

18732: Secure Software Systems

Trusted Computing II

Anupam Datta

CMU
Fall 2010

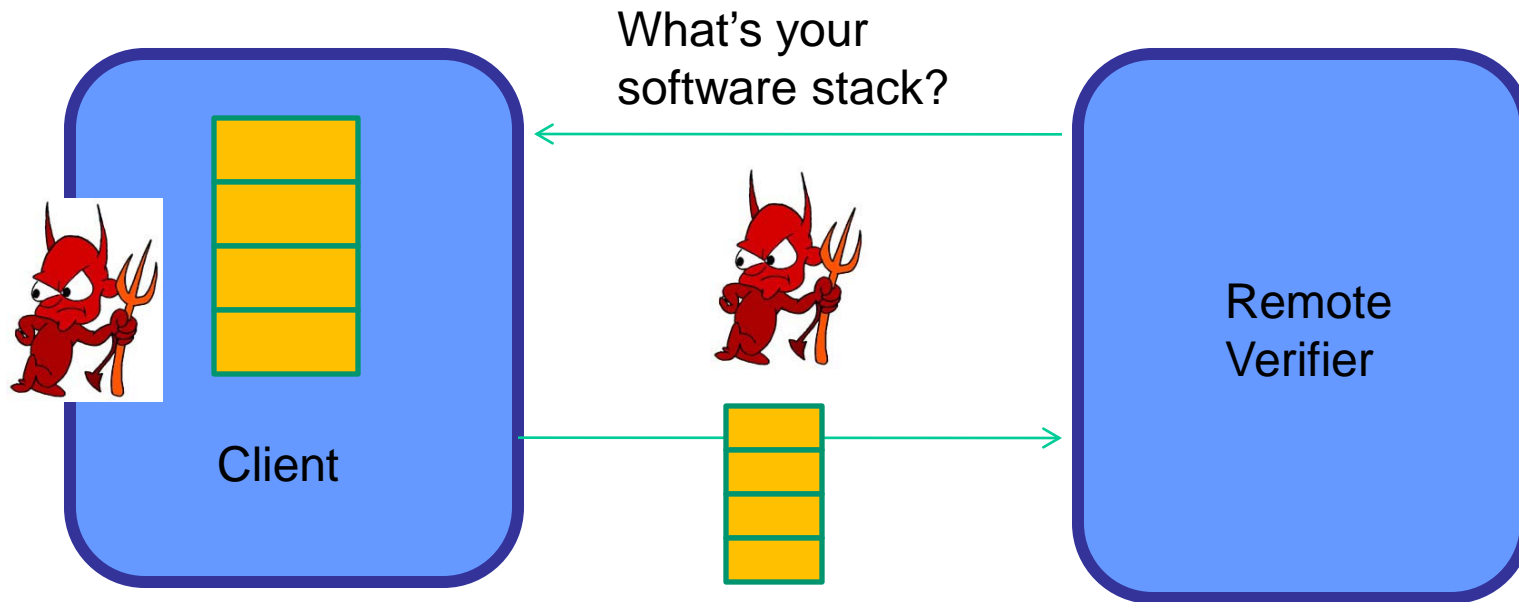
Plan for today

- Recap Static Root of Trust Measurement
 - Mechanism, adversary model, security property
 - Limitations
- Dynamic Root of Trust Measurement
 - Late Launch
 - Flicker
- Sealed Storage

