# Terra: A Virtual Machine-Based Platform for Trusted Computing by Garfinkel et al.

(Some slides taken from Jason Franklin's 712 lecture, Fall 2006)

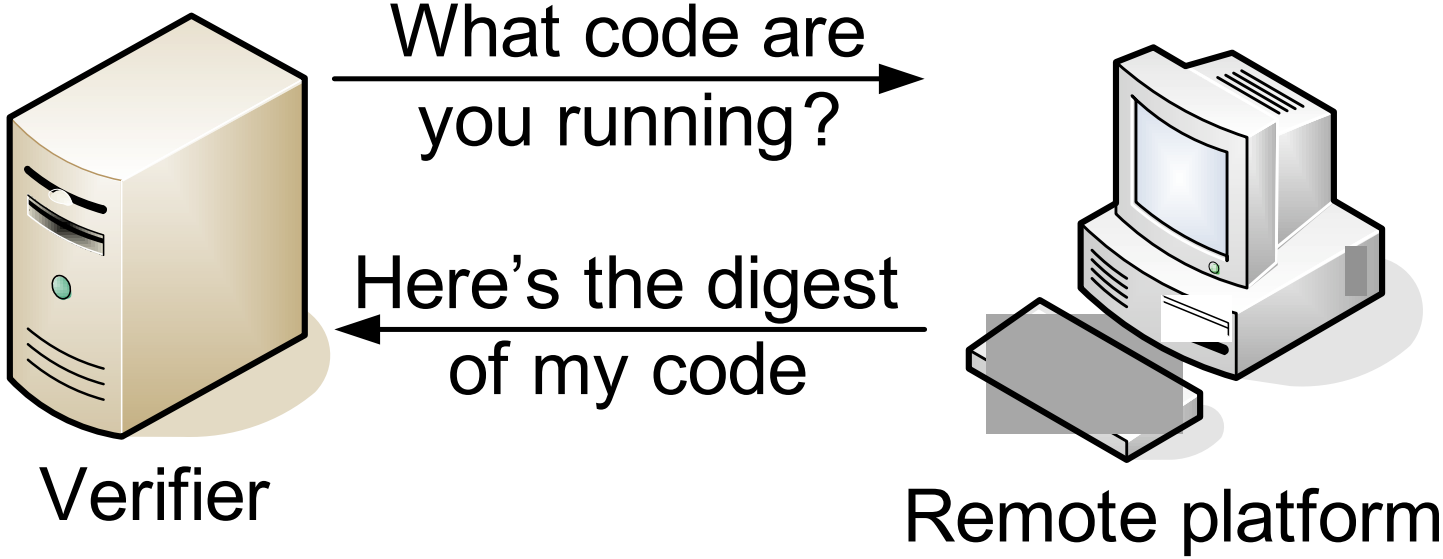
# Trusted Computing Hardware

- What can you do if you have “trusted” hardware?
  - Immutable, with deep control over the resulting behavior of the machine
  - Can use to guarantee certain behaviors and properties of the machine
- How can you do it?
  - Practically?
  - With legacy O/S and applications?

# Primitives of Trusted Computing

- Attestation
  - "I'm running what you think I'm running"
- Secure boot
  - "I can only run what is OK"
  - Less popular approach -- privacy/usability/monopoly concerns
- Note lots of policy/social/legal ?s
  - Can be useful tool
    - e.g., dga's distributed testbed
    - Prevent bots from hijacking bank session
  - Can be used for evil (DRM, lock-in, etc.)
    - "Sorry, can only play this CD under windows!"

# Trusting Software




**Code attestation enables us to establish trust in a remote platform**

# Attestation Today

- TCG (formerly known as TCPA) goal is to add secure platform primitives to each client (now the focus is also on servers, cell phones, PDAs, etc.)
- Industry consortium by AMD, IBM, Intel, HP, Microsoft, …
- These secure platform primitives include
  - Platform integrity measurements
  - Measurement attestation
  - Protected storage
  - Sealed storage
- These can be used to provide trusted boot
- Provides attestation, which enables an external verifier to check integrity of software running on host
  - Goal: ensure absence of malware; detect spyware, viruses, worms …

# Hardware Attestation Functions

- Starts from the bottom
  - Hash the firmware, bootstrap loader, OS, etc.
    - TPM can sign these with secret key (hardware protected)
- Trusted boot / remote attestation
  - Attest to value of integrity measurements to remote party
- Protected  storage
  - Provide “secure” data storage (think smartcard)
  - Secure storage for private key $K^{-1}_{TPM}$
  - Manufacturer certificate, for example $\{K_{TPM}\}_{K^{-1}_{IBM}}$
- Sealed storage
  - Unlock state under a particular integrity measurement

