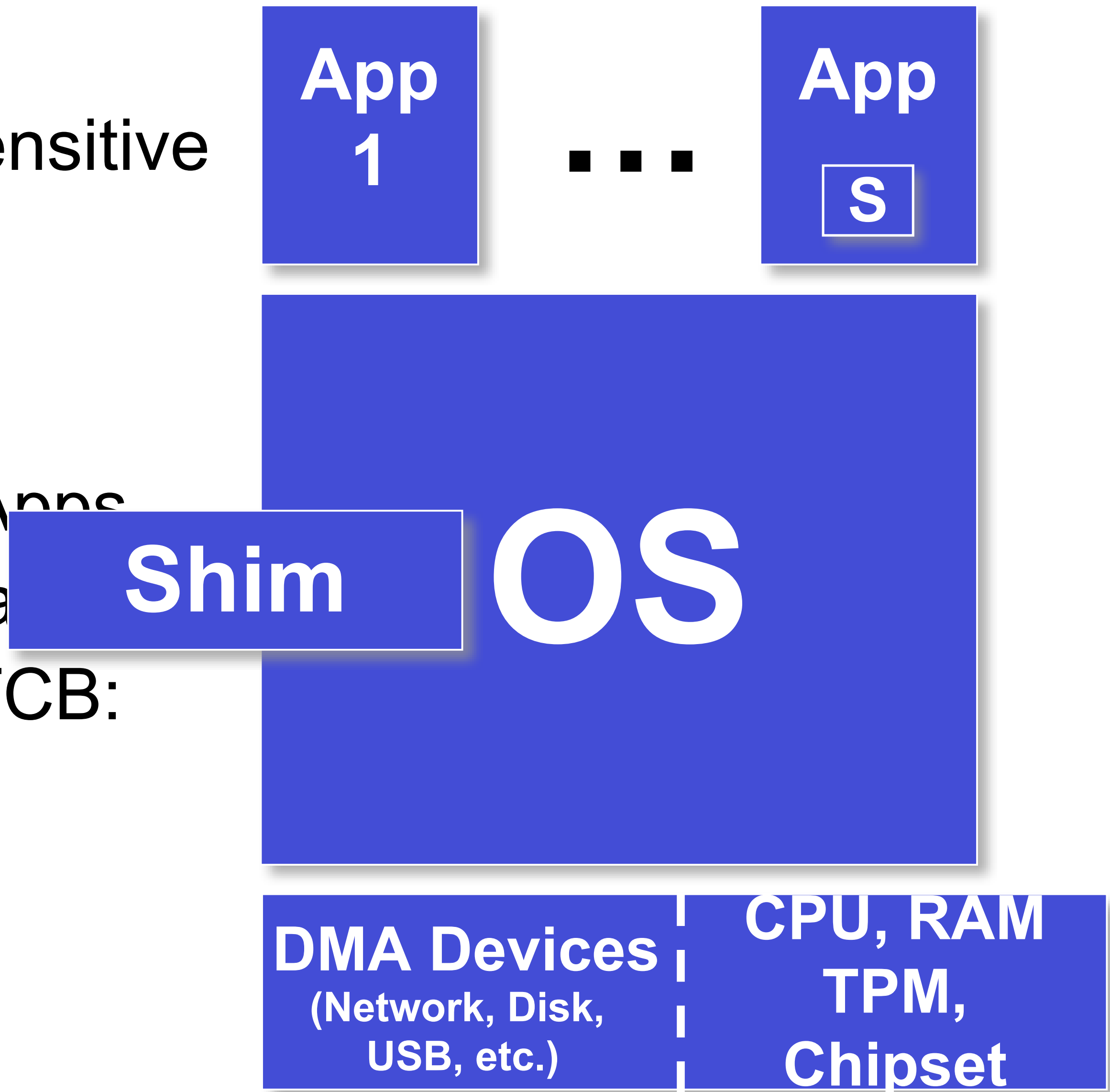
TCB Reduction with Flicker

Today, TCB for sensitive code S:

- Includes App
- Includes OS
- Includes other Apps
- Includes hardwa

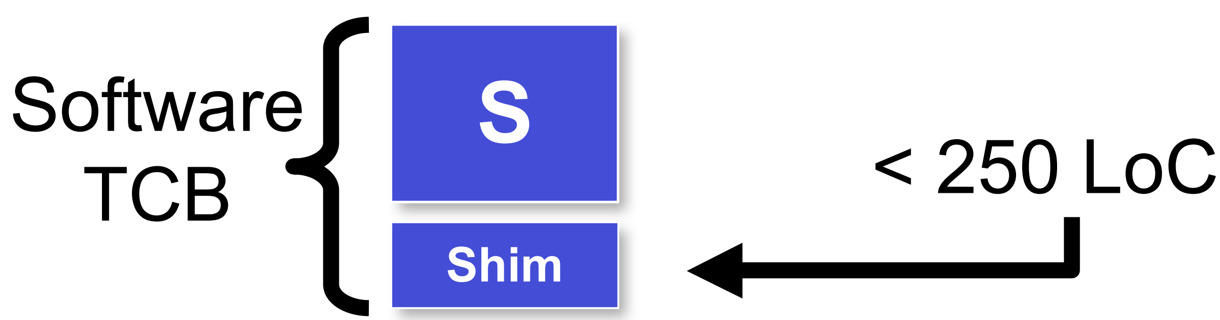
With Flicker, S's TCB:

