CIS 4360
Secure Computer Systems

Trusted Platform Module

Professor Qiang Zeng

Spring 2017

Some slides were stolen from Stanford's Security Course, Bruce Maggs, and Bryan Parno

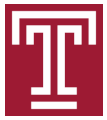
Previous Class

Does “a – b” concern you? How? What should you do?

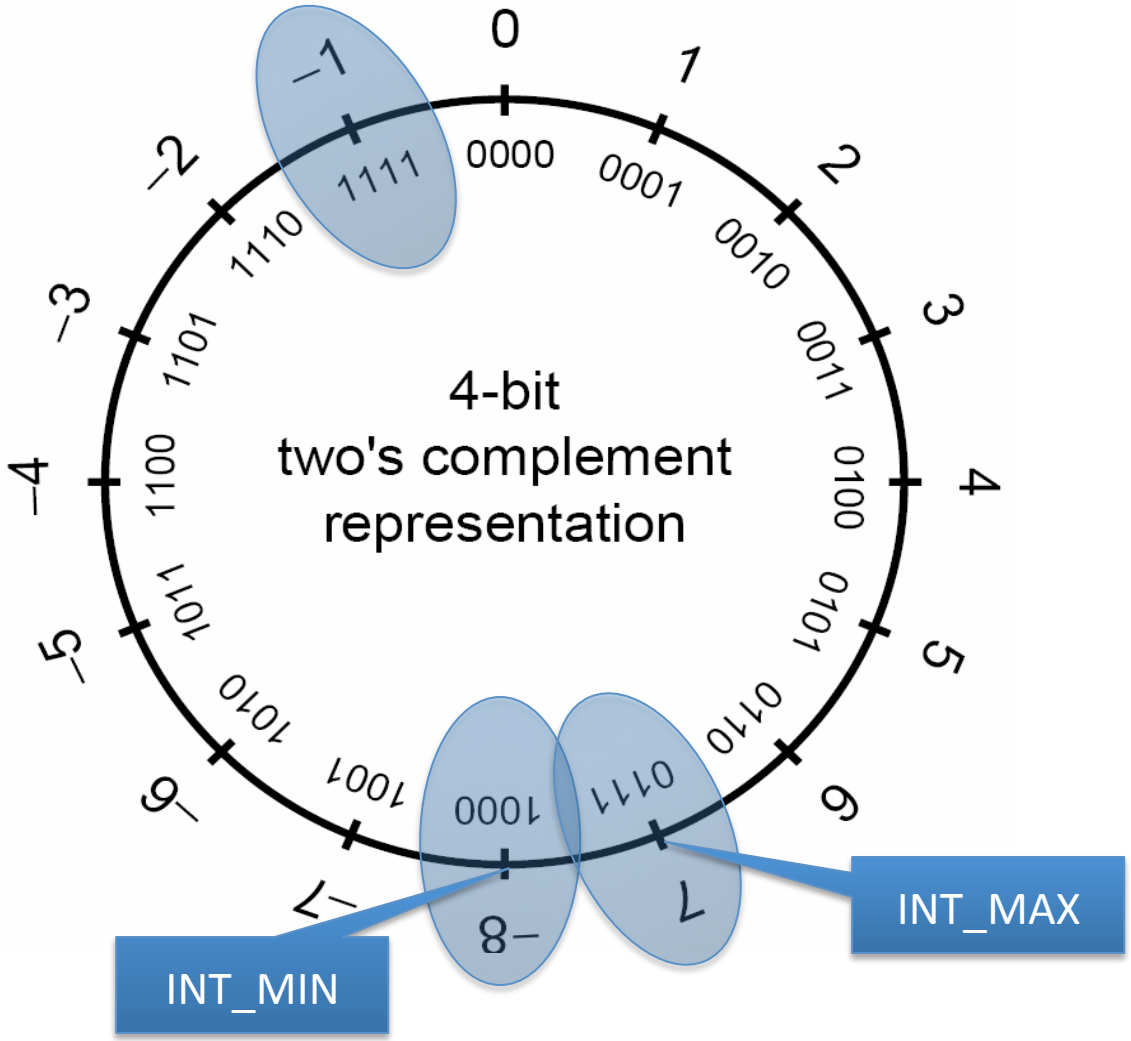
Yes, it may cause integer overflow.

E.g., assume a and b are of “int” type. a = INT_MIN and b = 1
then a – b = INT_MAX

```
if( (b>0 && a < INT_MIN + b) || (b <0 && a > INT_MAX + b) )
   // report overflow
else
  r = a – b;
```



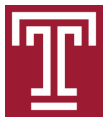
Signed Integer Representation



Previous Class

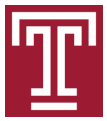
unsigned int x = -1; // What is x's value? Why?

The binary representation will be preserved. Since -1 is represented as 0xFFFFFFFF, which represents UINT_MAX for an unsigned integer



Re-cap about Conversion for “a = b”

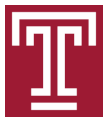
- A conversion to a type that can represent the value being converted is always well-defined
  - the value simply stays unchanged
  - E.g., signed -> larger signed; unsigned -> larger unsigned; unsigned short -> int
- A conversion to an unsigned type is always well-defined
  - y = x % (dst_type_max +1)
- A conversion to a signed type that cannot represent the value being converted results is implementation dependent
  - But in many systems: truncate the higher-order bits and re-interpret the remaining



Previous Class

short x = -1; int y = 0; // x > y? Why?

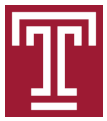
After integer promotion, i.e., int z = x, obviously z > y is false.



Previous Class

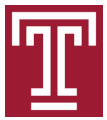
short x = -1; unsigned int y = 0; // x > y? why?

After integer promotion, i.e., int z = x; // z is still -1
Wrt z op y, since z and y have the same rank and y is unsigned, z will further be transformed to
an unsigned integer w = -1 // w is now UINT_MAX
Therefore, w > y is true and x > y is true.



Outline

- Format String Attacks
- TPM

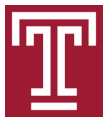


Vulnerable code example

- Is the following code vulnerable? Why?

```
void foo(char* user_provided) {
        printf(user_provided);
}
```

- Yes. E.g., user_provided can be “%08x %08x %08x %08x %08x”; this will leak information
- Why?



The format string can contain two types of data: printable characters and **format directives**, such as %d, %x, %u

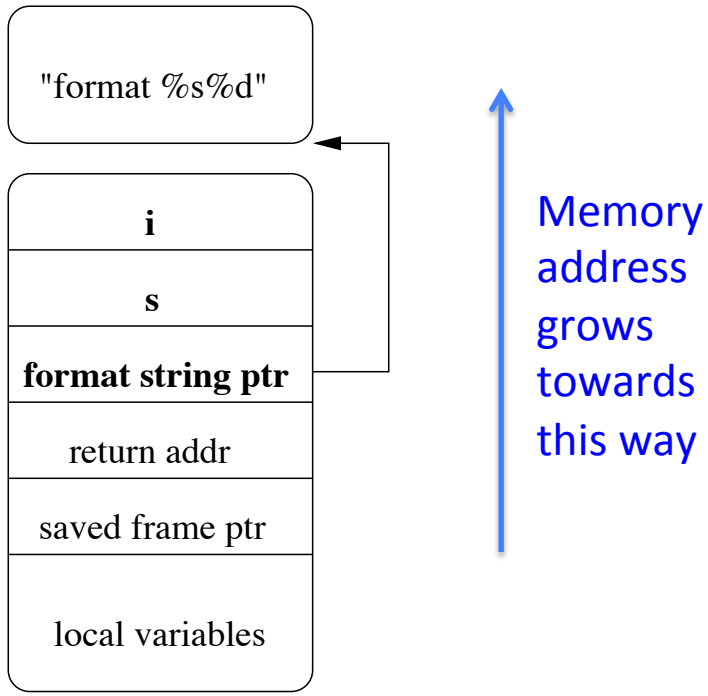
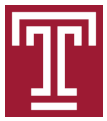
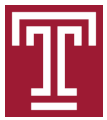


Figure 28. Activation record of *printf("format %s%d", s, i)* that has three parameters: the format string pointer, a string pointer, and an integer. A format function uses an internal stack pointer to access the parameters in the stack as it encounters the directives in the format string.



