

Security Hardware

Information Assurance

Fall 2006

Reading Material

- Intel Pentium II Software Developer's Manual: Volume 3. Sections 4.5 through 4.8
 - <http://developer.intel.com/design/pentium4/ma>
- TCG Specification Architecture Overview. Section 4 through 4.4.
 - <https://www.trustedcomputinggroup.org/groups>

Motivation

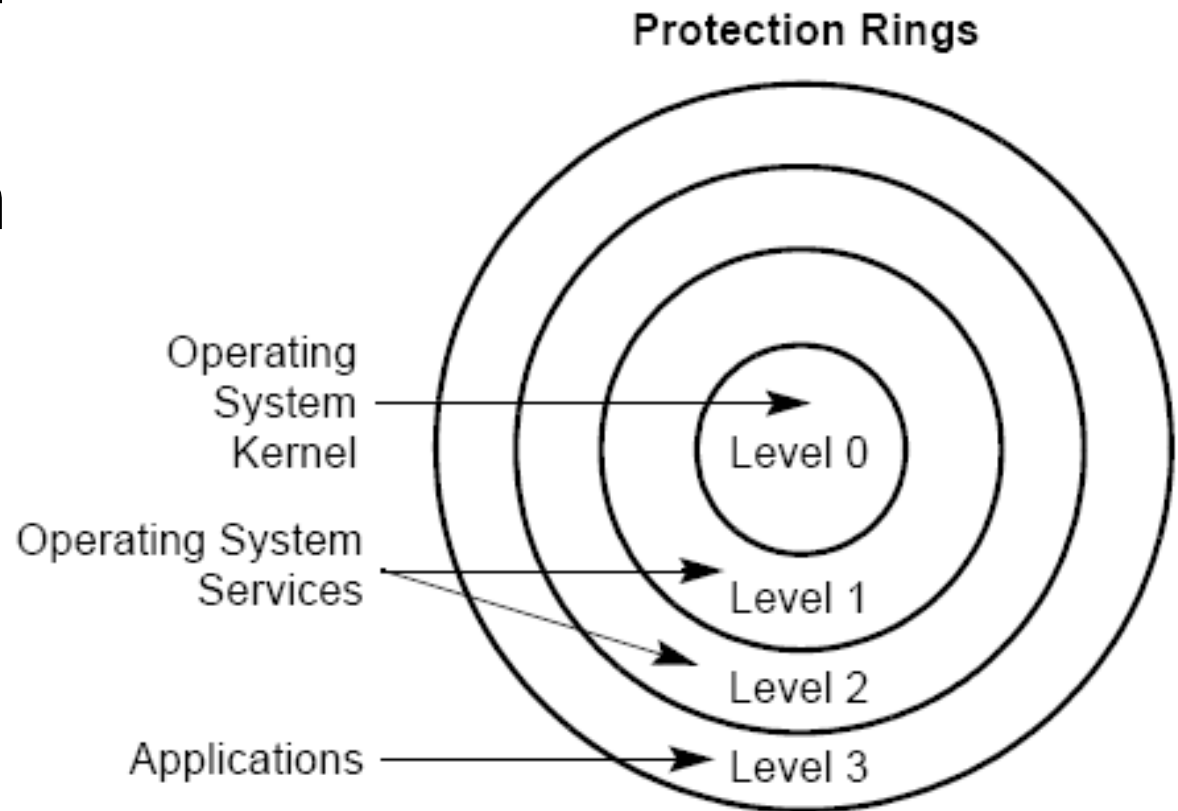
- As CS folks we have concentrated on security abstractions or software implementations
- Judicious use of security specific HW is beneficial
 - Feature Restriction
 - Physical separation
 - Performance benefits

Outline

- Memory ring protection architecture
- No execute bits
- Smart Cards and Trusted Platform Modules