Thinking about System Security

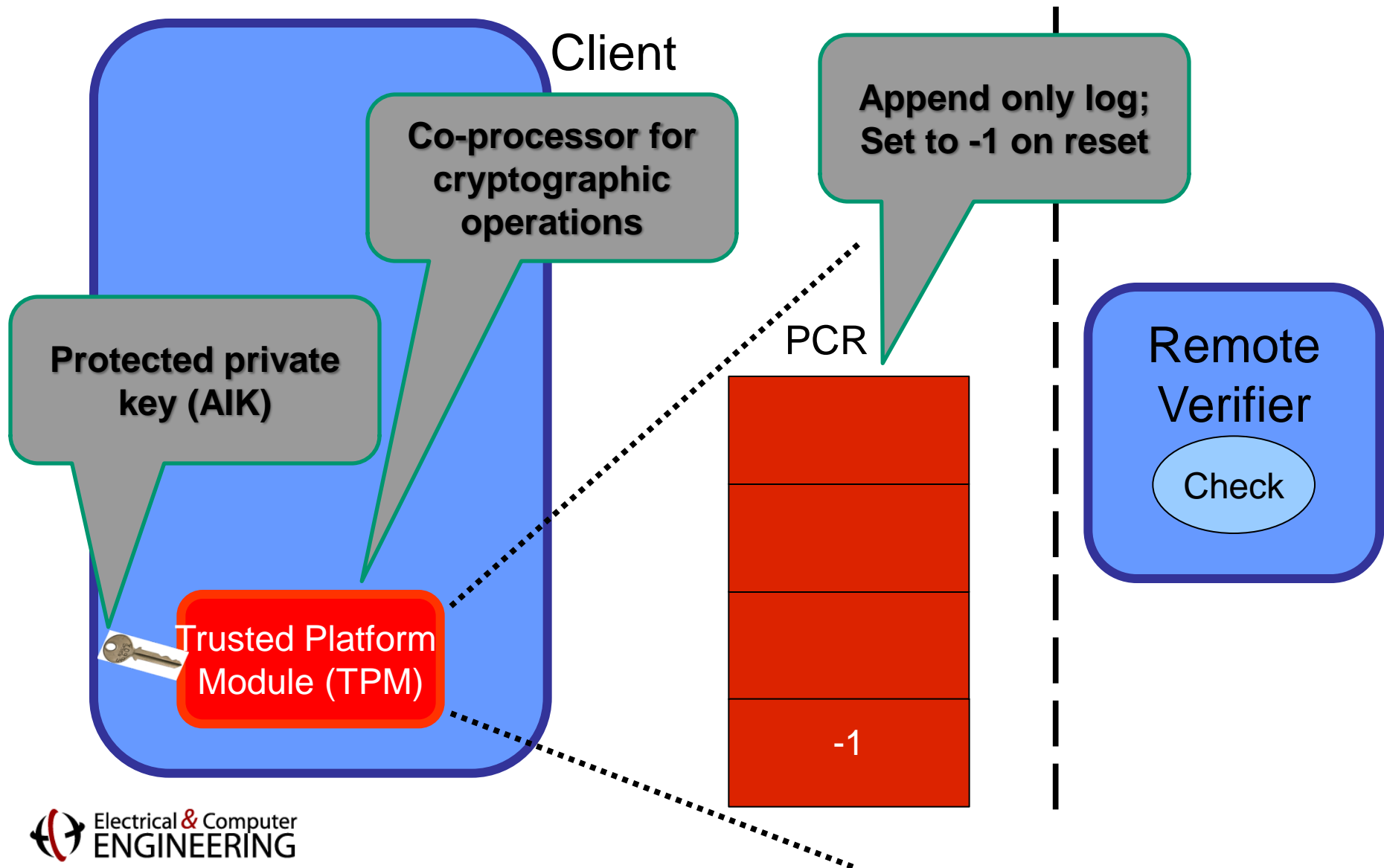
1. What is the mechanism?
2. What are the adversary capabilities?
3. What is the security property?

Static Root of Trust Measurement (SRTM)