Width Specifier (specifies the minimum width)

- printf("%*d", 5, 10) will result in “   10” being printed, with a total width of 5 characters.
- printf(”%*d", -5, 10) will result in “10   ” being printed, with a total width of left-aligned 5 characters.
- Similarly,
- printf(“%5d", 10) and printf(“%-5d", 5, 10) can achieve the same effect
- printf(“%05d”, 10) prints “00010”
- printf("%#05x\n", 12); prints “0x00c” (# for 0x and the width of 0x00c is 10)

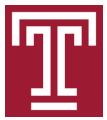


Attack one: viewing the memory

- Now you should be able to understand why the following code can leak information:

```
void foo(char* user_provided) {
        printf(user_provided);
}
```

- user_provided can be, e.g., “%08x %08x %08x %08x %08x” to leak information
- A special form of “buffer overflow”



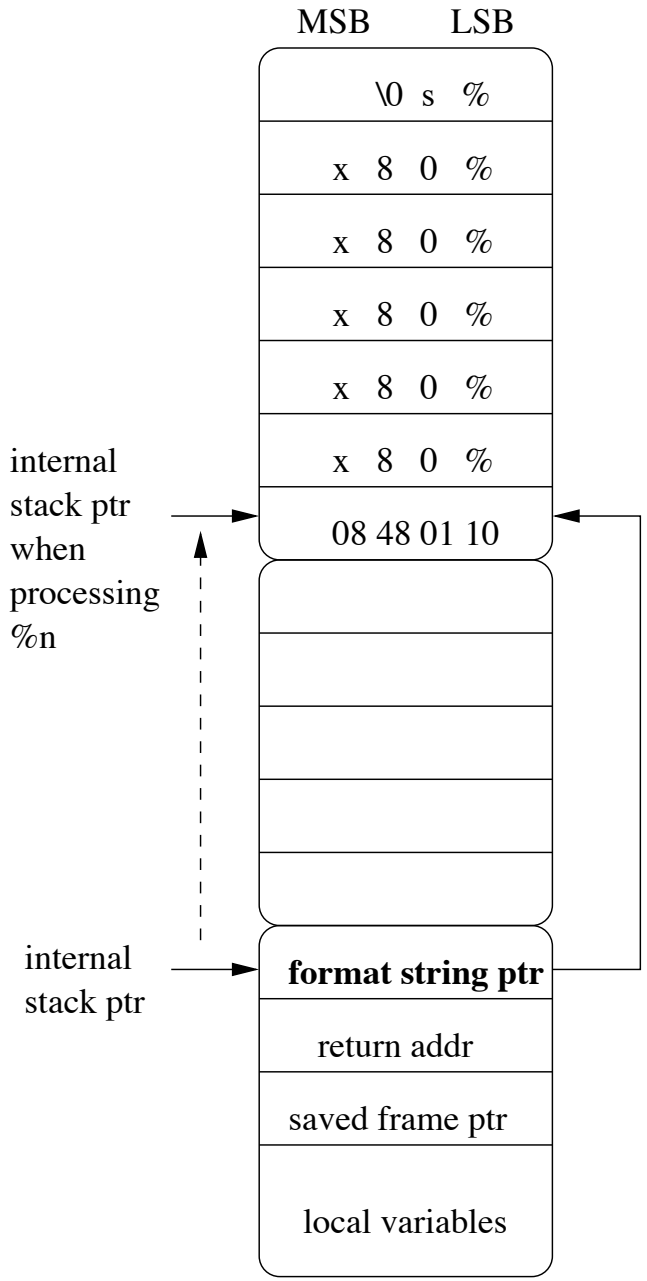
Attack two: Overwriting memory

The %n directive writes an integer value, the number of characters that is written by the format function so far, at the location pointed by the corresponding parameter. E.g.,

printf("\x10\x01\x48\x08%08x%08x %08x%08x%08x%n");

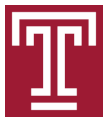
Writes 44 at 0x8480110 (four characters for "\x10\x01\x48\x08" and eight characters for each of the five "%08x")

You can make use of the Width Specifier, e.g., %10000d, to increase the value you want to write



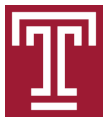
Attack three: Overflow a buffer

- char buf[100];
- // user-provided is “%104d\x80\x48\x55\66”
- sprintf(buf, user-provided)



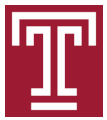
Suggestions

- Avoid using user-provided format string; try your best to use fixed predefined format string
- GCC can detect the mismatch (in terms of type and parameter number) between the directives and parameters when you use “-Wall or –Wformat”



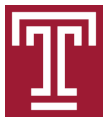
Writing Assignments

- Does “a – b” concern you? How? What should you do?
- unsigned int x = -1; // What is x’s value? Why?
- short x = -1; int y = 0; // x > y? Why?
- short x = -1; unsigned int y = 0; // x > y? why?

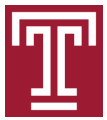


Outline

- Format String Attacks
- TPM

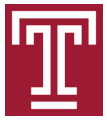


Bryan Parno's Travel Story



Attestation

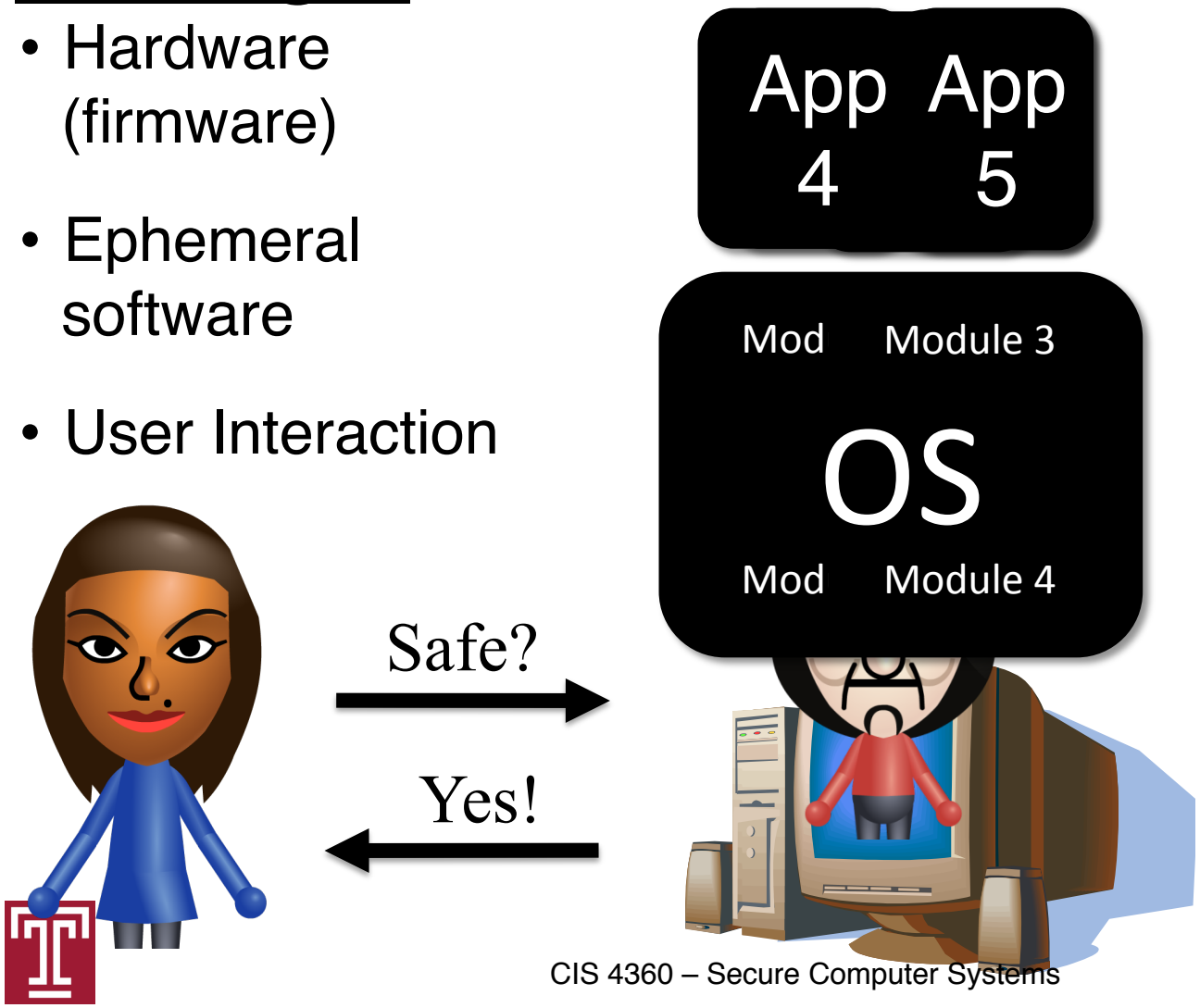
- How can we know that a system that we would like to use does not have KeyLogger installed?
- How does a judge determine whether a voting machine has been hacked by one of the president candidates (or not)?
- How do you (or the IT people) determine whether your computer has no rootkits installed?
- Attestation is the creation and validation of proof of some fact