Memory Protection Rings

- Originally in Multics
- In Intel arch since x386

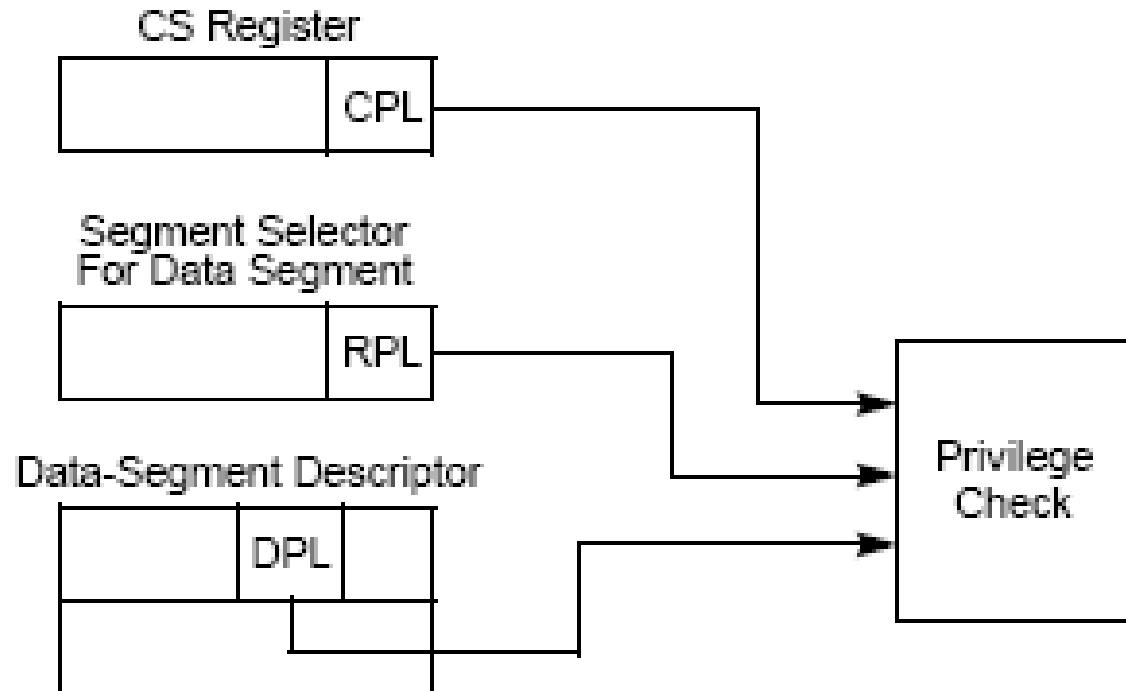


Privilege Levels

- CPU enforces constraints on memory access and changes of control between different privilege levels
- Similar in spirit to Bell-LaPadula access control restrictions
- Hardware enforcement of division between user mode and kernel mode in operating systems
 - Simple malicious code cannot jump into kernel space

Data Access Rules

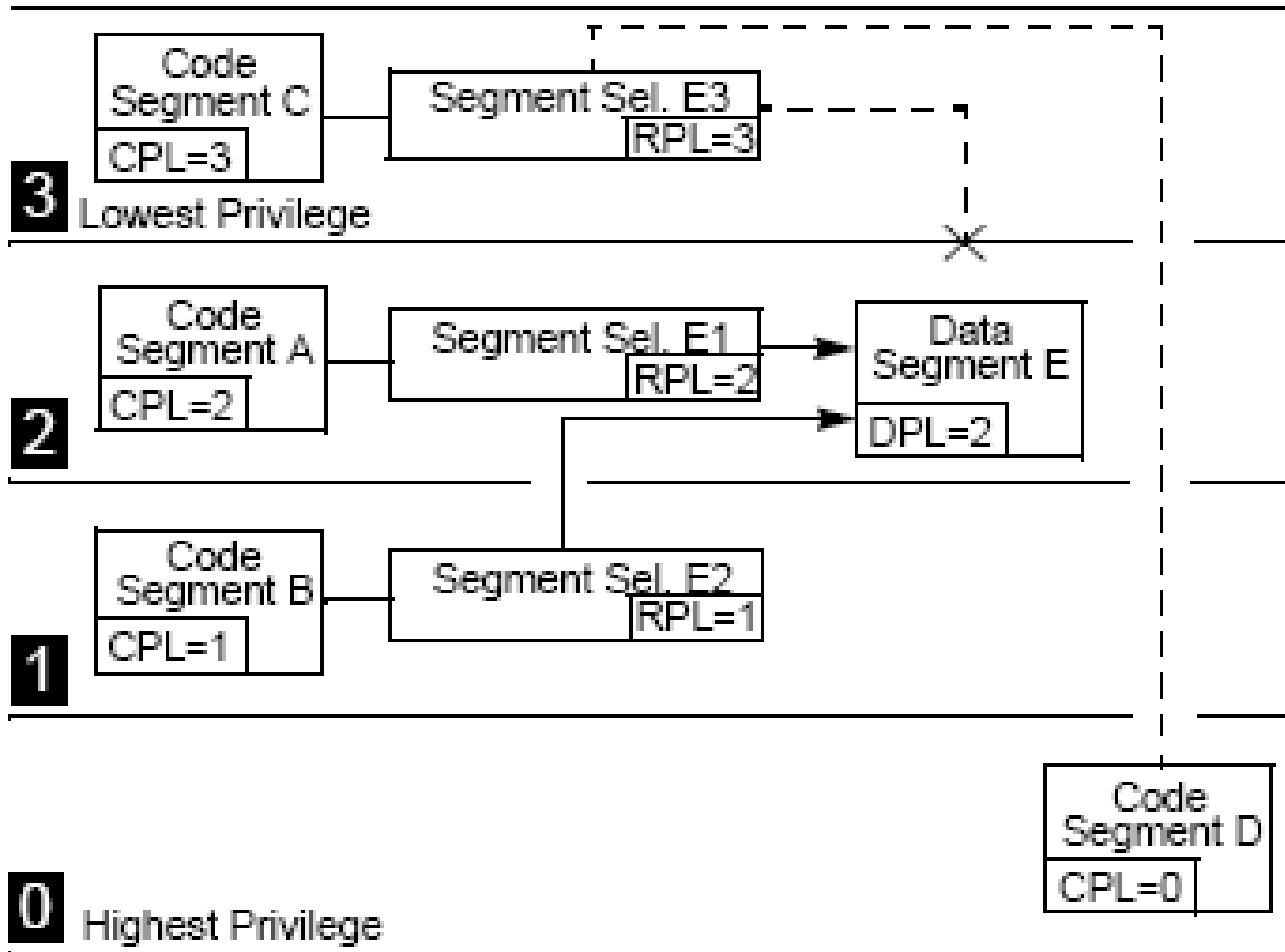
- Access allowed if
 - $CPL \leq DPL$ and $RPL \leq DPL$