- Includes Shim
- Includes some hardware

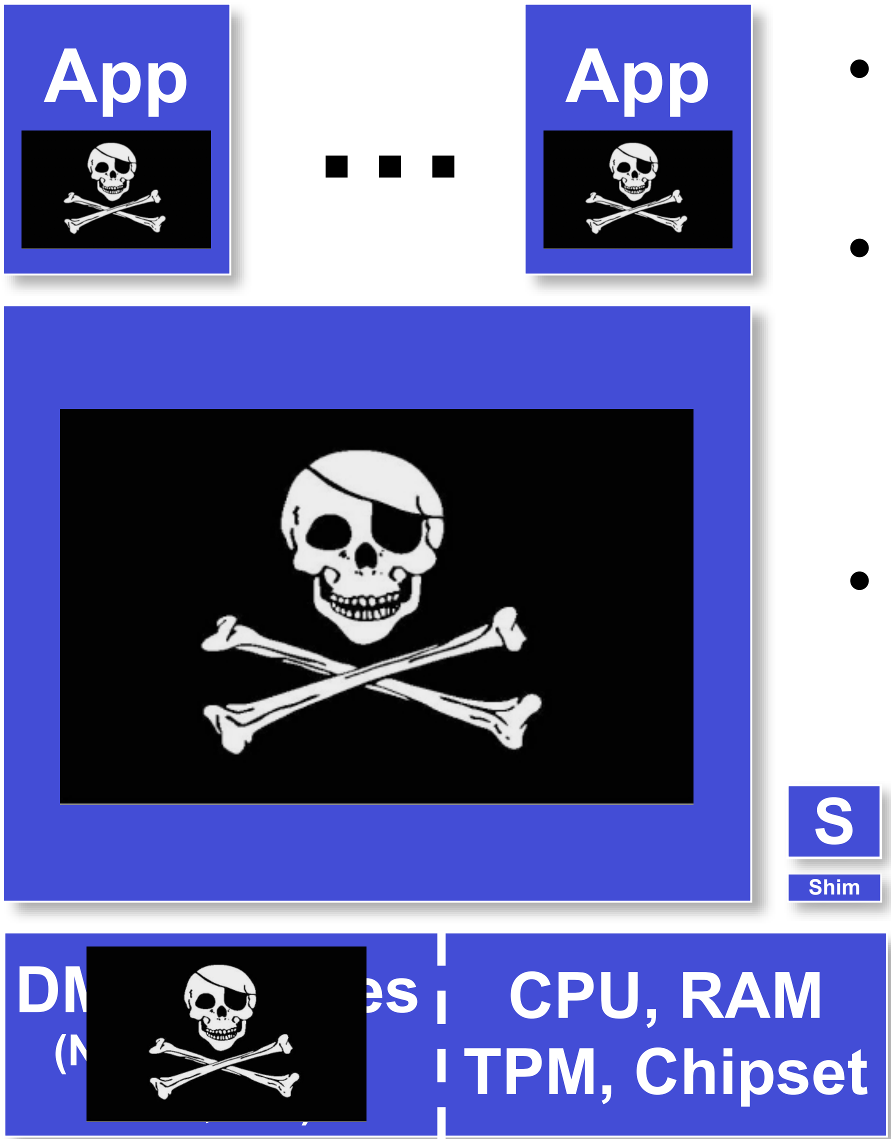


Contributions

- Isolate security-sensitive code execution from all other code and devices
- Attest to security-sensitive code and its arguments and nothing else
- Convince a remote party that security-sensitive code was protected
- Add < 250 LoC to the software TCB