Bootstrapping Trust is *Hard*!

Challenges:

- Hardware (firmware)
- Ephemeral software
- User Interaction



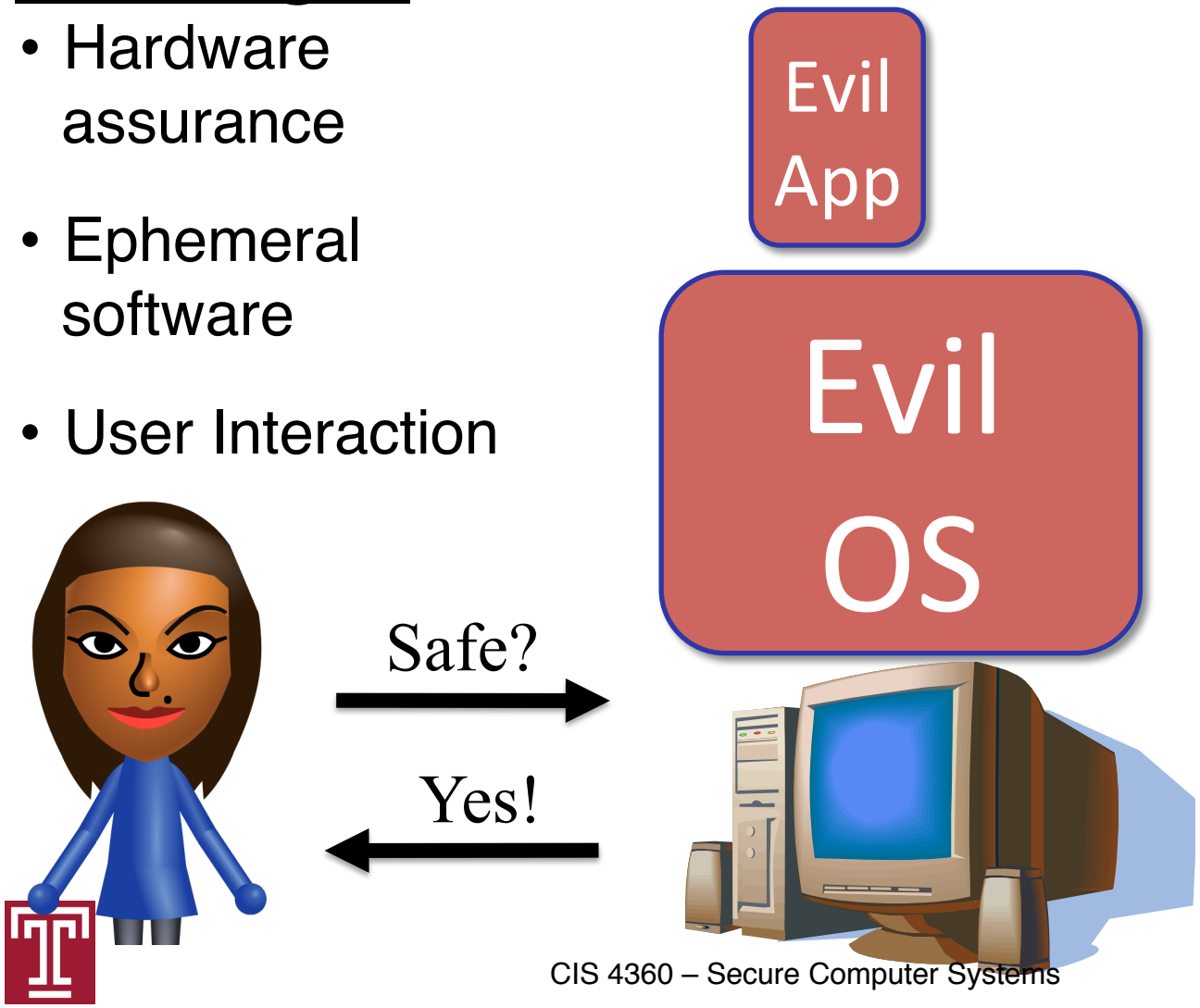
$S_{\pi_5}()$

$\hat{H}()$

Bootstrapping Trust is *Hard*!

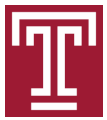
Challenges:

- Hardware assurance
- Ephemeral software
- User Interaction



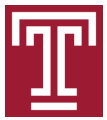
Background

- Trusted Computing Group: Founded in 1999 as TCPA.
  - Main players: (>200 members)
    AMD, HP, IBM, Infineon, Intel,
    Lenovo, Microsoft, Sun
- Goals:
  - **Hardware protected (encrypted) storage**:
    - Only "authorized" software can decrypt data (Digital Rights Management)
    - e.g.: protecting key for decrypting file system
  - **Secure boot**: to recognized "authorized" software
  - **Attestation**: Prove to remote server what software is running on my machine.