Data Access Rules

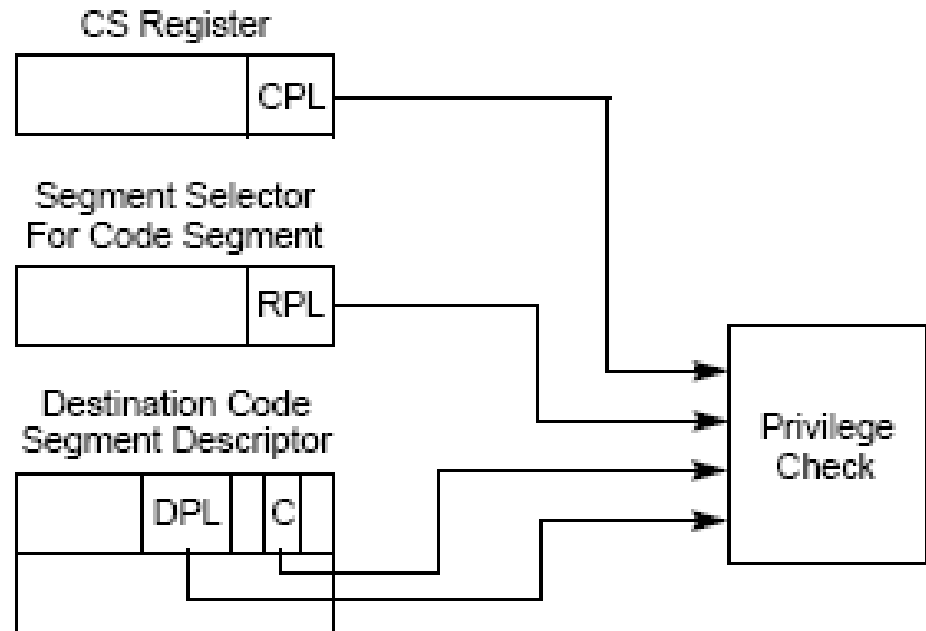
- Three players
 - Code segment has a current privilege level CPL
 - Operand segment selector has a requested privilege level RPL
 - Data Segment Descriptor for each memory includes a data privilege level DPL
- Segment is loaded if $CPL \leq DPL$ and $RPL \leq DPL$
 - i.e. both CPL and RPL are from more privileged rings

Data Access Examples

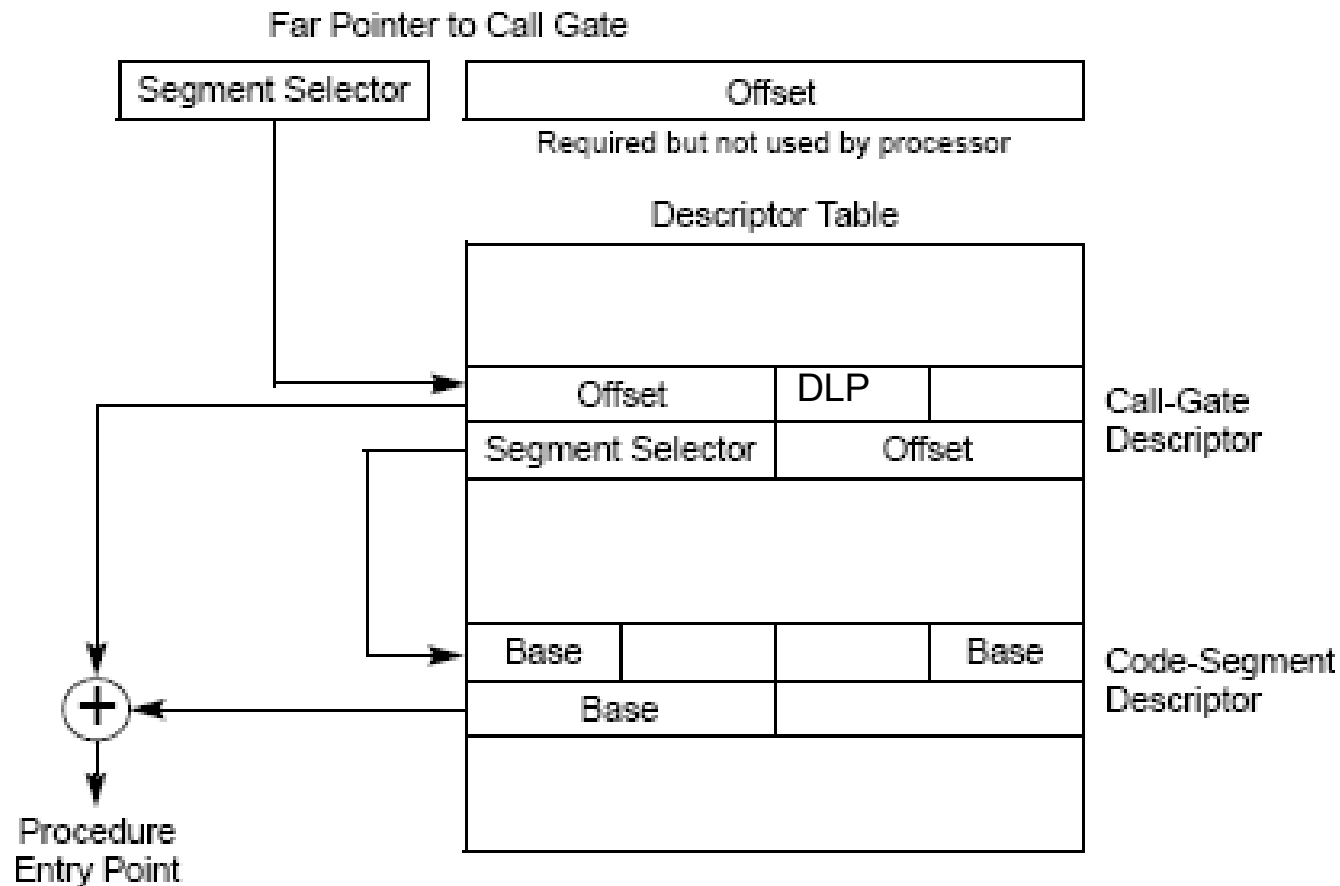


Direct Control Transfers

- For non-conforming code (the common case)
 - $CPL \leq DPL \ \&\& \ RPL \leq DPL$
 - Can only directly jump to code at same privilege level or less privileged