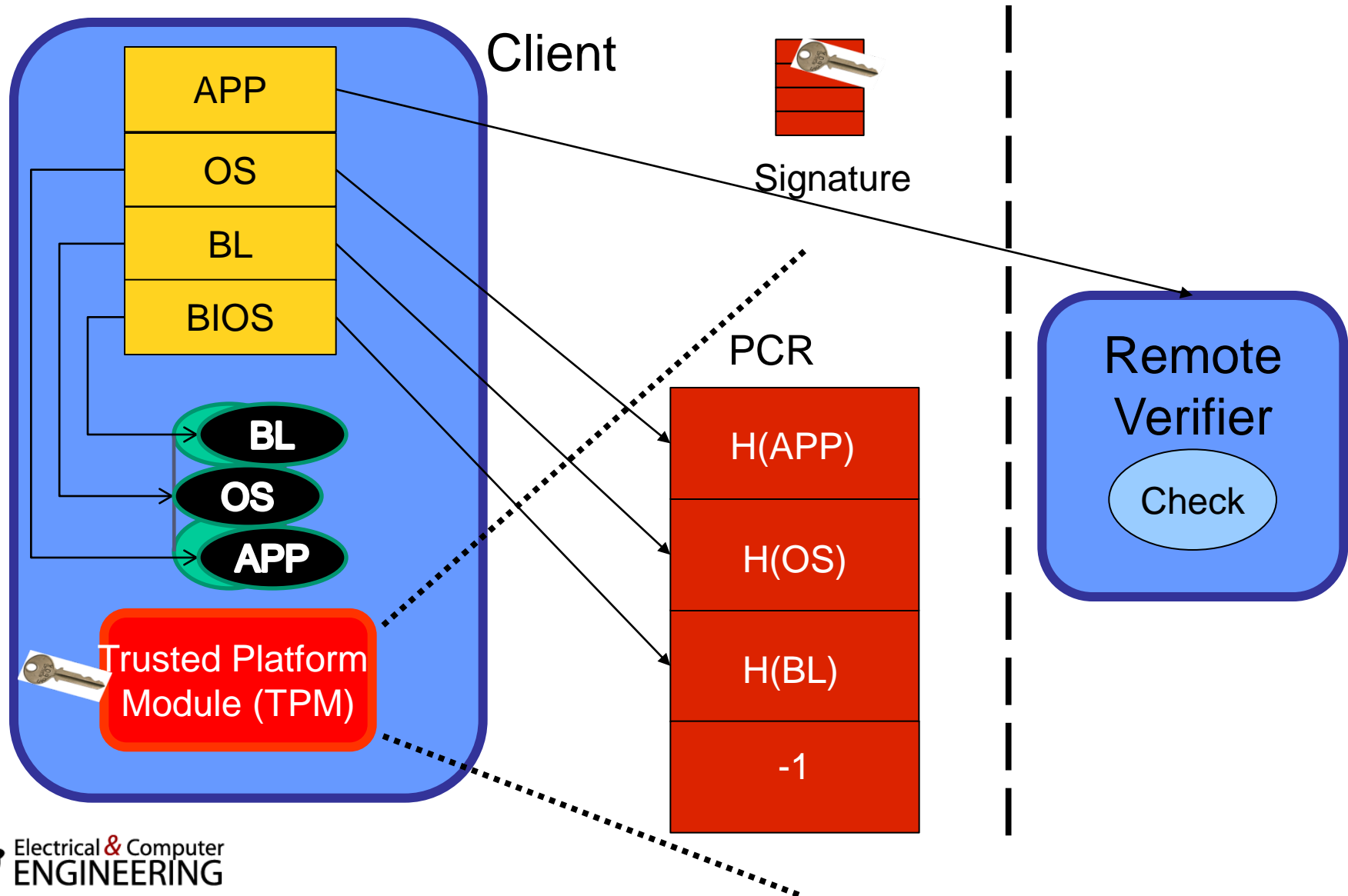


Why should the client's answer be trusted?

Static Root of Trust Measurement (SRTM)



Static Root of Trust Measurement (SRTM)

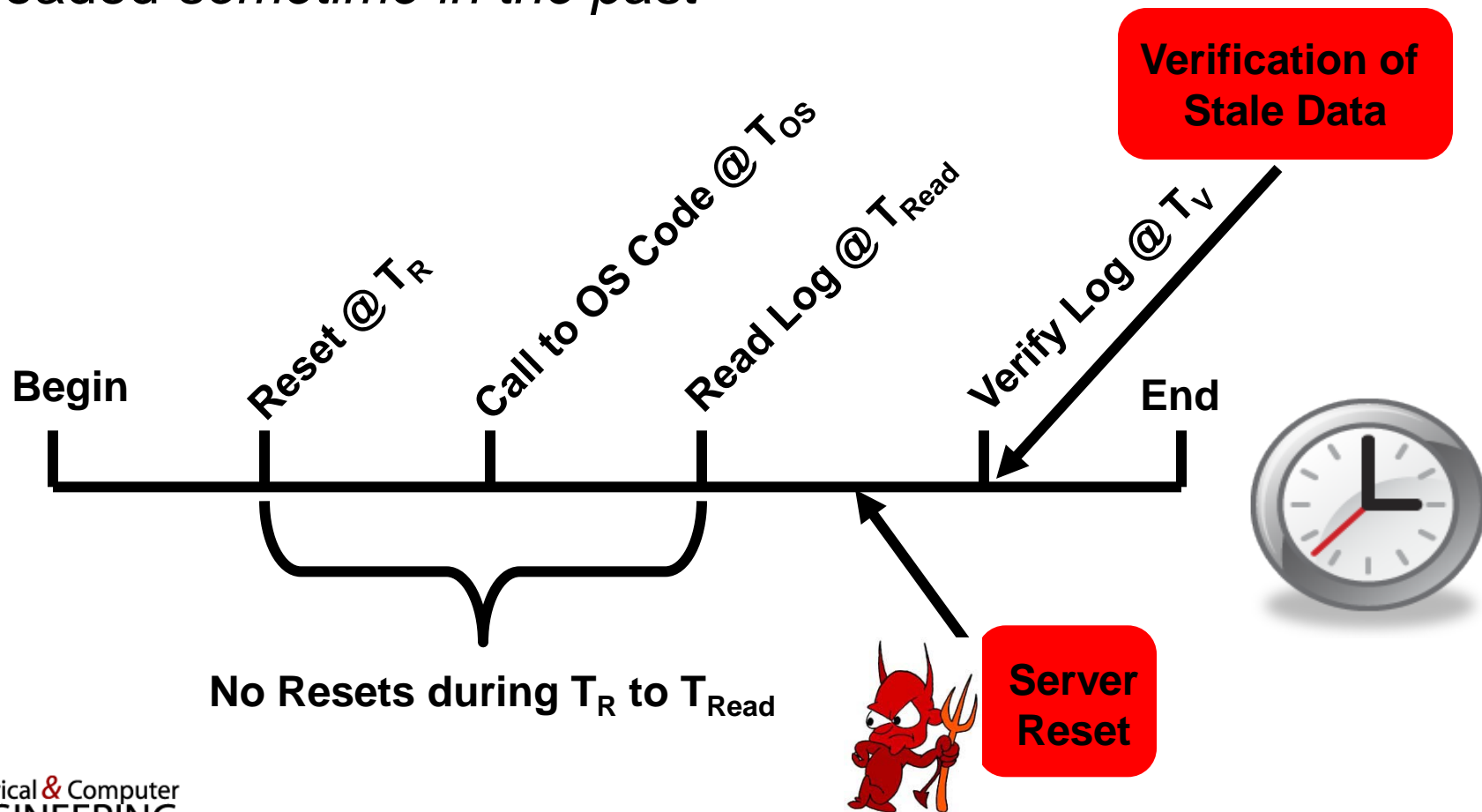


Adversary Model

- Network
 - Attacker has complete control over the network (read, intercept, inject messages)
- System
 - Attacker has complete control over the software stack (can modify, read), can reset machine
 - Attacker cannot break hardware protections
- Cryptography
 - Attacker cannot break cryptography (hash function is *collision resistant*, signatures are *unforgeable*)