Adversary Capabilities

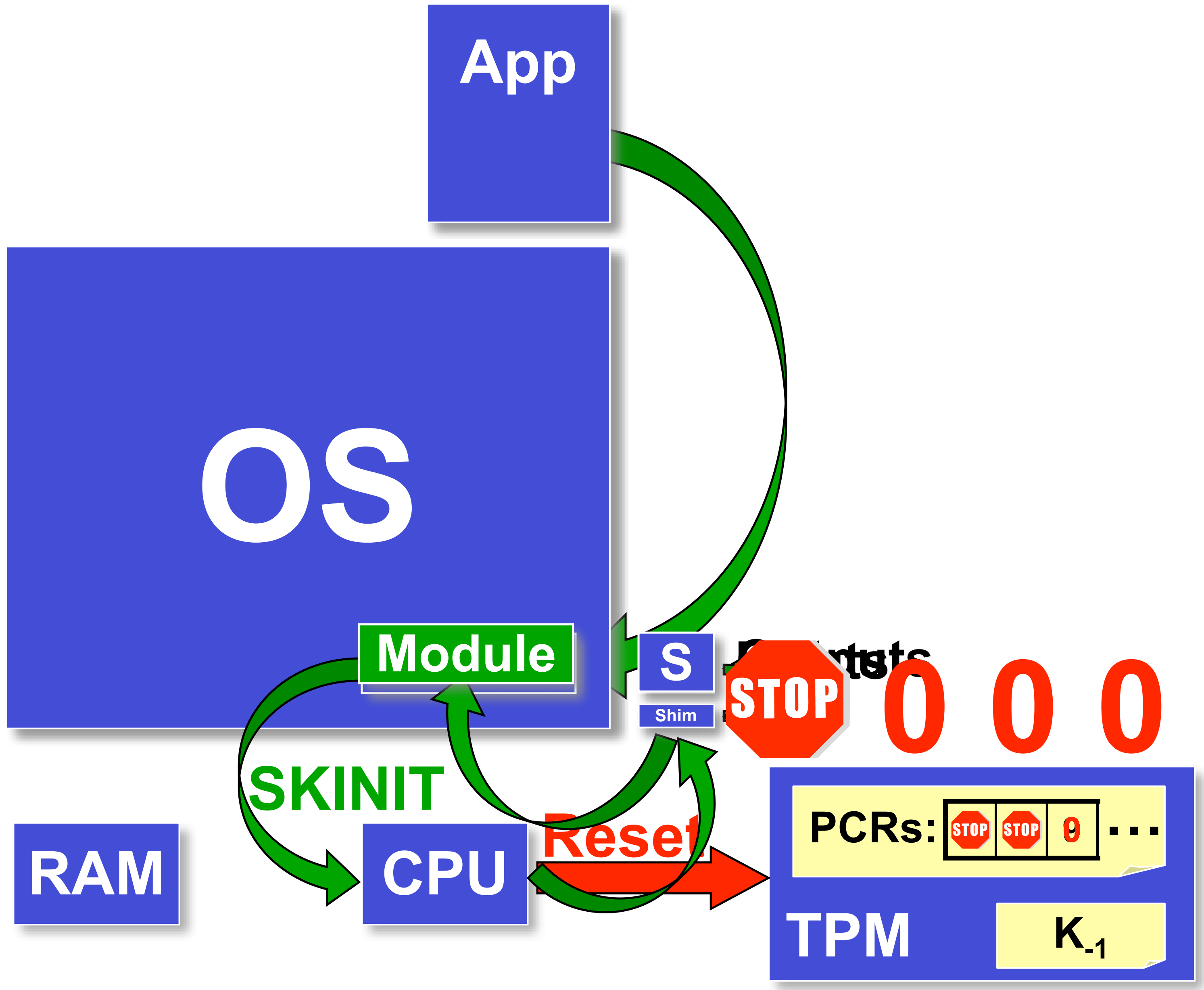


- Run arbitrary code with maximum privileges
- Subvert any DMA-enabled device
  - E.g., network cards, USB devices, hard drives
- Perform limited hardware attacks
  - E.g., power cycle the machine
  - Excludes physically monitoring/modifying CPU-to-RAM communication

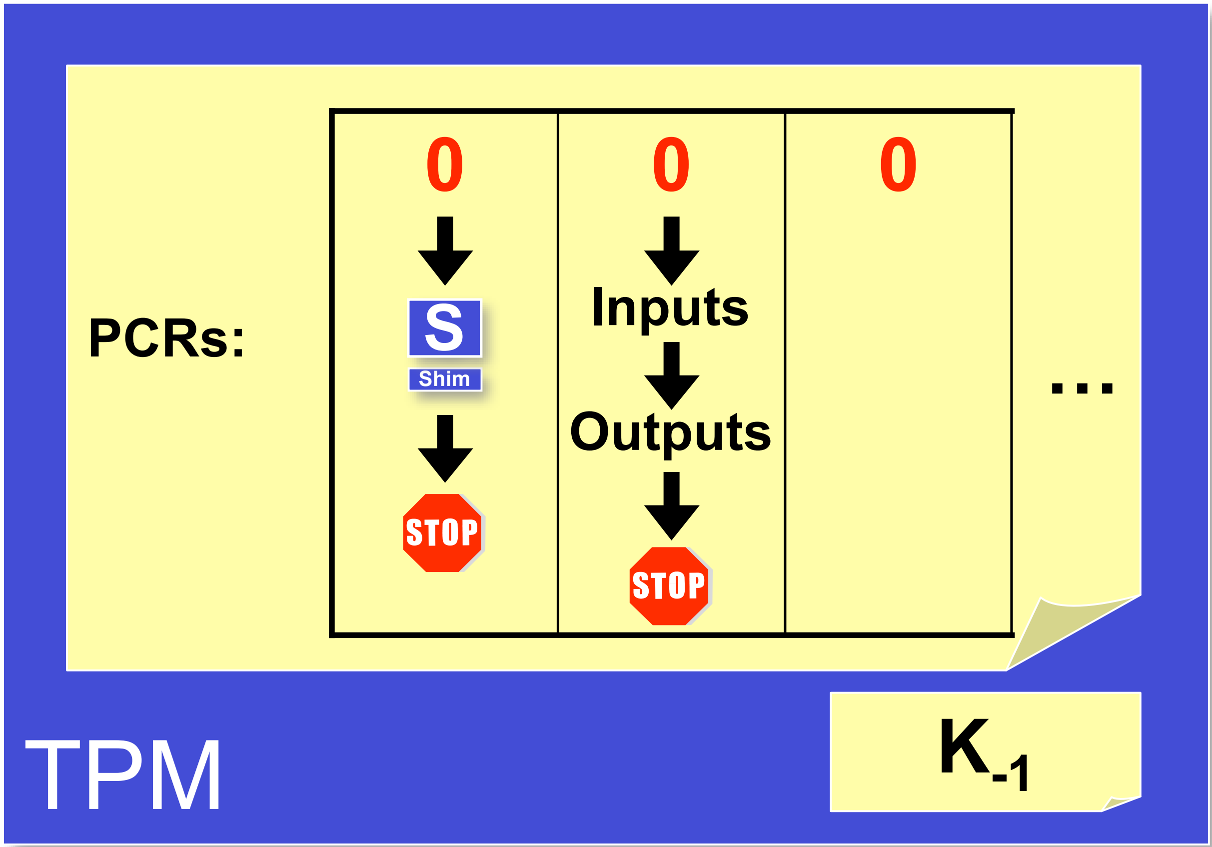
Architecture Overview

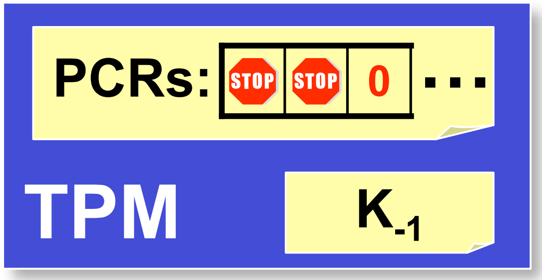
- Core technique
  - Pause current execution environment
  - Execute security-sensitive code with hardware-enforced isolation
  - Resume previous execution
- Extensions
  - Preserve state securely across invocations
  - Attest **only** to code execution and protection
  - Establish secure communication with remote parties