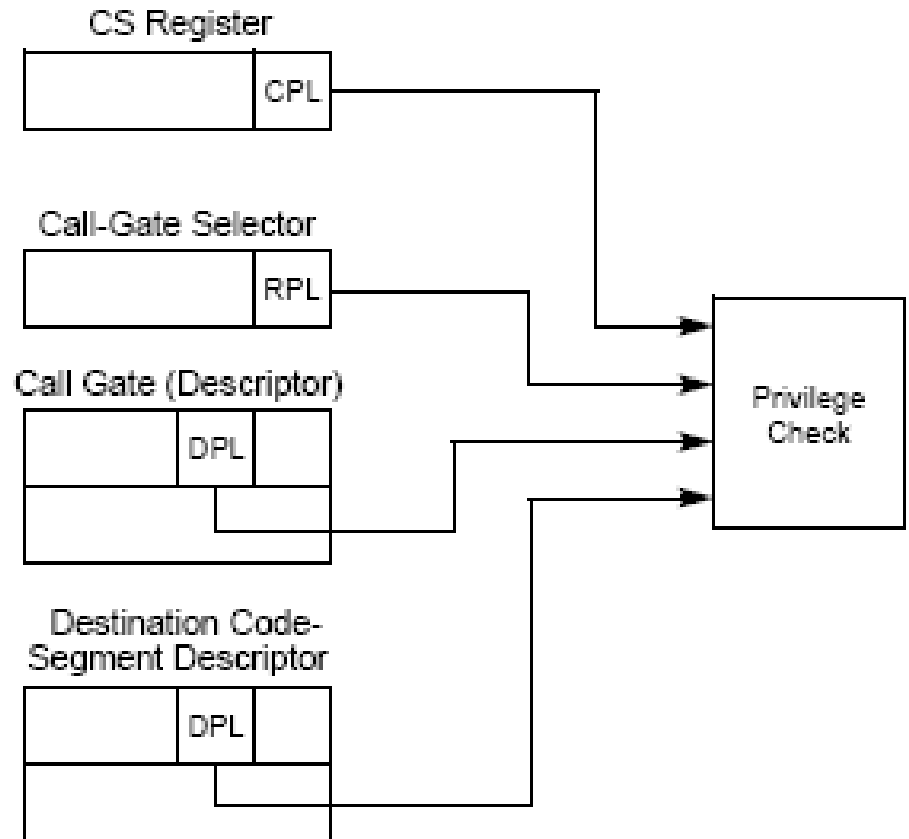


Calling Through Gates

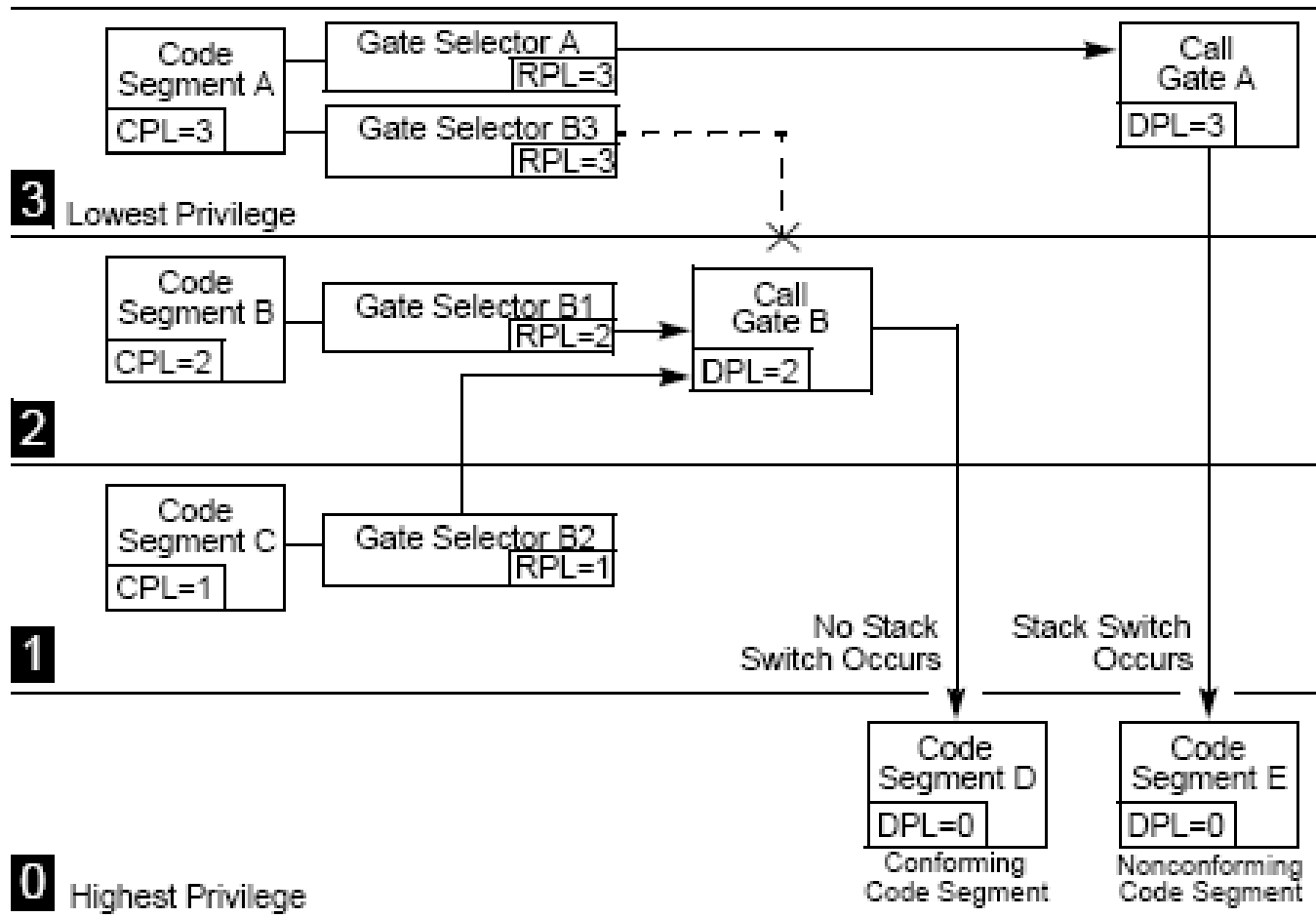


Call Gate Access Rules

- For Call
 - $CPL \leq CG\ DPL$
 - $RPL \leq CG\ DPL$
 - $Dst\ CS\ DPL \leq CPL$
- Same for JMP but
 - $Dst\ CS\ DPL == CPL$



Call Gate Examples



Stack Switching

- Automatically performed when calling more privileged code
 - Prevents less privileged code from passing in short stack and crashing more privileged code
 - Each task has a stack defined for each privilege level

Hardware Rings

- Only most basic features generally used
 - 2 rings
 - Installed base
- Time to adoption
 - Must wait for widespread system code, e.g. Windows NT

Limiting Memory Access Type

- The Pentium architecture supports making pages read/only versus read/write
- A recent development is the Execute Disable Bit (XD-bit)
 - Added in 2001 but only available in systems recently
 - Supported by Windows XP SP2
- Similar functionality in AMD Altheon 64
 - Called No Execute bit (NX-bit)
 - Actually in machines on the market sooner than Intel

Windows Support

- Enabled in Windows XP SP2 as Data Execution Prevention (DEP)
 - Software version if no hardware support
- Check to see if you have the bit
 - Control Panel -> System -> Advanced -> DEP tab

Delay to widespread deployment

- First hardware in 2001
- Wait for OS support
- Wait for vendors willing to sell
- Generally available in 2005

Capabilities HW

- Intel iAPX 432 (mid '70s)
 - Tried to put even more security enforcement in hardware
 - Capabilities and object-oriented
 - Implementation too complex and compiler technology not sufficiently smart
 - Too slow and died on the vine
 - http://en.wikipedia.org/wiki/Intel_iAPX_432
- IBM System/38