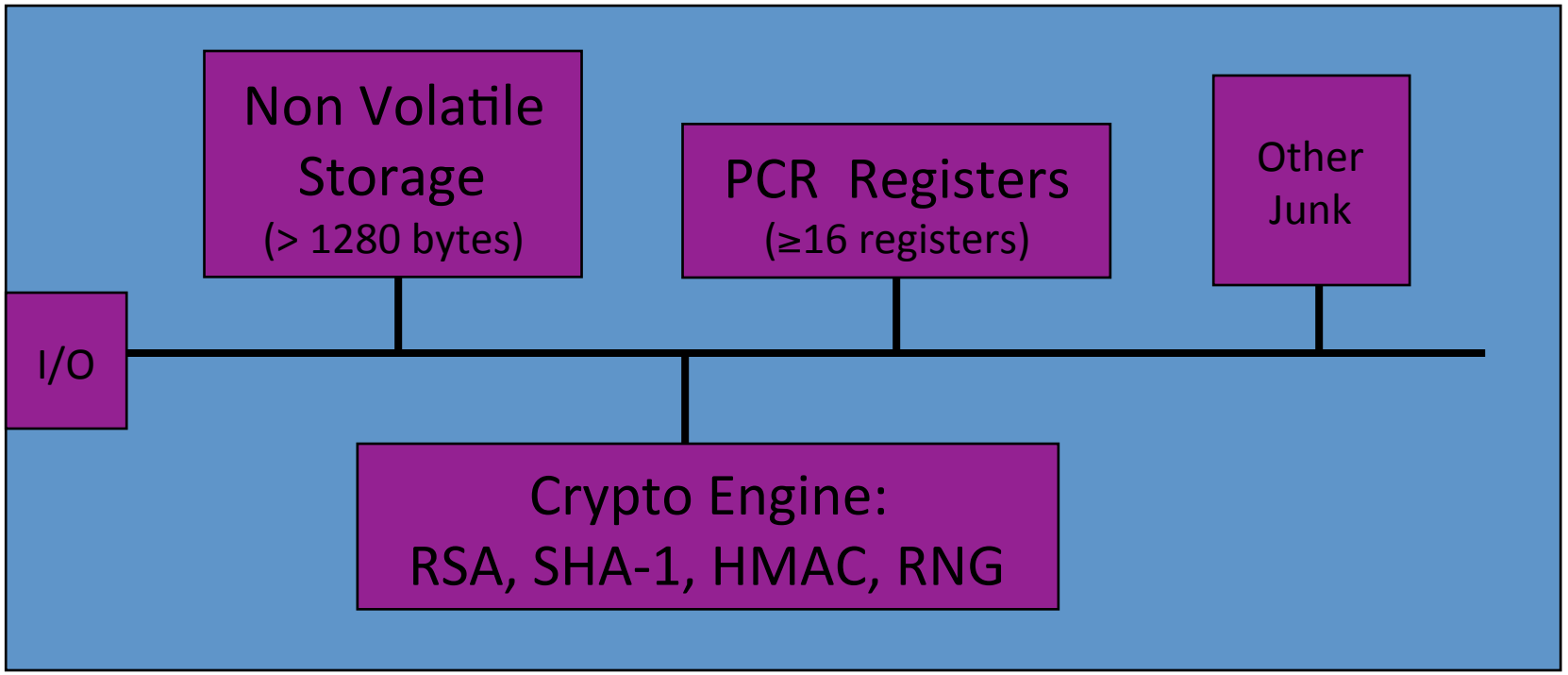
TCG: changes to PC or cell phone

- Extra hardware: **TPM**
  - ~ 1$
  - TPM Chip vendors:
    - Atmel, Infineon, National, STMicro
    - Intel D875GRH motherboard
- Software changes (depending on applications):
  - BIOS
  - OS
  - Apps



TPM Chip is in almost every PC

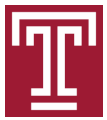
Components on TPM chip



RSA: 2048 bit modulus

SHA-1: Outputs 20 byte digest

RNG: Random Number Generator



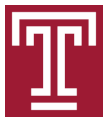
Non-volatile storage

1. Endorsement Key (EK) (2048-bit RSA)
   - Created at manufacturing time. Cannot be changed.
   - Used for "attestation" (described later)

2. Storage Root Key (SRK) (2048-bit RSA)
   - Used for implementing encrypted storage
   - Created after running
     TPM_TakeOwnership(OwnerPassword, …)
   - Can be cleared later with TPM_ForceClear from BIOS

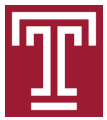
3. OwnerPassword (160 bits) and persistent flags

Private EK, SRK, and OwnerPwd never leave the TPM



PCR: the heart of the matter

- *PCR: Platform Configuration Registers*
  - Lots of PCR registers on chip (at least 16)
  - Register contents: 20-byte SHA-1 digest (+junk)
- Updating PCR #n :
  - TPM_Extend(n,D): PCR[n] ← SHA-1 (PCR[n] || D)
  - Important: You cannot "set" a PCR, you can only extend it
  - TPM_PcrRead(n): returns value(PCR(n))
- PCRs initialized to default value (e.g. 0) at boot time

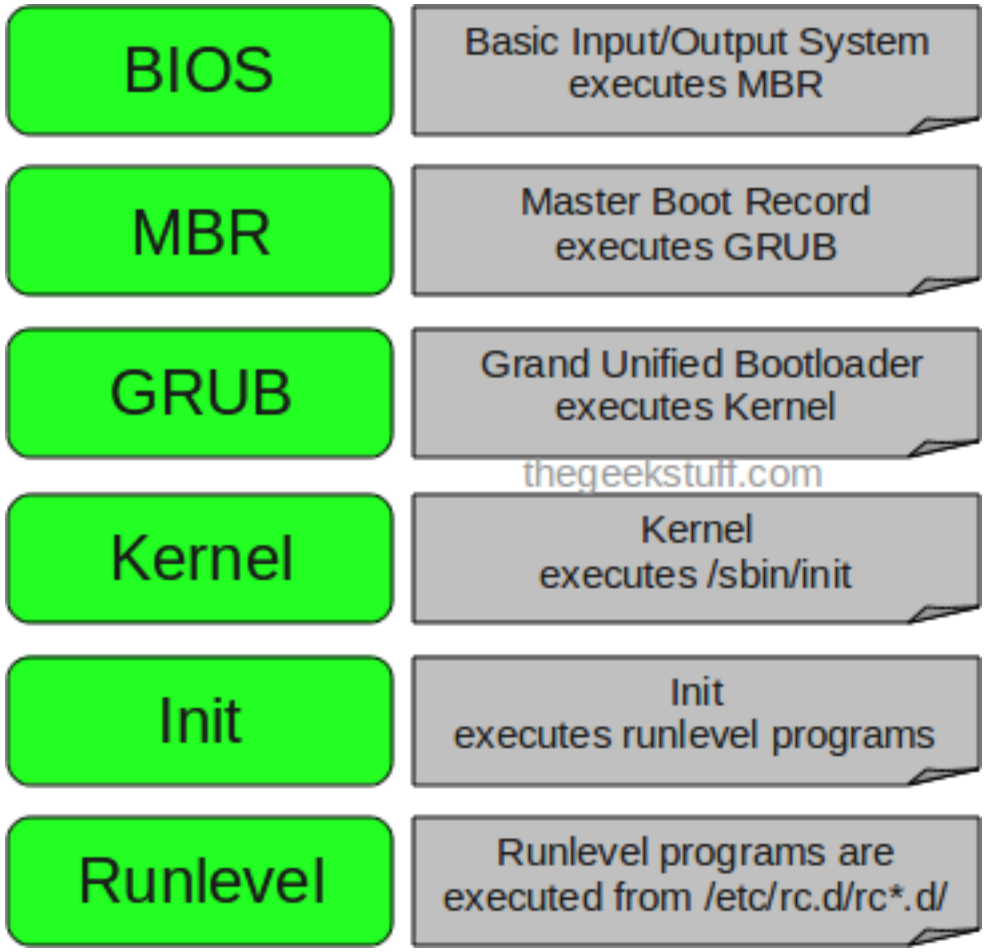


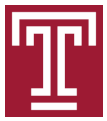
Overall, you can understand TPM as a passive component
(1) that does signing/encryption/decryption,
(2) that receives a value to extend a PCR, and
(3) that protects the private keys

It seems that it is pretty “dumb”…

Can we do something useful with a TPM?