Execution Flow

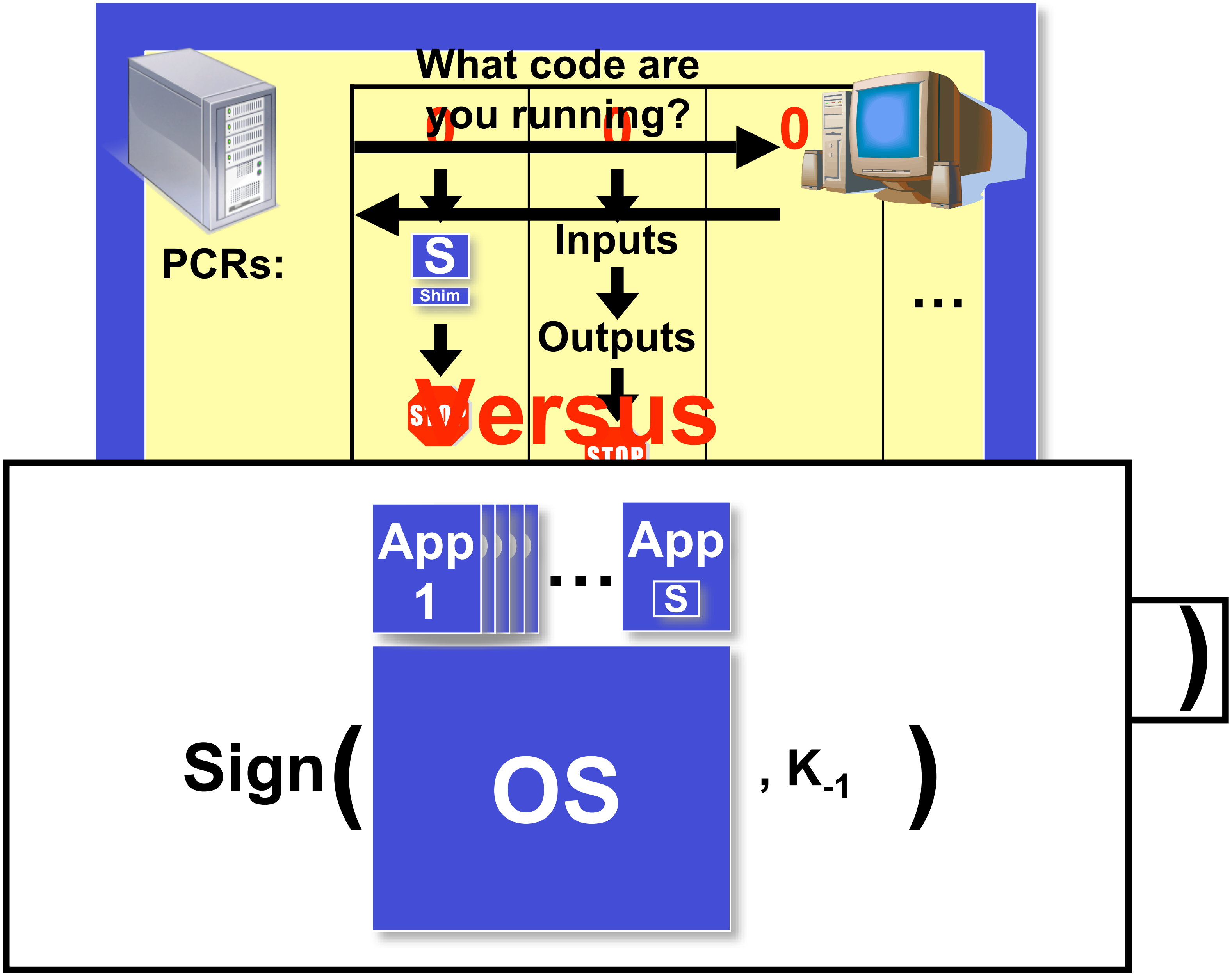


Attestation



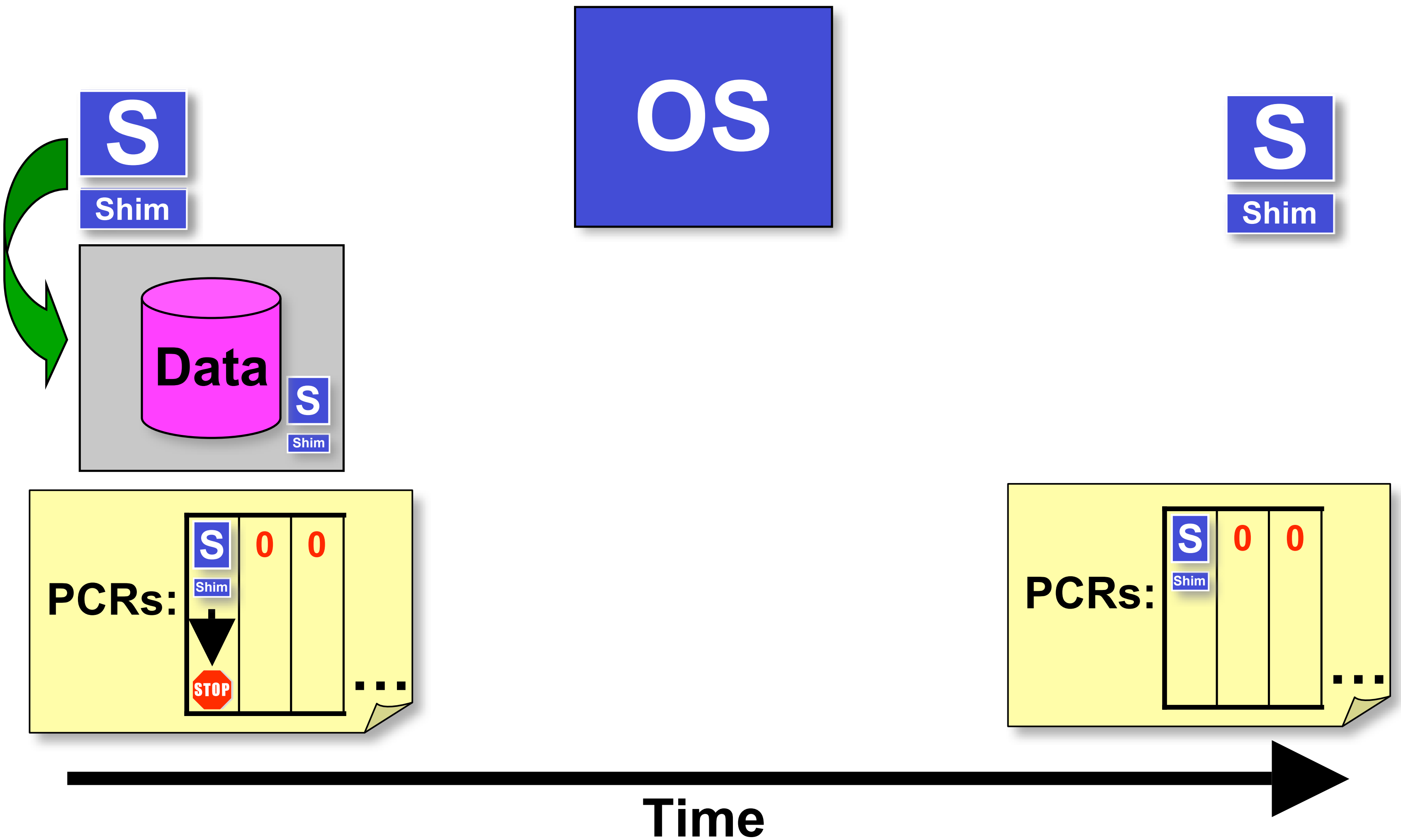


Attestation



Context Switch with Sealed Storage

- **Seal** data under combination of code, inputs, outputs
- Data unavailable to other code