 - From about the same time period
 - Also had hardware capabilities support

Consider encrypted files

- Each file or directory may be encrypted with a unique key
 - How are the encryption keys stored?
 - Protected by the file system access control?
 - What if system root is compromised?
 - Encrypted by a master key?
 - How is the master key stored?
 - Protected by pass phrase?
 - » Then human must be present
 - » If multiple users use system, all must know pass phrase
 - Hide it in a good place and hope nobody finds it?

Another solution

- Secure separate storage for root keys
 - Smart card
 - Secure co-processor
- Keys never leave security processor
 - Protocol to send encrypted blob to security processor and return decrypted data
- Tamper proof
 - Data is destroyed when tampering is detected
 - Prevents sophisticated adversary from pulling secrets from data

Crypto/Smart Cards

- Fortezza Card
 - Associated with proposed key escrow scheme
 - Used with secure phones
 - Vendor link
http://www.spyrus.com/products/fortezza_cryptocard.asp
- Smart Cards
 - Much lower cost point
 - So much lower storage and computation ability
 - Primarily used for authentication and/or tracking small amounts of information
 - E.g., frequent buyer programs or phone cards
 - <http://www.gemplus.com/smart/cards/basics/download/smartcard>
- Can use smart or crypto cards to link into many authentication schemes
 - Windows GINA, Linux PAM, Radius

Secure Co-Processors

- Co-located on a server or laptop
 - Prevents secure root information from being accessed by malicious programs on the general CPU
- IBM sells security processors
 - <http://www-03.ibm.com/security/cryptocards/>
 - <http://www.research.ibm.com/journal/sj/403/sr>