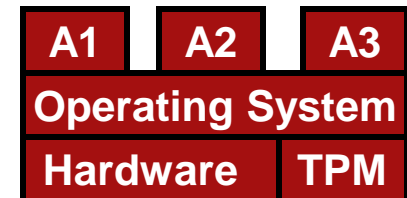
SRTM Property

Temporal property: software stack was loaded *sometime in the past*



Shortcomings of Static Root of Trust

- Integrity measurements are done at load-time not at run-time
 - Time-of-check-time-of-use (TOCTOU) problem
- Coarse-grained, measures entire system
 - Every host is different: different firmware versions, different drivers, different patches, different apps, different spyware
 - What does a PCR mean in this context?
 - TCB includes entire system!
- No guarantee of execution



Dynamic Root of Trust

Dynamic Root of Trust

- Security primitive provided by latest CPUs and TPMs (AMD SVM, Intel TXT)
- Late launch
 - Supports isolated code execution
 - Example: AMD's SKINIT instruction

SKINIT (Secure Kernel Init)

- Accepts address of Secure Loader Block (SLB)
 - Memory region up to 64 KB
- SKINIT executes atomically
 - Enables DMA protection for entire 64 KB SLB
 - Disables interrupts to prevent other code from gaining control
 - Debugging access is disabled (even for h/w debuggers)
 - Causes TPM to reset PCRs 17-23 and hash SLB contents and extend PCR 17
 - Begins executing at SLB's entry point

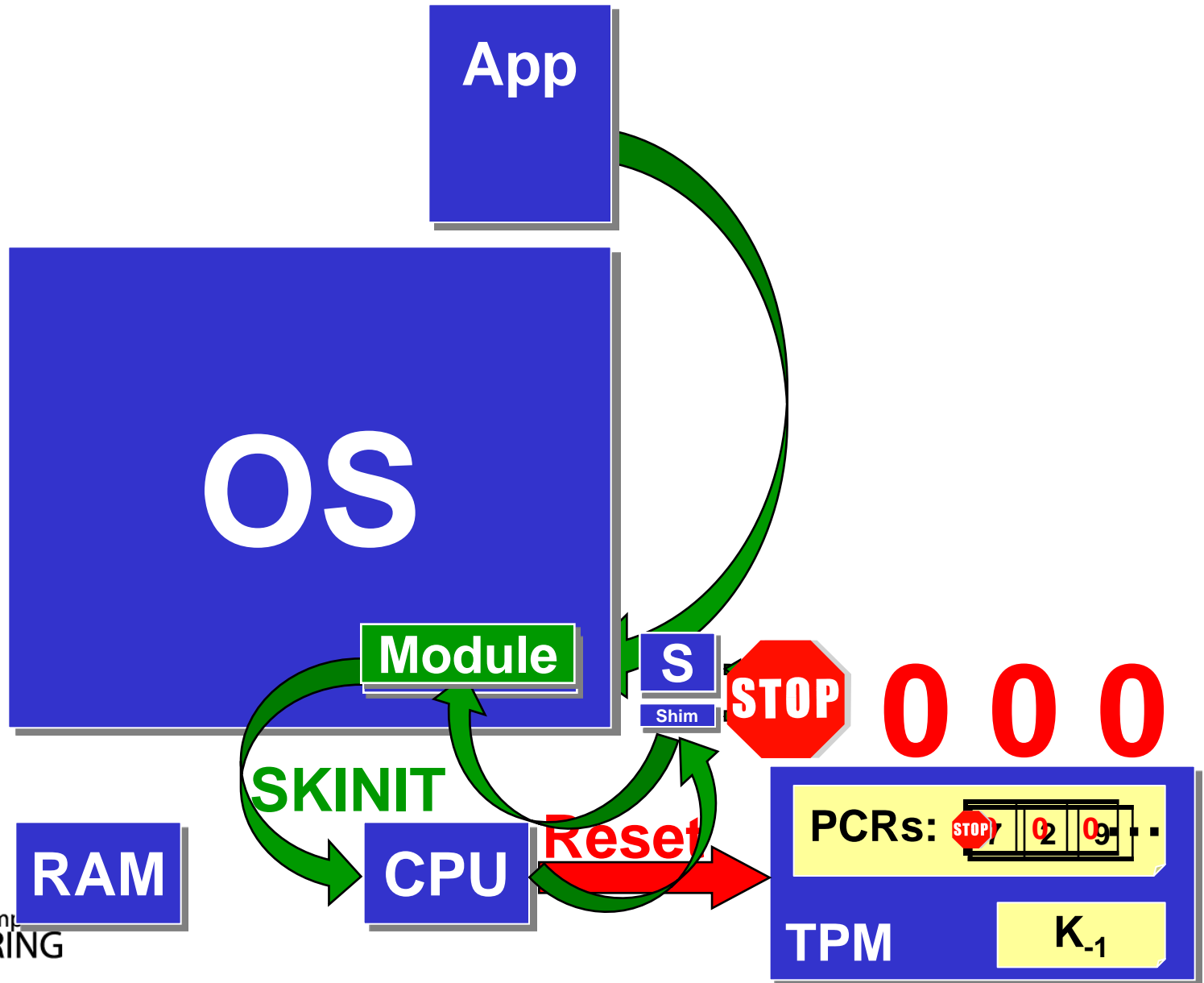
Basic Flicker Architecture

1. Pause current execution environment (legacy OS)
2. Execute security-sensitive code using *late launch*
3. Preserve session-state with TPM *sealed storage*
4. Resume previous environment

Not the intended use of late launch, sealed storage

- Intended use is an infrequent, disruptive event
 - Use to replace lowest-level system software
 - All but one CPU must be halted for late launch
- Flicker use resembles a context switch
 - Setup protected execution environment for sensitive application

Flicker Execution Flow



What is S?