Background: how a system boots

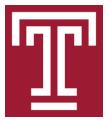




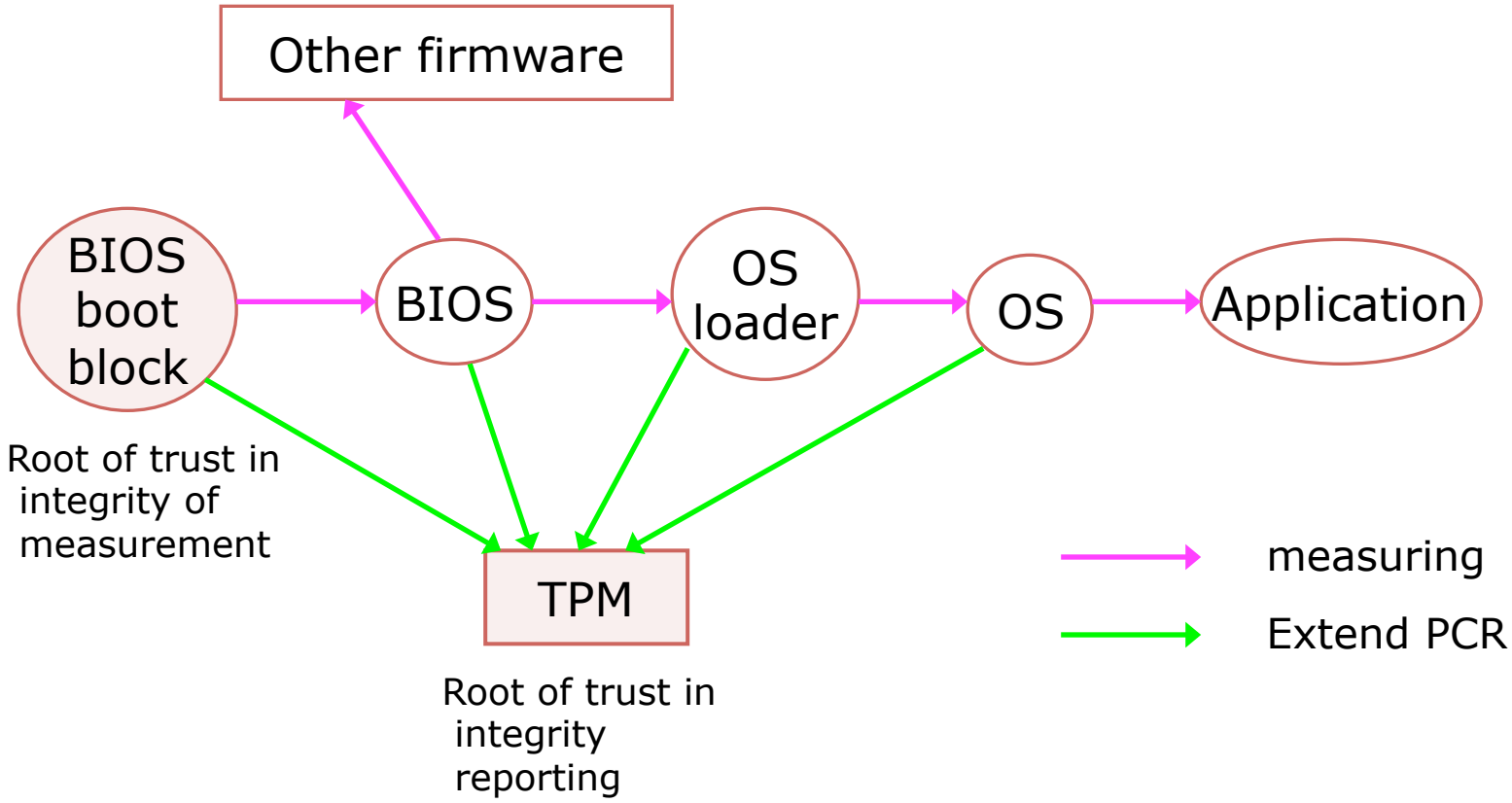
Using PCRs: the TCG boot process

- BIOS **boot block** executes
  - Calls TPM_Startup (ST_CLEAR) to initialize PCRs to 0
  - Calls PCR_Extend(n, <BIOS code>)
  - Then loads and runs BIOS post boot code
- BIOS executes:
  - Calls PCR_Extend(n, <MBR code>)
  - Then runs MBR (master boot record), e.g. GRUB.
- MBR executes:
  - Calls PCR_Extend(n, <OS loader code, config>)
  - Then runs OS loader

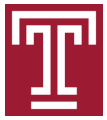
... and so on



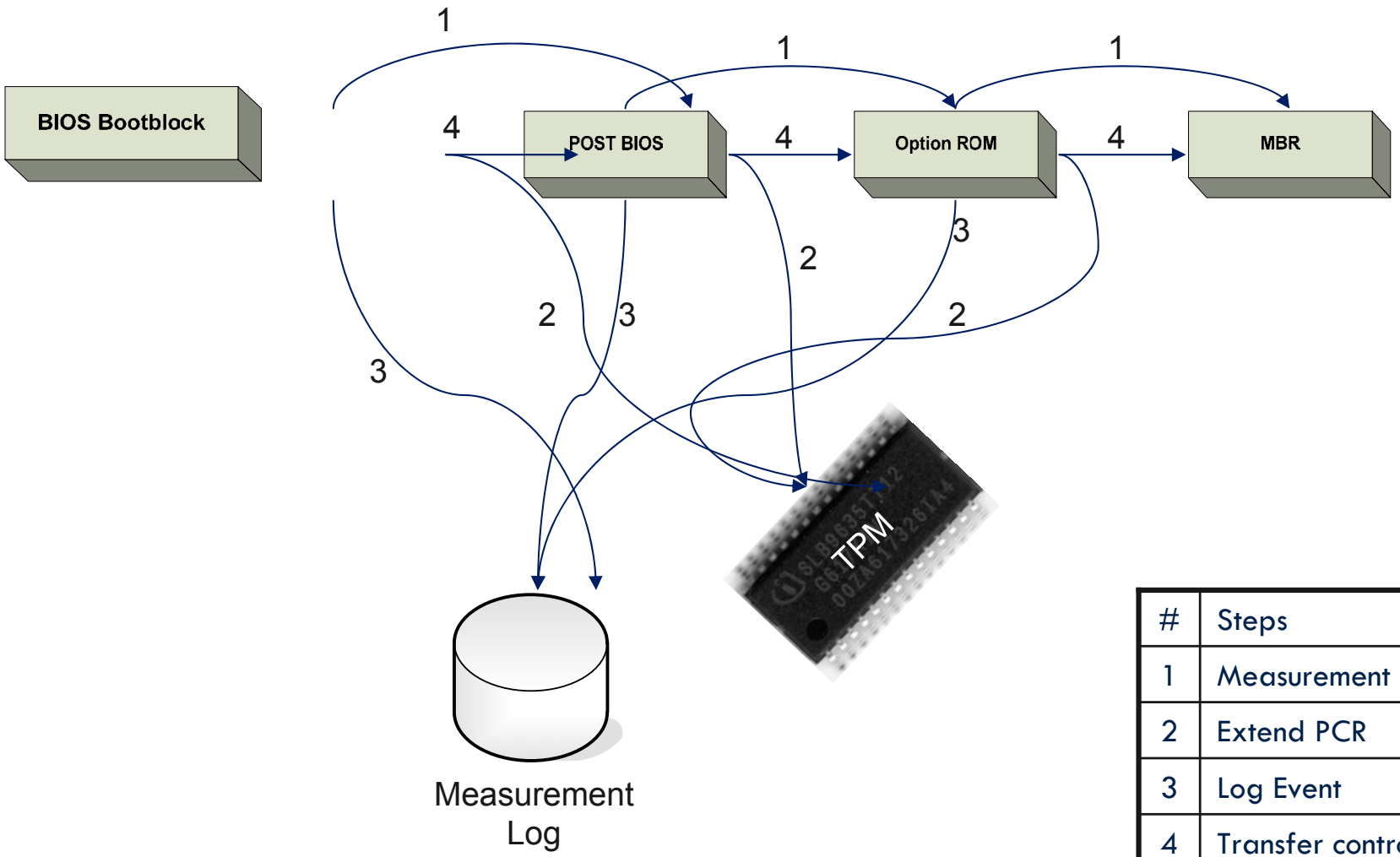
In a diagram



- After boot, PCRs contain hash chain of booted software
- Properties of SHA-1 prevent manipulation



Measured Boot



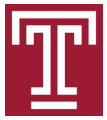
#	Steps
1	Measurement
2	Extend PCR
3	Log Event
4	Transfer control

Attestation: what it does

- **Goal**: prove to remote party what software is running on my machine.