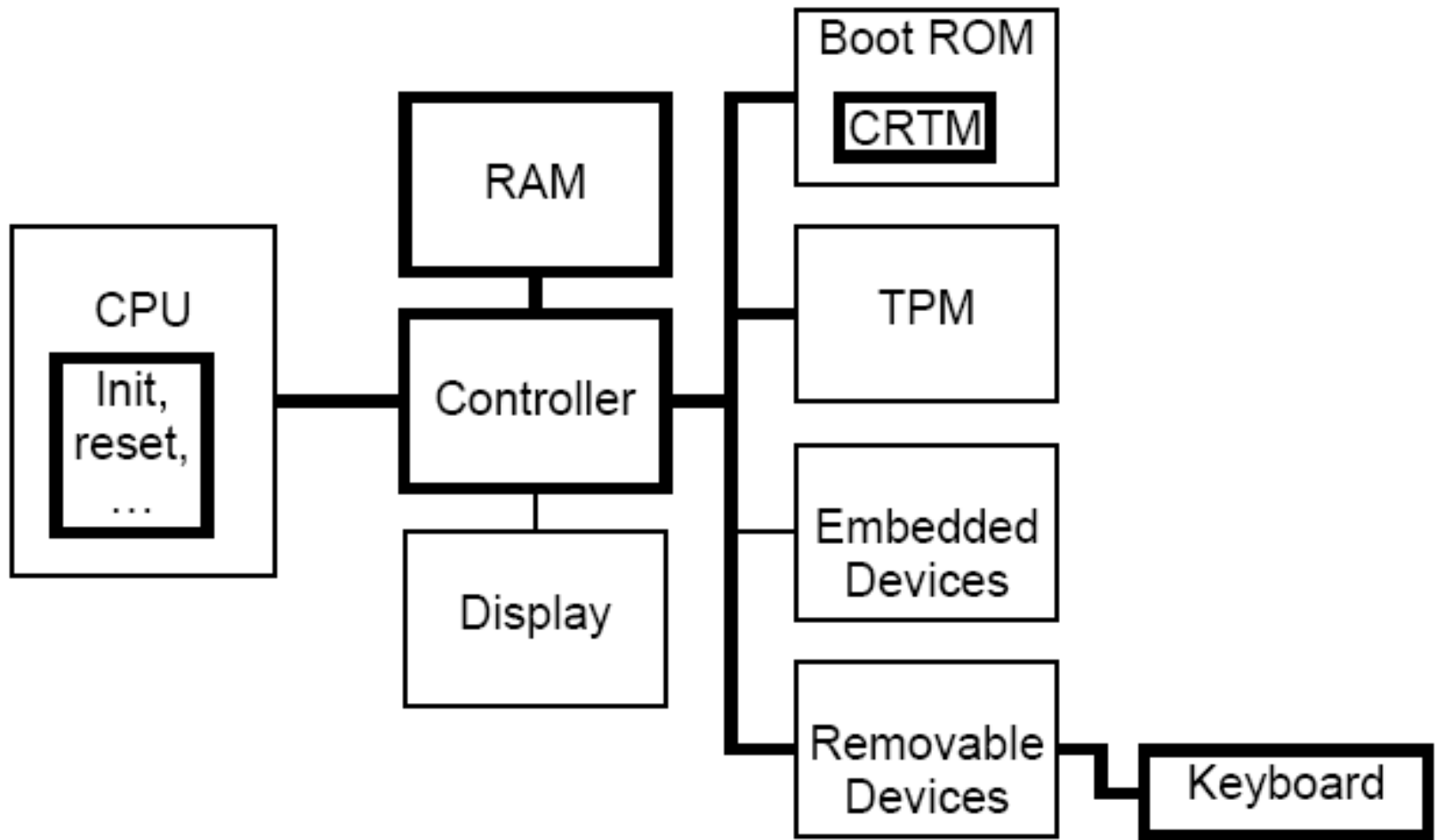
Trusted Computing Group

- Consortium developing standards for computer architectures using secure co-processors
 - Called the Trusted Platform Module (TPM)
 - <http://trustedcomputinggroup.org>
- Numerous computers (particularly laptops) already ship with TPM's
 - <http://www.tonymcfadden.net/tpmvendors.html>
 - Many vendors targeting specific enterprises like Health Care that are particularly concerned with privacy (due to HIPPA)
 - Supported by Vista/Longhorn

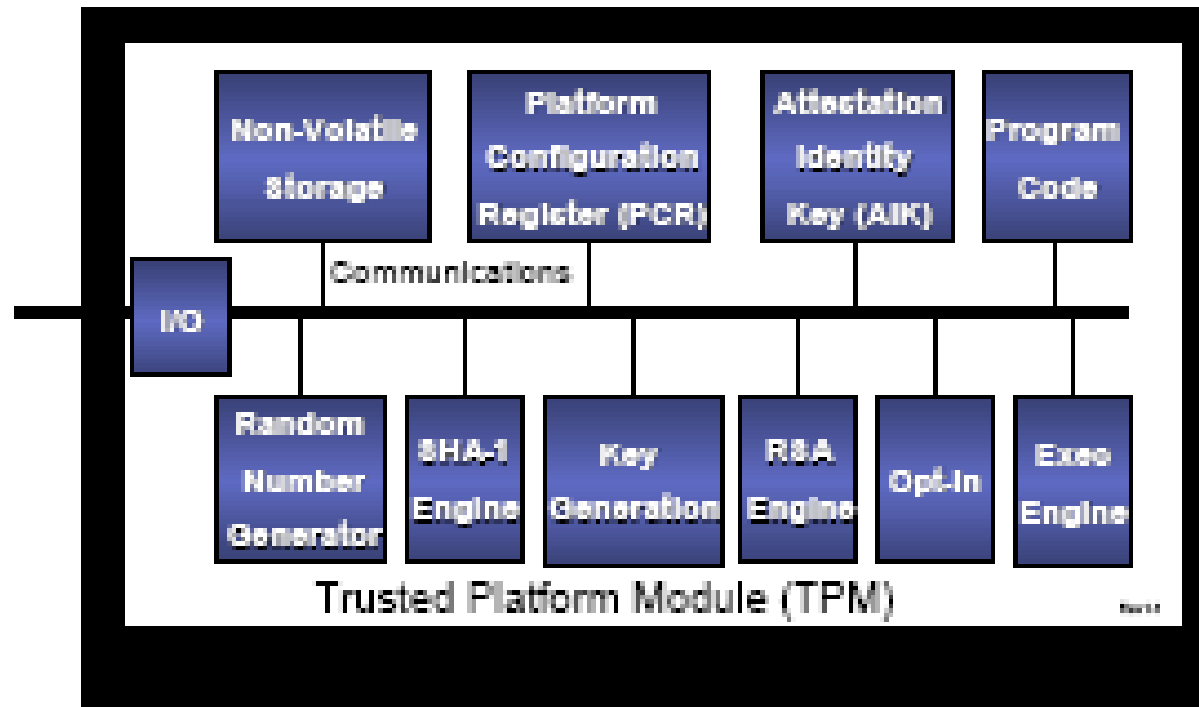
Major Functions of TPM

- Attestation
 - Proof of accuracy of information
 - Attestation by TPM
 - Sign internal TPM information by TPM
 - Attestation of Platform
 - Proof of platform integrity
 - Authentication of Platform
 - Signature by non-migratable key
- Protected Communication