- Self-contained code in an application
- Data secrecy and integrity requirements
- General-purpose computing
- Some examples
 - Manages a private key for web server or CA
 - Manages Access Control List (ACL)
 - Is a compute client in distributed setting

TCB Reduction with Flicker

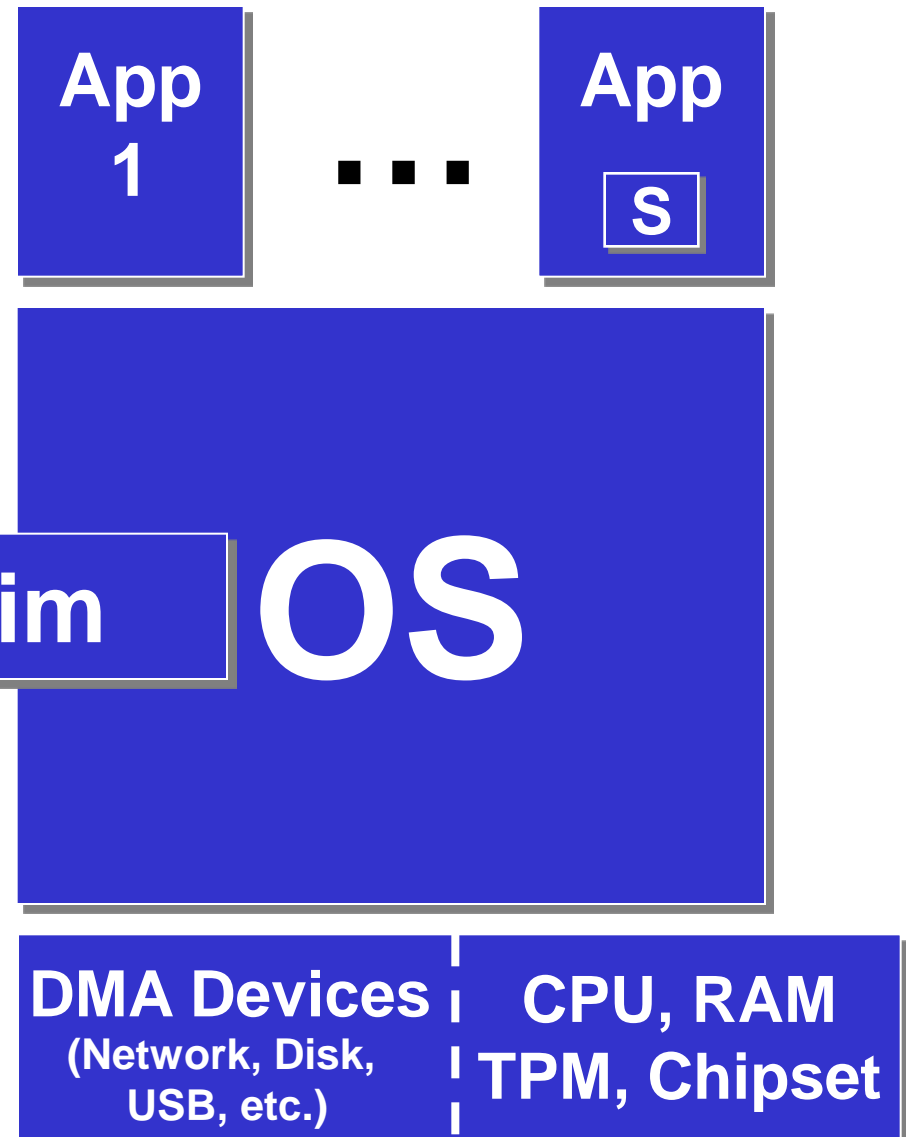
Today, TCB for sensitive code S:

- Includes App
- Includes OS
- Includes other Apps
- Includes hardware

With Flicker, S's TCB:

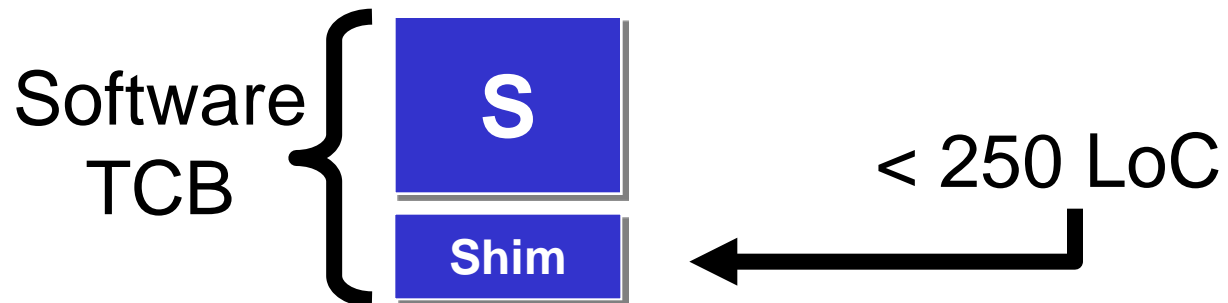
- Includes Shim
- Includes some hardware