# Terra Argument

- Need to deploy secure systems with commodity computing systems
- Commodity systems (hardware and software) impose “fundamental limitations” on security
  - Poor isolation between applications (processes)
  - Weak mechanisms to authentication applications to peers (distributed computing)
  - No trusted paths between users and trusted computing base (TCB)

# Two Worlds

## Open Box

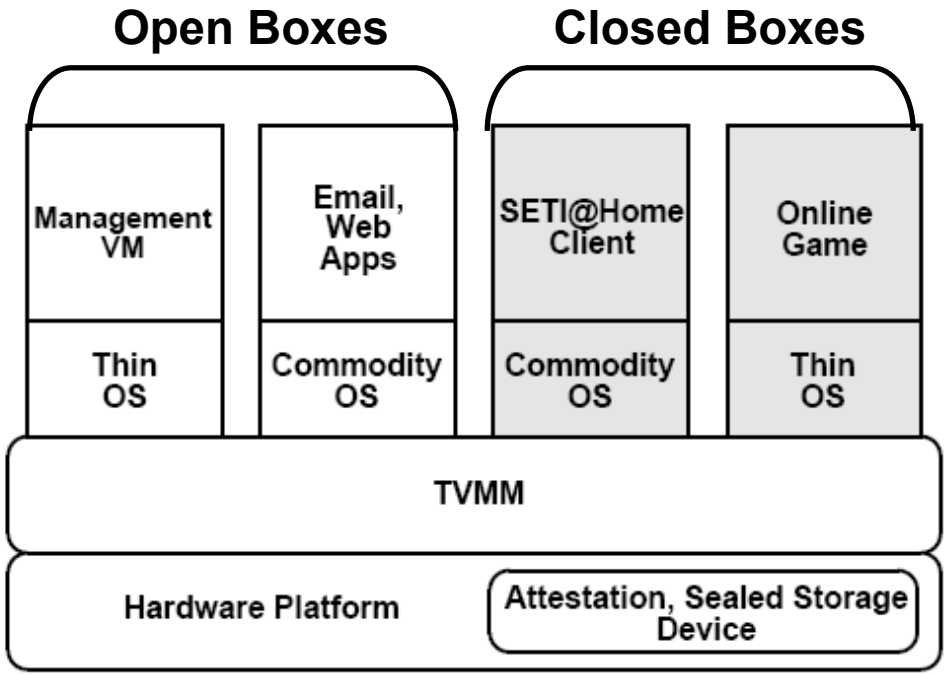

## Closed Box

# Two Worlds

- Open Box
  - General-purpose
  - Extensible
  - Runs huge body of existing code
  - Economies of scale
  - Rich functionality
  - Few security guarantees
- Closed Box
  - Hardware tamper-resistance
  - Embedded cryptographic keys
  - Higher assurance than open box

# Uniting Two Worlds with a TVMM

- Trusted virtual machine monitor (TVMM) “partitions a single tamper-resistant, general-purpose platform into multiple isolated virtual machines”



# Trusted Computing and Closed-box VMs

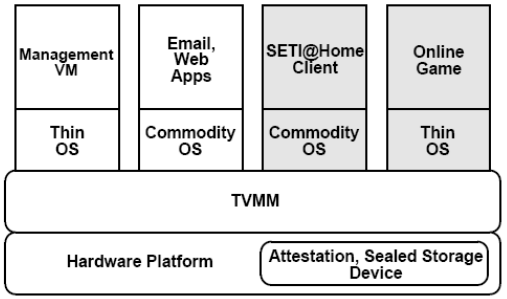
- Terra's Goal: make closed-box VMs equivalent to dedicated hardware and software of closed-box platforms
  - While still allowing open-box VMs
  - And do it all on general purpose hardware
- TVMM protects privacy and integrity of closed-box VM's contents
  - Applications inside closed-box VM can redefine software stack to suit application
- TVMM can authenticate the contents of a closed-box VM (attestation)

# Assumptions

- Assume VMM is free of software vulnerabilities (i.e., trusted)
- Hardware support required
  - Hardware attestation
    - Like the Trusted Computing Group's (TCG's) Trusted Platform Module (TPM)
  - Sealed Storage
    - Decryption (unseal) of data (storage) only possible in same state as during encryption (sealing)
  - Hardware support for virtualization (optional)
    - Intel VT or AMD Pacifica
  - Hardware support for secure I/O (trusted path)
  - Secure counter (optional)
    - Increment only counter
  - Device isolation
    - Countering "attacks from below" by DMA
  - Real-time support
  - Tamper-resistant hardware (not disk but CPU, memory, etc.)