- Good applications:
  - Bank allows money transfer only if customer's machine runs "up-to-date" OS patches.
  - Enterprise allows laptop to connect to its network only if laptop runs "authorized" software
  - Quake players can join a Quake network only if their Quake client is unmodified.

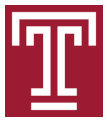
- DRM:
  - MusicStore sells content for authorized players only.



Attestation: how it works

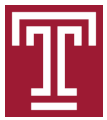
- Recall: EK (Endorsement Key) private key on TPM.
  - Cert for EK public-key issued by TPM vendor
  - The attestation initiator has the public key of all TPM vendors and thus can verify the TPM's cert

- Step 1: Create Attestation Identity Key (AIK)
  - AIK Private key known only to TPM
  - AIK public cert issued only if EK cert is valid



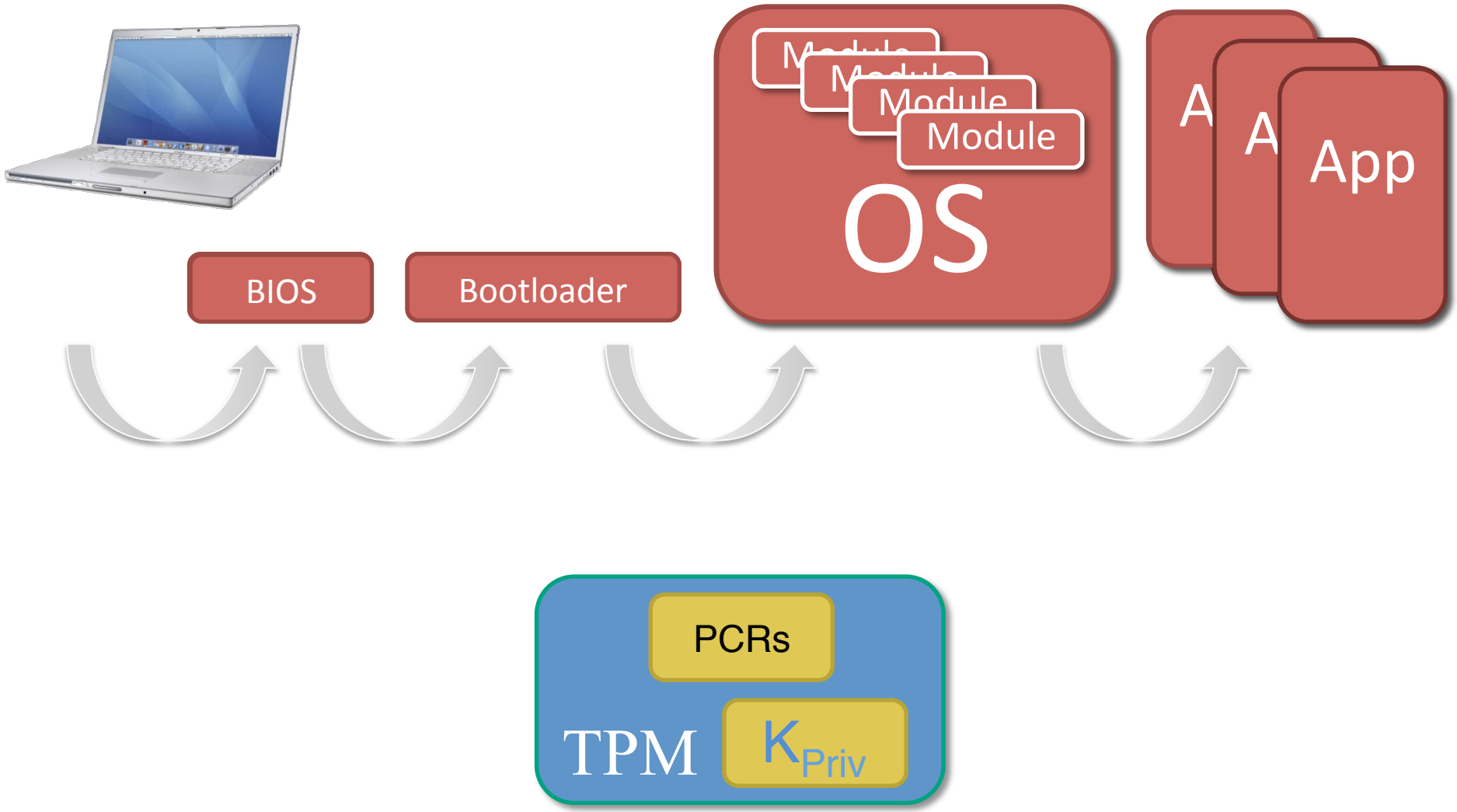
Attestation: how it works

- Step 2: sign PCR values (after boot)
  - Call TPM_Quote (some) Arguments:
    - keyhandle: which AIK key to sign with
    - KeyAuth: Password for using key `keyhandle'
    - PCR List: Which PCRs to sign.
    - Challenge: 20-byte challenge from remote server
      - Prevents replay of old signatures.
    - Userdata: additional data to include in sig.
  - Returns signed data and signature.