*Adversary controls
everything else*



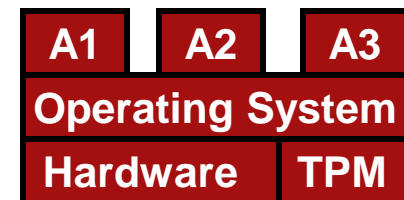
Properties of the Flicker System

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



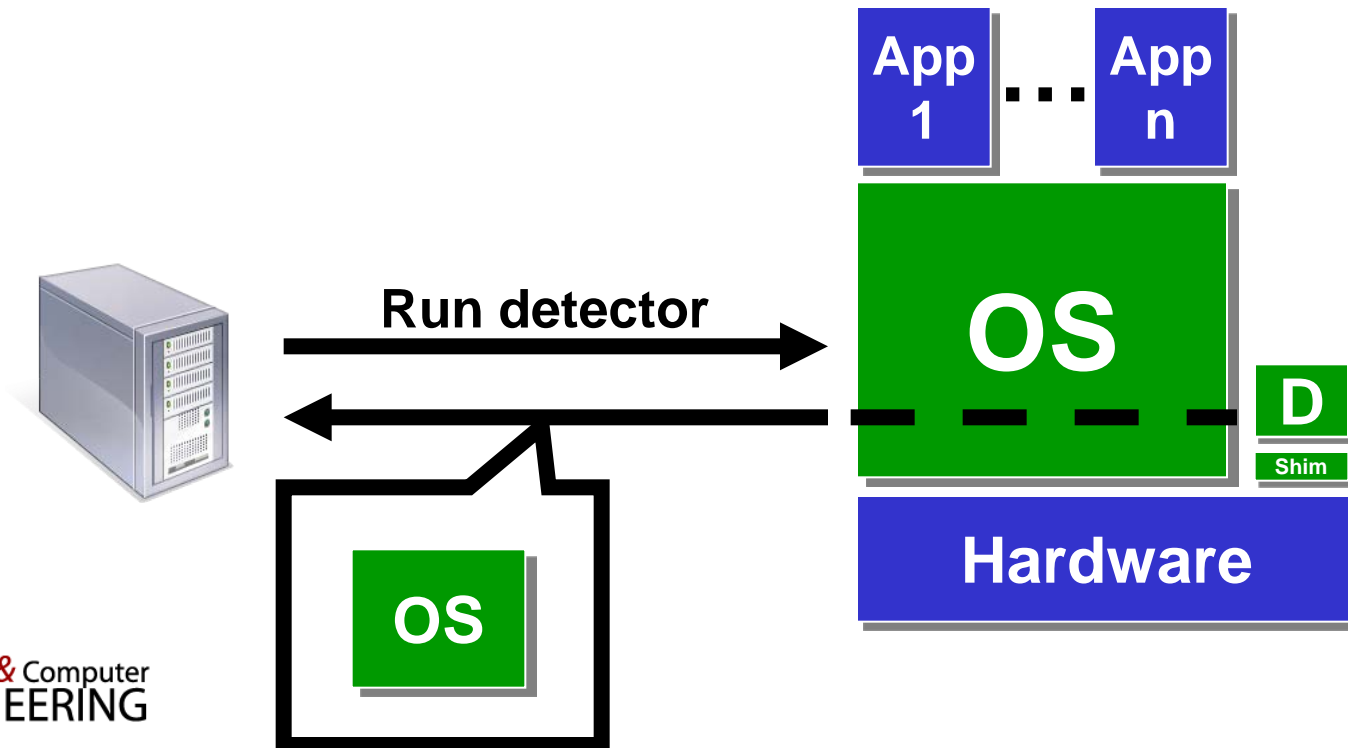
Recall: Shortcomings of Static Root of Trust

- Integrity measurements are done at load-time not at run-time
 - Time-of-check-time-of-use (TOCTOU) problem
- Coarse-grained, measures entire system
 - Every host is different: different firmware versions, different drivers, different patches, different apps, different spyware
 - What does a PCR mean in this context?
 - TCB includes entire system!
- No guarantee of execution