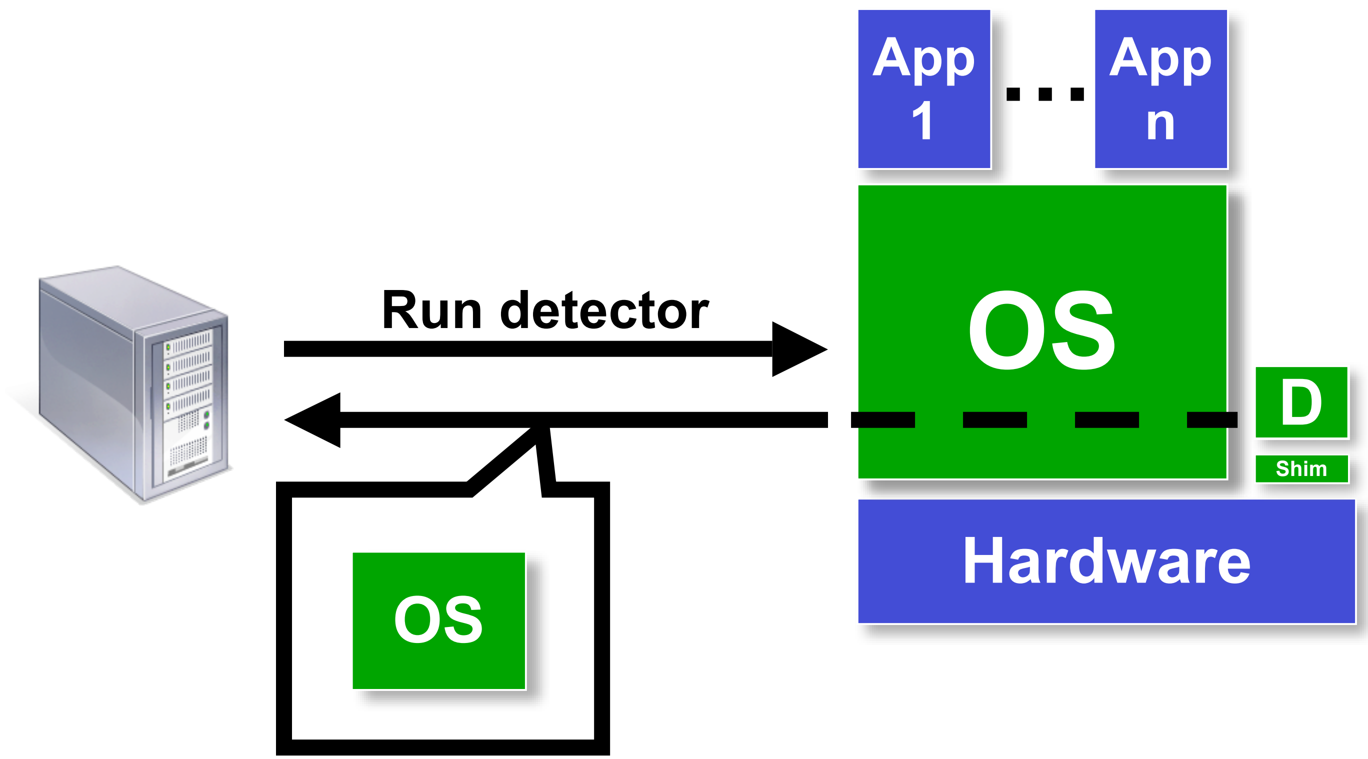


Functionality

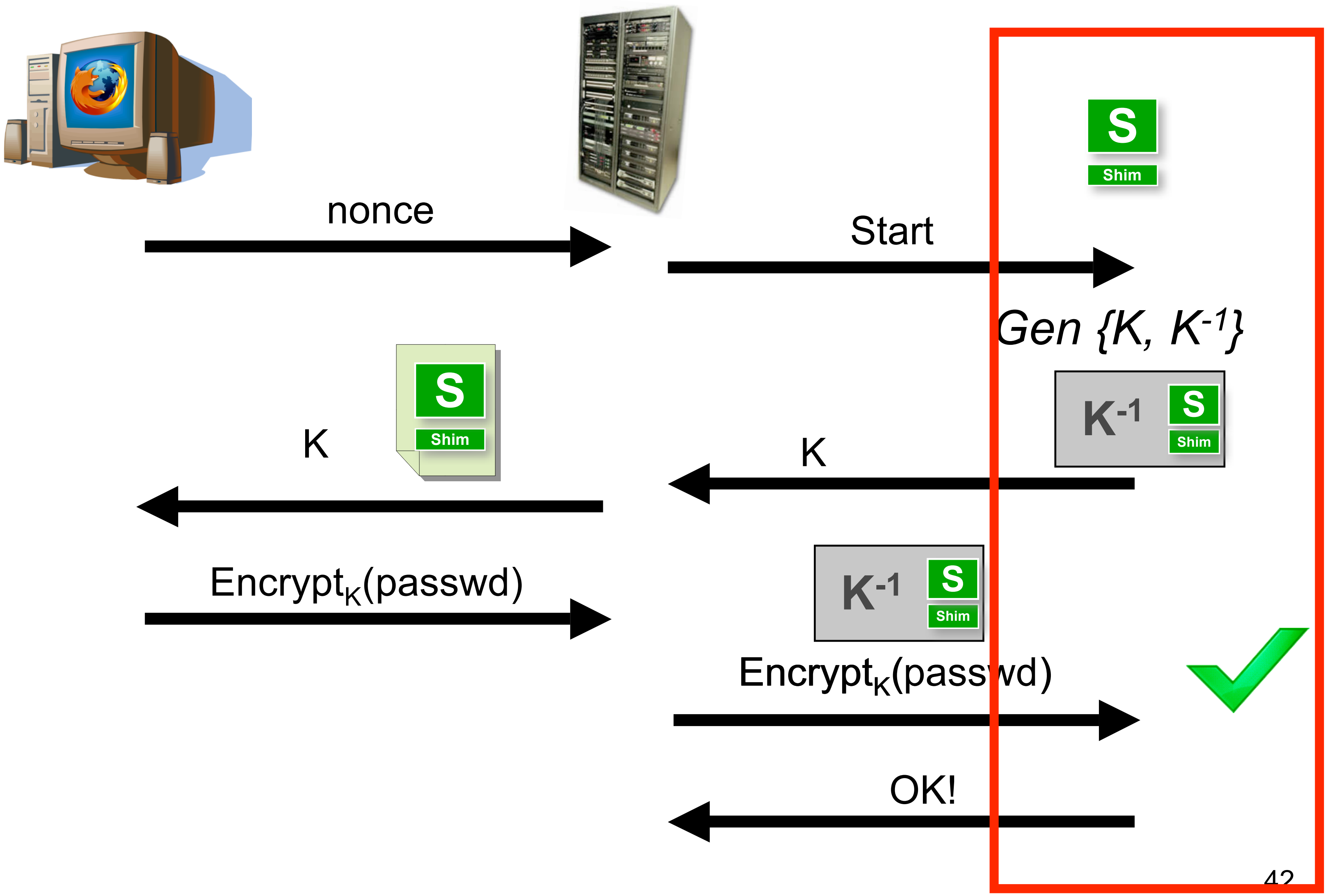
- Shim can execute arbitrary x86 code but provides very limited functionality
- Fortunately, many security-sensitive functions do not require much
  - E.g., key generation, encryption/decryption, FFT
- Functionality can be added to support a particular security-sensitive operation
- We have partially automated the extraction of support code for security-sensitive code

Application: Rootkit Detector

- Administrator can check the integrity of remote hosts
  - E.g., only allow uncompromised laptops to connect to the corporate VPN