TPM Architecture Overview



TPM Layout



TPM Features

- Unique, unmigratable keys
 - Unique Endorsement Key (EK) set at manufacture
 - Can generate additional Attestation Identity Keys (AIK)
 - Binds key use to particular piece of hardware
 - Potential Privacy concerns
- Protected Capabilities
 - Shielded storage
 - Platform Configuration Registers (PCR)

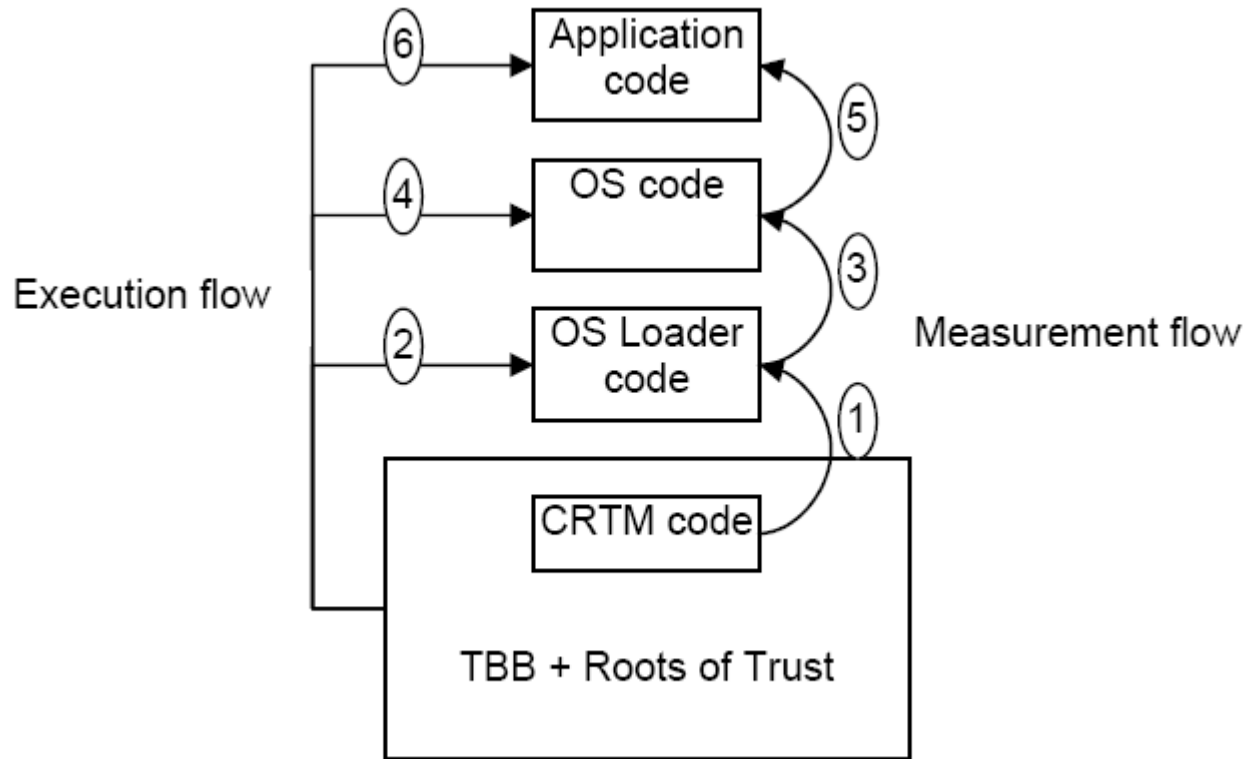
TPM Protected Message Exchanges

- Binding – Encrypting using public key
 - If using non-migratable key value is bound to TPM
- Signing – Encrypt with private key
 - Some keys are indicated as signing only keys
- Sealing – Binding a message with set of platform metrics (expressed in PCRs)
 - So can only unseal values when the platform metrics match
- Sealed-signing – Have a signature also be contingent on PCR values

Roots of Trust

- Root of Trust for Measurement (RTM)
 - Capable of making inherently reliable integrity measurements
- Root of Trust for Storage (RTS)
 - Capable of making accurate summary of values of integrity digests and sequences of digests
- Root of Trust for Reporting (RTR)
 - Capable of reliably reporting data held by RTS