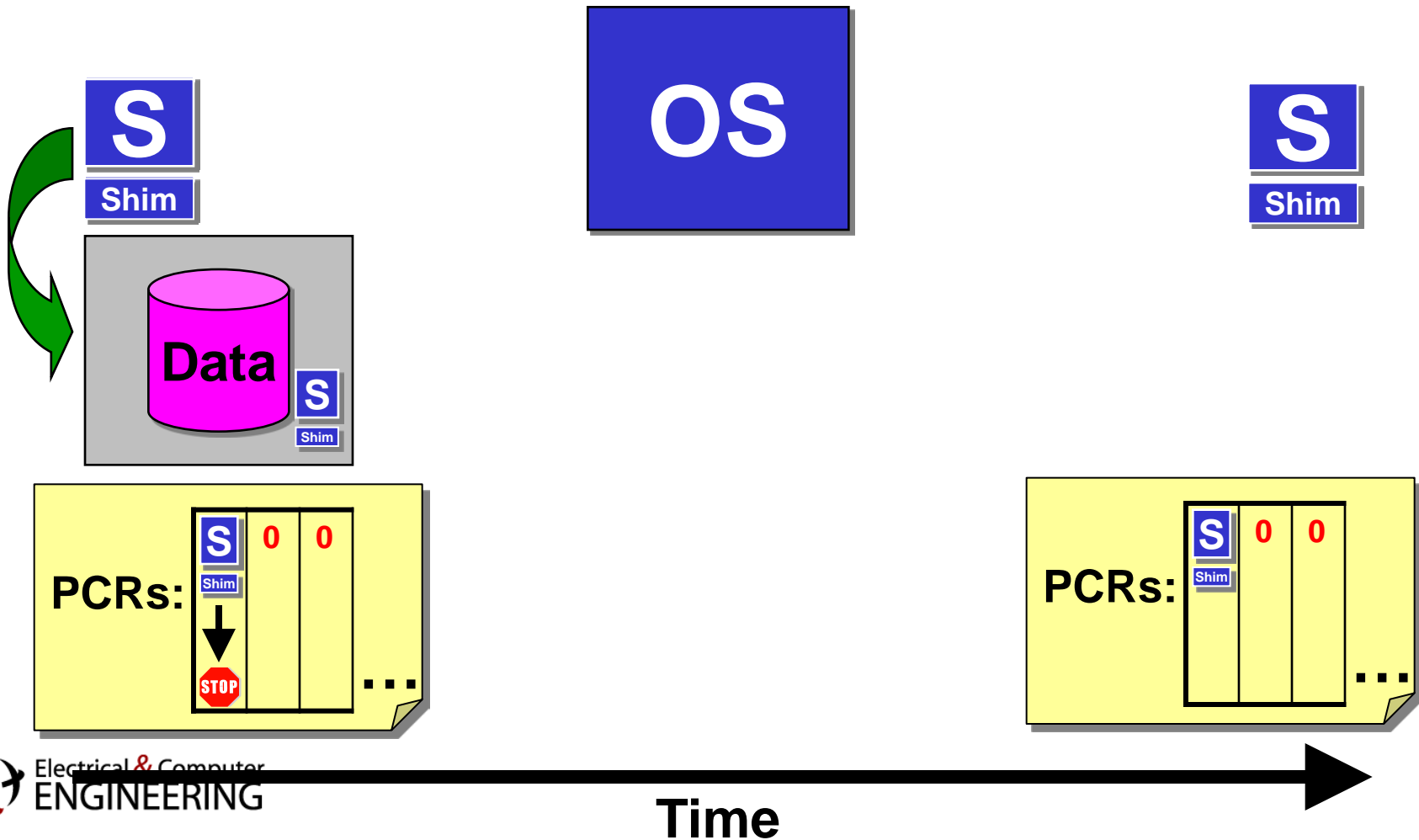
Application: Rootkit Detector

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



Flicker Context Switch with Sealed Storage

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



Limitations of Flicker

- Overhead of late launch is high for certain applications
 - Example: web server application S that performs sensitive signing operations and protects the signing key
 - TrustVisor: An alternative architecture that addresses this issue (optional reading)
- Only works for self-contained applications

Sealed Storage Details

Using PCR values after boot

- Step 1: `TPM_TakeOwnership(OwnerPassword, ...)`
 - Creates 2048-bit RSA Storage Root Key (SRK) on TPM
 - Cannot run `TPM_TakeOwnership` again without `OwnerPwd`
 - Done once by IT department or laptop owner.
- (optional) Step 2: `TPM_CreateWrapKey / TPM_LoadKey`
 - Create more RSA keys on TPM protected by SRK
 - Each key identified by 32-bit keyhandle

Protected Storage

- Main Step: Encrypt data using RSA key on TPM
 - **TPM_Seal** Returns encrypted blob; arguments:
 - keyhandle: which TPM key to encrypt with
 - KeyAuth: Password for using key `keyhandle`
 - PcrValues: PCRs to embed in encrypted blob
 - data block: at most 256 bytes (2048 bits)
- **Main point:** blob can only be decrypted with **TPM_Unseal** when PCR-reg-vals = PCR-vals in blob. *Embedding PCR values in blob ensures that only certain apps can decrypt data.*

Sealed storage: applications

- Lock software on machine:
 - OS and apps sealed with Master Boot Record's PCR.
 - Any changes to MBR (to load other OS) will prevent locked software from loading.
 - Prevents tampering and reverse engineering
 - e.g. software integrity on voting terminals
- Microsoft Bitlocker
 - Disk encryption
- Web server: seal server's SSL private key
 - Goal: only unmodified Apache can access SSL key

Trusted vs Trustworthy

- A trusted system or component is one whose failure can break the security policy
- A trustworthy system or component is one that will not fail
- Trusted Computing as used in the title of this lecture is really intended to mean “Towards Trustworthy Computing”

From “Trusted” to “Trustworthy”