Application: SSH Passwords

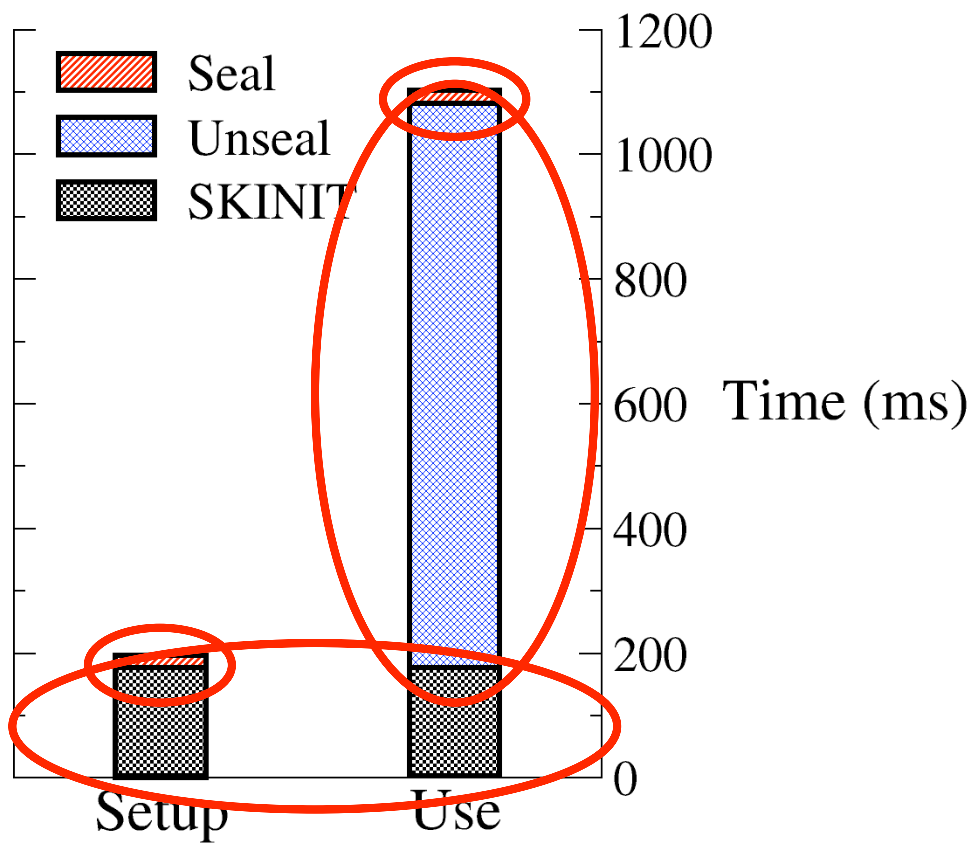


Other Applications Implemented

- Enhanced Certificate Authority (CA)
  - Private signing key isolated from entire system
- Verifiable distributed computing
  - Verifiably perform a computational task on a remote computer
  - Ex: SETI@Home, Folding@Home, distcc