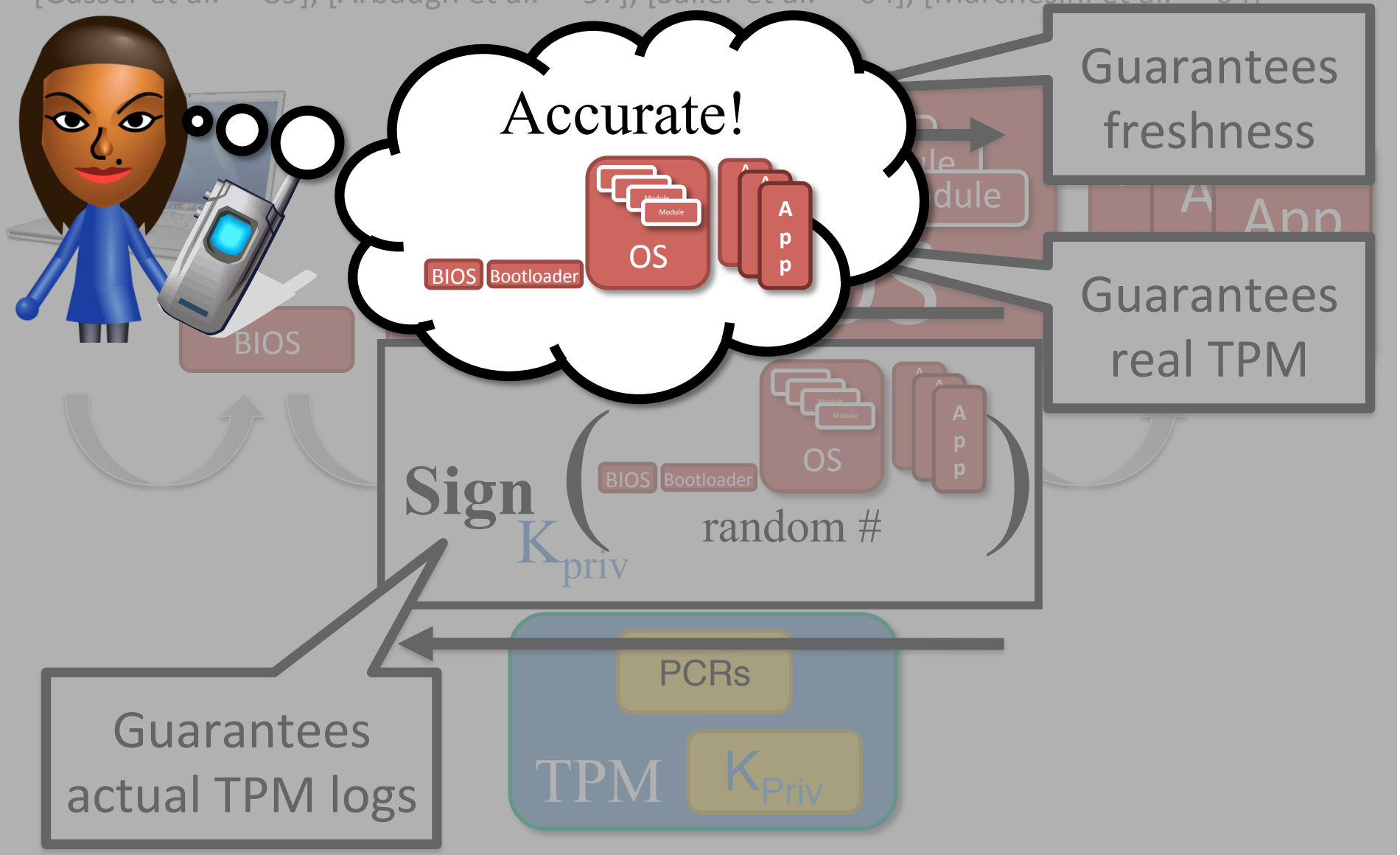
TPM-Based Attestation Example

[Gasser et al. ‘89], [Arbaugh et al. ‘97], [Sailer et al. ‘04], [Marchesini et al. ‘04]



Establishing Trust via a TPM

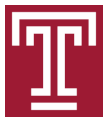
[Gasser et al. '89], [Arbaugh et al. '97], [Sailer et al. '04], [Marchesini et al. '04]



We have covered Attestation;
next, we will cover another main application: Sealed Data

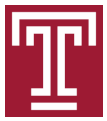
Sealing

- Sealing: TPM encrypts a pieced of data using a private key of TPM and records the PCR values upon encryption the client is interested in
  - TPM then returns an encrypted blob = Encrypt(K, {data, PCR values})
  - The encrypted blob is NOT stored in TPM
- Unsealing: The encrypted blob is sent to TPM for decryption, but TPM will refuse to release the clear text unless it finds the current values in PCR registers match those recorded in the blob



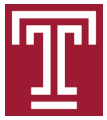
How to seal a key?

- Main Step: Encrypt data (e.g., a key)
  - TPM_Seal (some) Arguments:
    - keyhandle: the index of a TPM key used to encrypt the data
    - KeyAuth: Password for using key `keyhandle'
    - PcrValues: PCRs to be embedded in encrypted blob
    - data block: at most 256 bytes (2048 bits), e.g., an AES key
  - Returns encrypted blob, which will be stored in disk
- In BitLocker, an AES key is used to encrypt the disk in BitLocker. The AES key is sealed in a blob, which is stored in the disk
- The AES key will not be released by TPM unless the PCR values match those recorded in the blob



What is Measured?

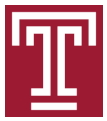
- Core Root of Trust of Measurement (CRTM), BIOS, and Platform Extensions (PCR 0)
- Option ROM Code (PCR 2)
- Master Boot Record (MBR) Code (PCR 4)
- NTFS Boot Sector (PCR 8)
- NTFS Boot Block (PCR 9)
- Boot Manager (PCR 10)
- BitLocker Access Control (PCR 11)



Question

Is it possible that an attacker installs some malicious firmware used to sniffer the AES key released by TPM?

No, if the firmware is infected, the corresponding PCR value will change. So TPM will simply refuse to release the AES key

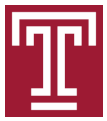


Question

A disk is protected by BitLocker +TPM. Can I insert it into another computer to get the data in the disk?

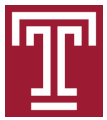
No, unless the TPM in the second computer has the same Storage Root Key. Recall that the SRK is needed to decrypt the encrypted blob stored in the disk

In this case, what you need is the “recovery key”, which is the AES key used to encrypt the disk. The user is asked to store the recovery key in a another place (e.g., a USB drive)



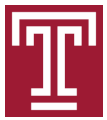
Summary

- What a TPM is:
  - Hardware protected keys
    - Endorsement key: for signing (PCR values)
    - Storage root key: for encrypting storage keys
  - Crypto-processor
    - RSA
    - HMAC
    - SHA-1 (SHA-2 in TPM 2.0)
  - PCR
- What the TPM can do
  - Attestation
  - Sealed storage

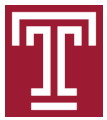


Writing Assignments

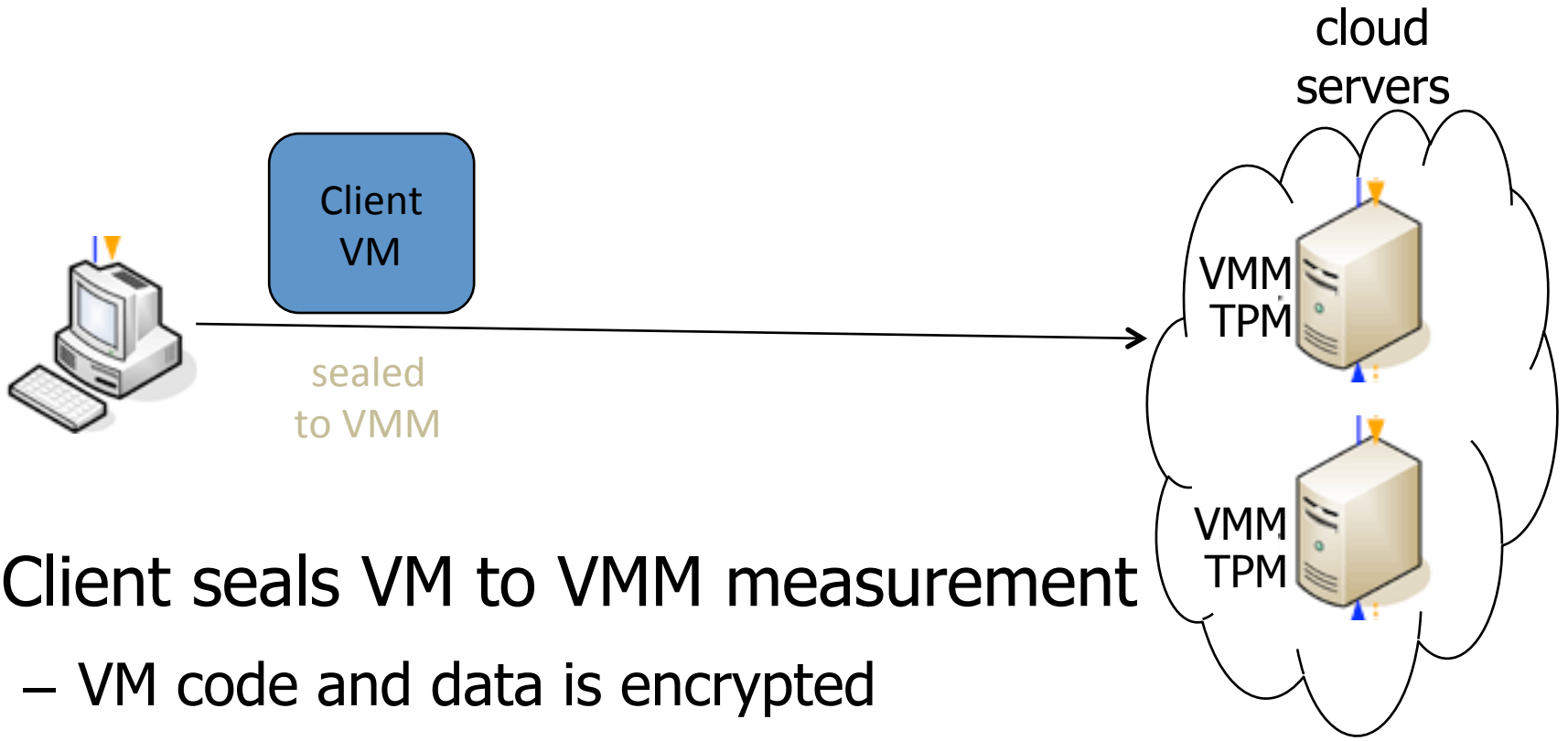
- What is Measured Boot?
- What attacks can you do if you have extracted the private keys from a TPM
- What knowledge should an IT service have for attestation over a laptop of the company?