# TVMM Revisited

- TVMM provides standard VMM properties:
  - Isolation
    - Each VM runs in own hardware protection domain
  - Extensibility
    - VM is a dedicated platform
  - Efficiency
    - Negligible virtualization overhead
  - Compatibility
    - Zero modifications required to run commodity OSs
  - Security
    - Small code size, narrow/stable/well-defined interface (like drivers?)



# TVMM Revisited

- TVMM only capabilities:
  - Root secure
    - Security against tampering by root user
  - Attestation
    - Hey peer!  What code are you running?
  - Trusted path (unimplemented)
    - Direct to the TCB communication channel with guarantees of data authenticity, secrecy, and integrity

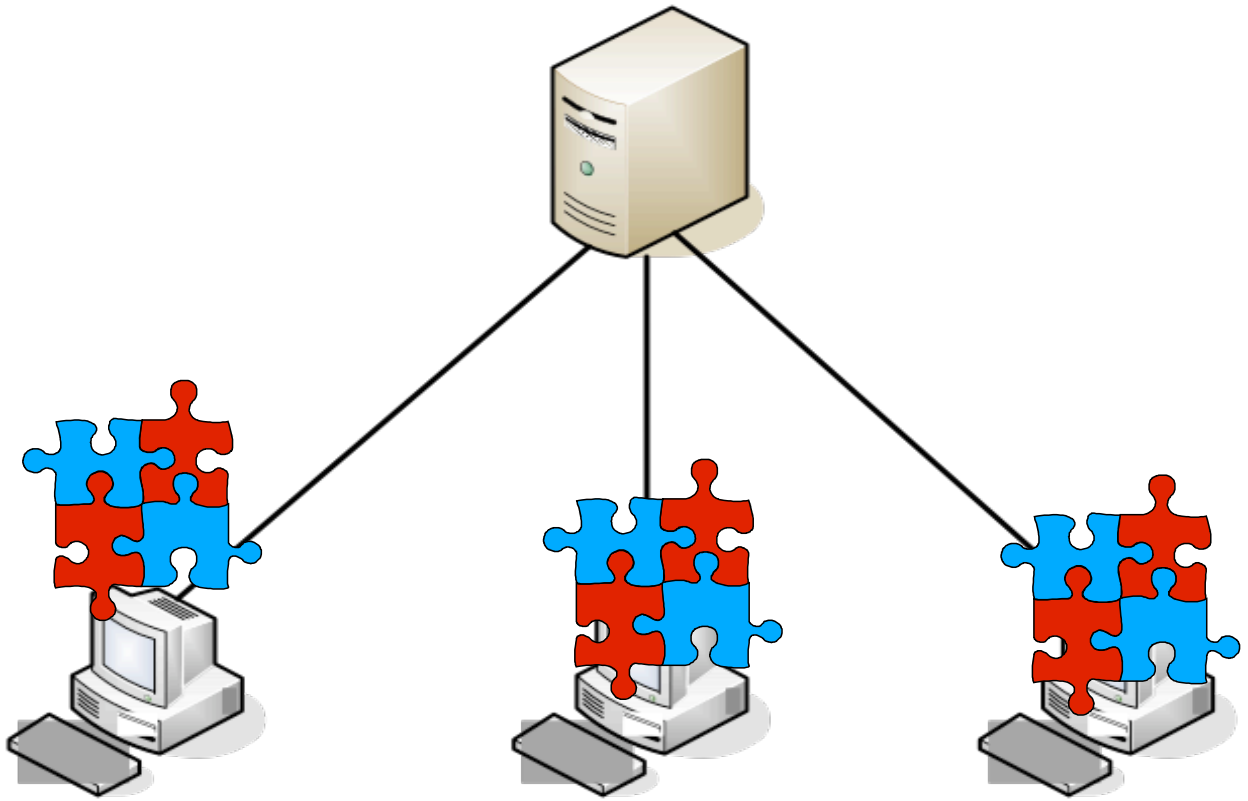
# Local Security Model

- Two components: *TVMM* and *management VM*
  - TVMM runs at the highest privilege level and is secure against tampering by administrator (root secure)
    - TVMM dictates policy for attestation (all other policy decisions made by management VM)
    - TVMM cannot guarantee availability
  - Management VM
    - Formulates all platform access control and resource management policies
      - Grants access to peripherals, issues CPU and memory limits, etc.
    - Management VM run by platform owner
      - Security guarantees of the TVMM cannot depend on management VM