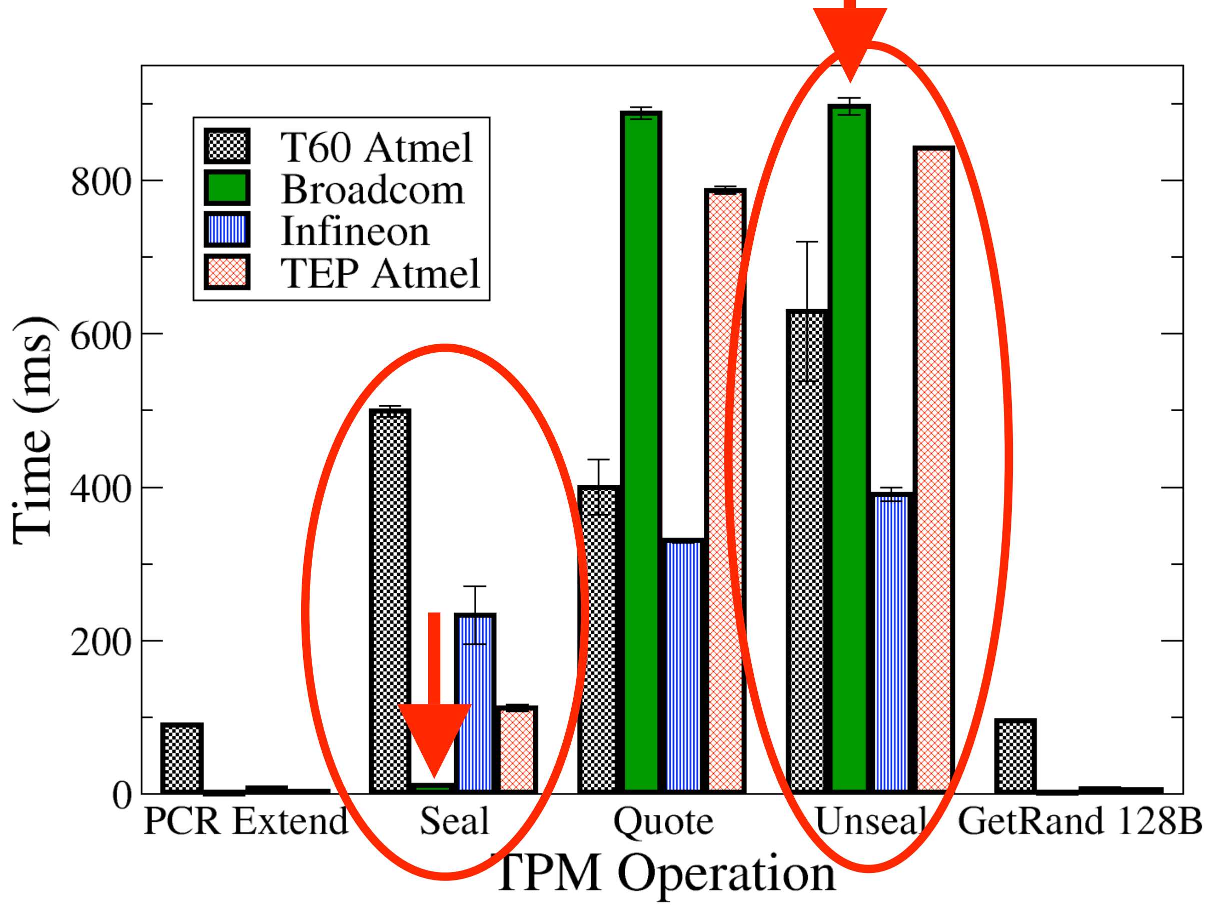
TPM-related Performance

- During every Flicker context switch
  - Application state protection while OS runs



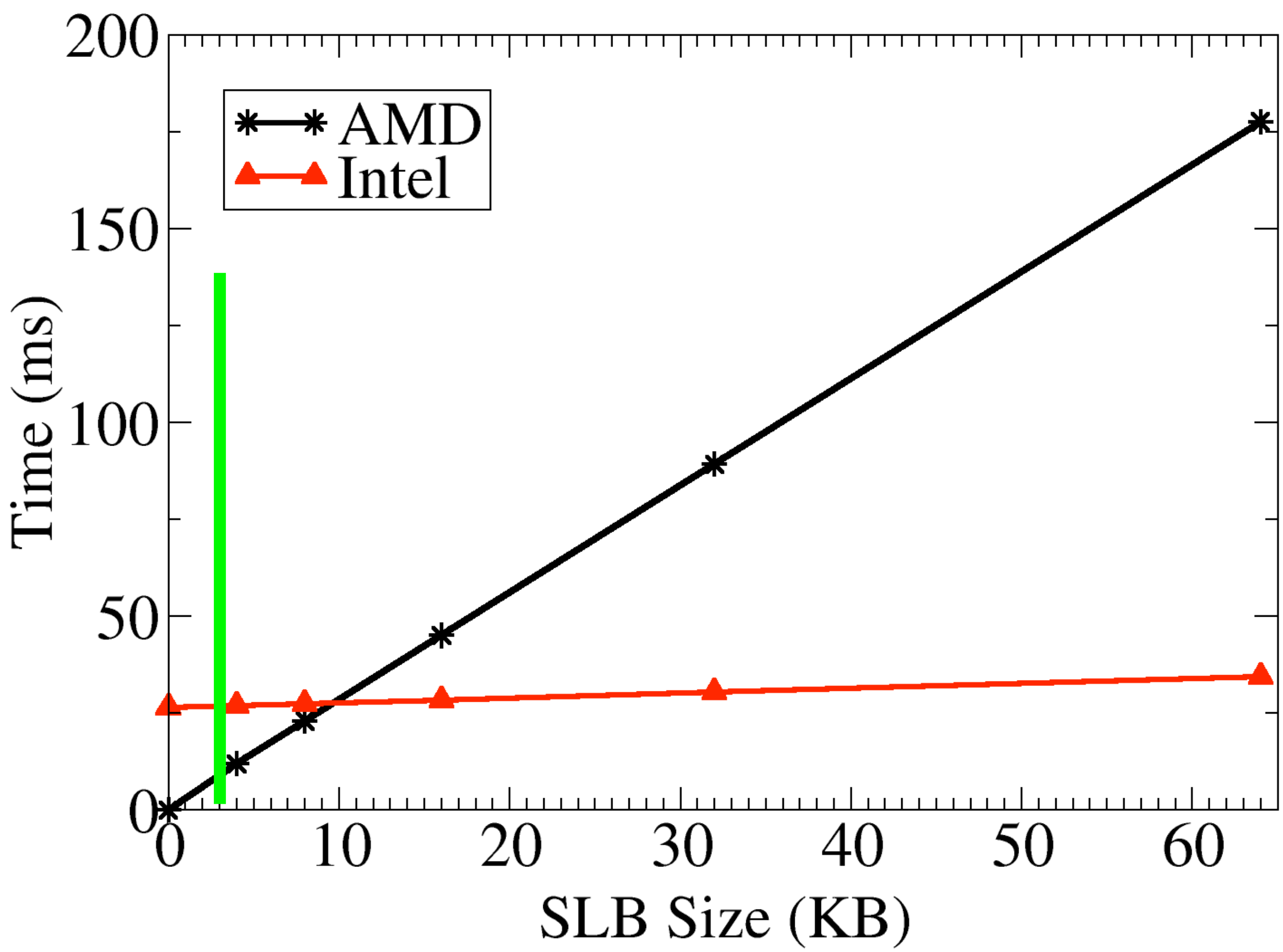
TPM Microbenchmarks

- Significant variation by TPM model



Breakdown of Late Launch Overhead

- After ~4KB, code can measure itself



Ongoing Work

- Containing malicious or malfunctioning security-sensitive code
- Creating a trusted path to the user
- Porting implementation to Intel
- Improving automatic privilege separation

Conclusions

- Explore how far an application's TCB can be minimized
- Isolate security-sensitive code execution
- Provide fine-grained attestations
- Allow application writers to focus on the security of their own code

Thank you!

jonmccune@cmu.edu