- Trusted computing systems are useful to convince a third party that a “trusted” program P is running on a remote system
- We will discuss methods for establishing that P is “trustworthy” in the next two modules of the course
 - Model checking, static & dynamic analysis, type systems
 - Easier to apply if “ P ” is small and simple (i.e., smaller and simpler TCB is better)

Thanks

Acknowledgement

- Many of the slides used in this lecture are based on slides provided by Jon McCune and on slides from Dan Boneh and John Mitchell's CS155 computer security course at Stanford

Additional Details about Late Launch

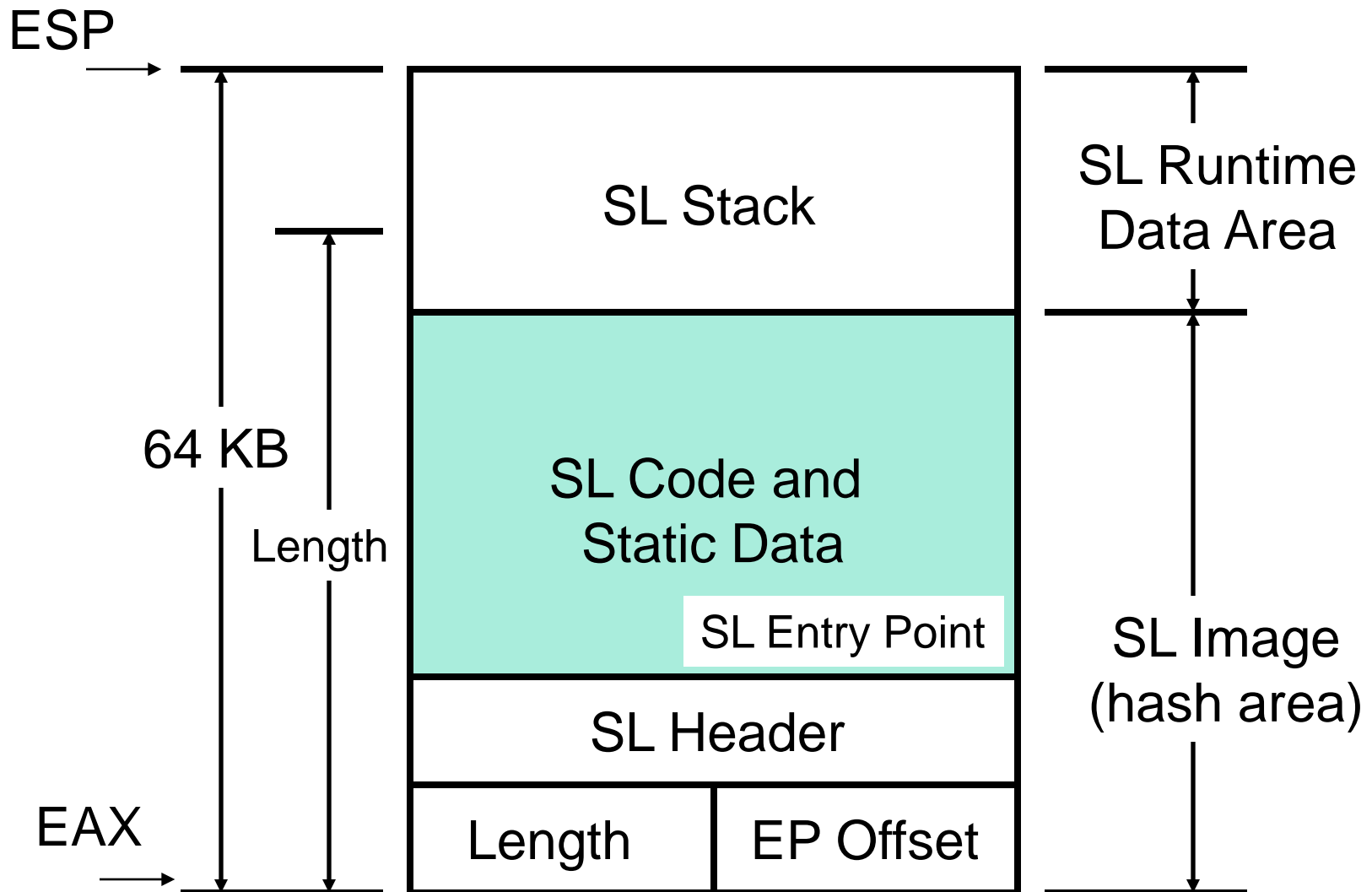
AMD Secure Virtual Machine

- Two categories of hardware support
 - Virtualization support
 - External DMA access 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 TPM Operations

- TPM v1.2 includes notion called **locality**
 - Similar to software privilege level
 - 4 is highest, 0 is lowest
 - Certain PCRs associated with localities
 - PCR 17 is associated with locality 4
 - SKINIT is the only locality 4 operation
- SKINIT sends contents of SLB to TPM
 - TPM hashes SLB to create a measurement
 - TPM **resets** PCR17, sets PCR17 = 0
 - Distinct from boot-time value of PCR17= -1
 - Allows verifier to know that SKINIT was executed
 - TPM performs PCR_Extend(17, hash(SLB))

Secure Loader Block Layout



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
 - “Late launch” or “dynamic root of trust” remove trusting the operating system, Bios, and even DMA-capable devices