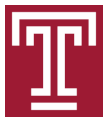
Backup slides…



A cloud application [JPBM' 10]

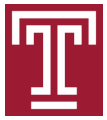


- Client seals VM to VMM measurement
  - VM code and data is encrypted
  - Can only be decrypted on valid cloud server
  - Cloud operator cannot easily access data



BitLocker Driver Encryption

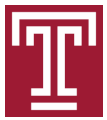
- Volume Master Key (VMK) encrypts disk
  - More precisely, VMK encrypts the Full Volume Encryption Key, which in turns encrypts disk
- VMK is sealed (encrypted) under TPM SRK (Storage Root Key) using
  - Master Boot Record (MBR) Code (PCR 4),
  - NTFS Boot Sector (PCR 8),
  - NTFS Boot Block (PCR 9),
  - NTFS Boot Manager (PCR 10), and
  - Other Critical Components (PCR 11)



1. Attesting to code integrity only

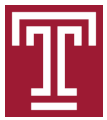
- Attestation only attests to what code was loaded.

- Does not say whether running code has been compromised.
  - Ineffective against buffer overflows



2. AV gets confused by encrypted viruses

- Suppose malicious music file exploits bug in Windows Media Player.
  - Music file is encrypted.
  - TCG prevents anyone from getting music file in the clear.
  - Can anti-virus companies block virus without ever seeing its code in the clear?

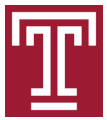


TCG Alternatives

- IBM 4758: Supports all TCG functionality and more.
  - Tamper resistant 486 100MhZ PCI co-processor.
  - Programmable.
  - … but expensive ~ $2000. TPM ~ $1.

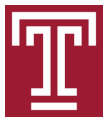
- AEGIS System: Arbaugh, Farber, Smith ’97:
  - Secure boot with BIOS changes only.
  - Cannot support sealed storage.
  - **Phoenix TrustConnector 2**

- SWATT: Seshadri et al., 2004
  - Attestation w/o extra hardware
  - Server must know precise HW configuration



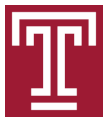
How to configure a TPM

- 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.



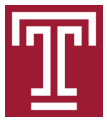
Sealed storage: applications

- Web server: seal server's SSL private key
  - Goal: only unmodified Apache can access SSL key
  - Problem: updates to Apache or Apache config
- Lock software on machine:
  - OS and apps sealed with MBR'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

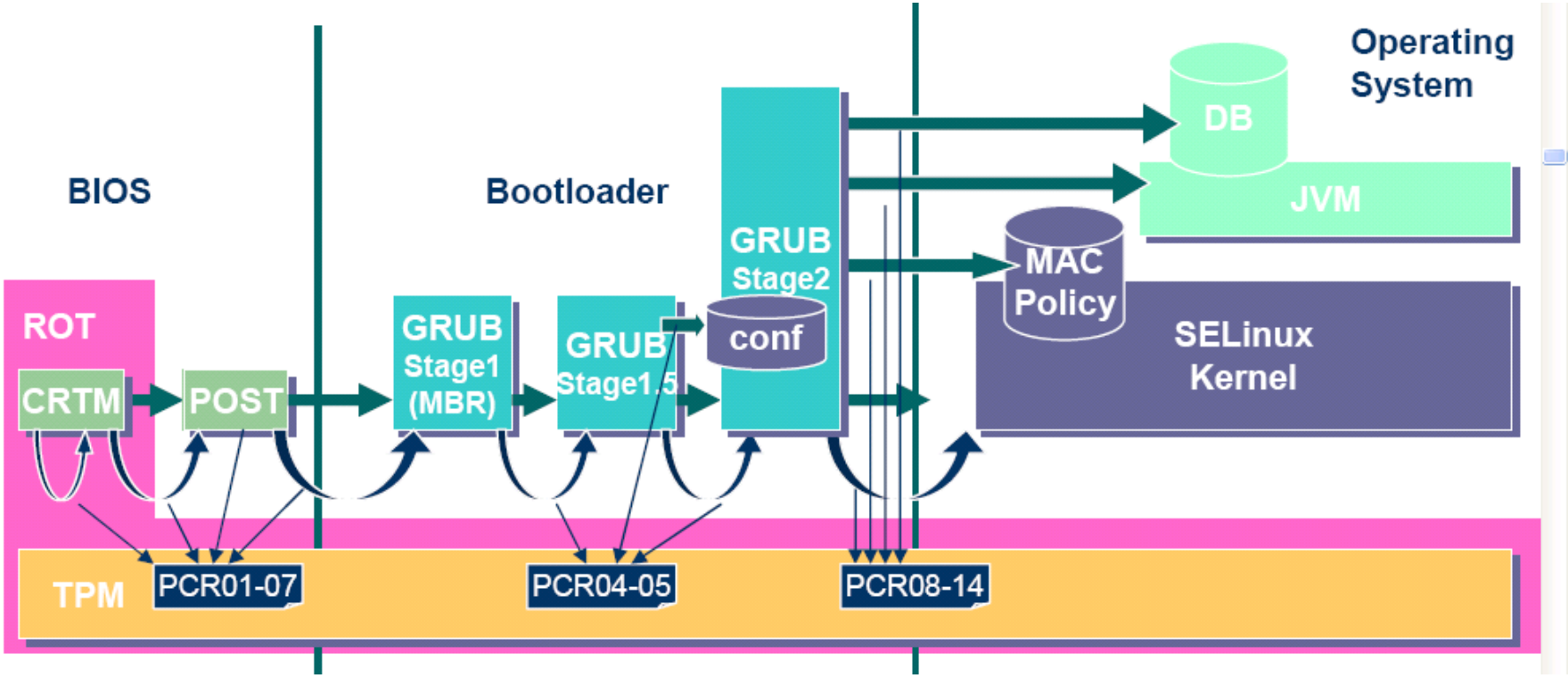


Better root of trust

- DRTM – Dynamic Root of Trust Measurement
  - AMD: **skinit** Intel: **senter**
  - Atomically does:
    - Reset CPU. Reset PCR 17 to 0.
    - Load given Secure Loader (SL) code into I-cache
    - Extend PCR 17 with SL
    - Jump to SL
- BIOS boot loader is no longer root of trust
- Avoids TPM_Init attack: TPM_Init sets PCR 17 to -1



Example: Trusted GRUB (IBM' 05)



What PCR # to use and what to measure specified in GRUB config file