Transitive Trust



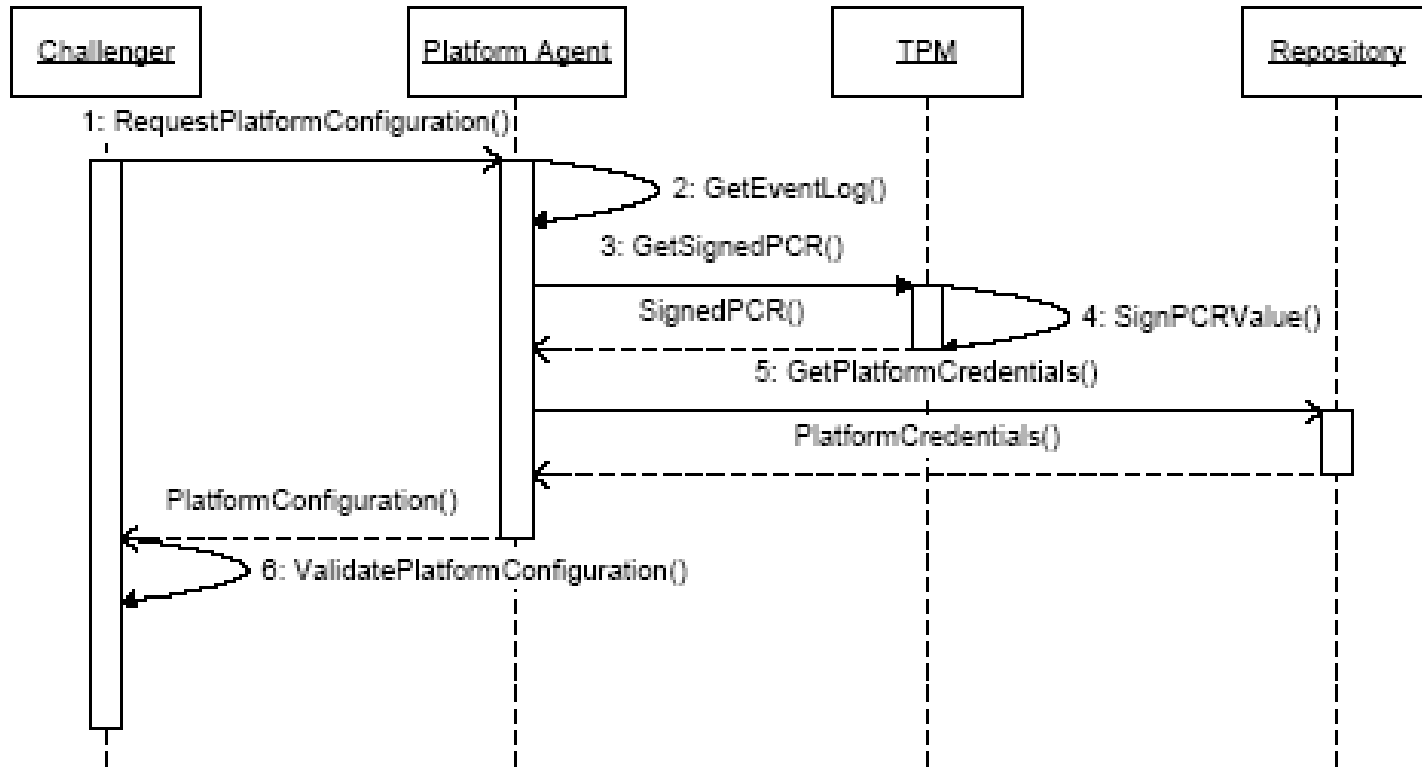
Integrity Measurements

- Measurement values
 - Representation of data or program code
 - Can be stored anywhere
 - Related measurement values stored in Stored Measurement Log (SML)
- Measurement digests
 - Hash of the measurement values
 - Stored in the TPM
 - Fixed number of Platform Configuration Registers (PCRs)

Integrity Reporting

- Two purposes
 - Expose shielded locations for storage of integrity measurements
 - Means to manipulate PCR's
 - Attest to the authenticity of stored values based on trusted platform identities
 - Integrity reports signed by Attestation Identity Keys (AIK)
 - TPM shipped with unique Endorsement Key (EK) with is used to generate AIK's

Example Reporting Protocol



Usage Scenarios

- Store root secrets in secure co-processor
- In an enterprise, IT group is responsible for machine admin
 - They set up the TPM
 - End user cannot muck with TPM even if they are root on the machine
- Ensure platform is in particular configuration
 - Verify the digest values of SML of configurations of interest

Digital Rights Management (DRM)

- One scenario concerns protecting data from the user for the vendor
 - Alice buys a song from Recording Company
 - License agreement says that Alice buys song for personal use
 - Trivial for Alice to share song with 10,000 of her closest friends
 - Hard for Recording Company to track
 - Want to protect their assets
 - Can use specialized players, as in Sony's recent rootkit problems

Using TPM for DRM

- Alice registers with Record Company for the ability to play their songs
 - Record Company sends her certificate to store on in her TPM and a player to install
 - On boot, TPM verifies that player has not been changed
- Alice buys a song from Record Company
 - Song is sealed to the “correct” player configuration on Alice’s computer
- To play song
 - Player passes sealed blob to TPM
 - TPM detects that it is invoked from legal player
 - TPM decrypts if sealed PCR values match
 - Player plays it
 - No unauthorized program can decrypt song

Limitations of TPM for DRM

- Even if no other program can spoof player in TPM interactions
 - Root user can use program debugger to access decrypted program in memory
 - Then may unencrypted copy for use outside player
- Could use more stringent OS mechanisms
 - One paper describes using SE Linux type enforcement to compartmentalize
 - <http://www.cs.dartmouth.edu/~sws/papers/acsac04.pdf>
 - But if I own system, I can bypass most any OS mechanism

Key Points

- A little bit of hardware support can enable a great many security options
 - Hardware protection to protect from malicious software
- Putting too much support in hardware not practical
 - Turn around time in HW is long
 - Specialized hardware lags high volume hardware in performance
 - HW is expensive
 - Often cost point is very low in target application