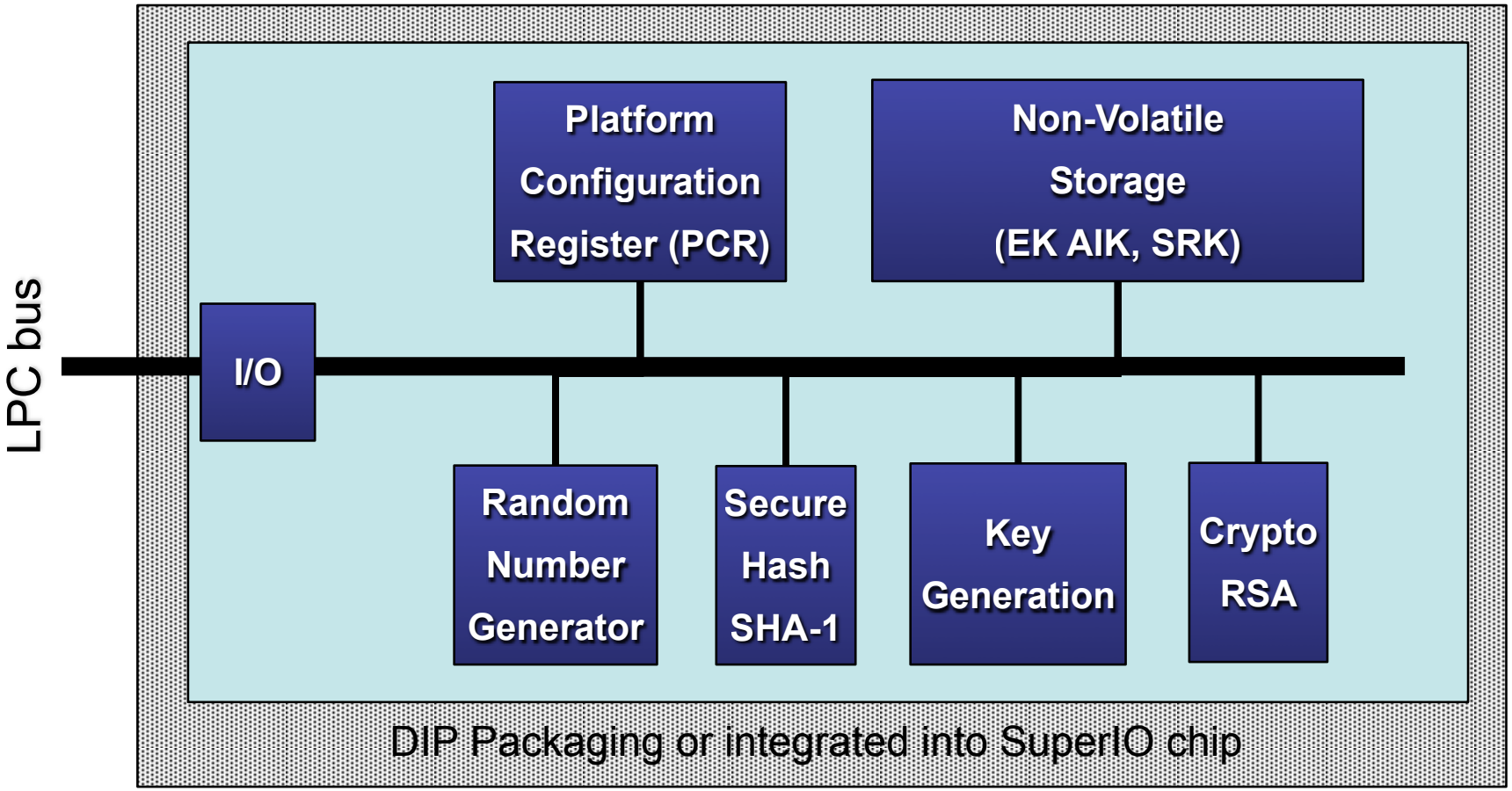
# Application Assurance

- Commodity OS kernels
  - Poor assurance, easily compromised
  - Difficult to reason about isolation
  - Platform security equivalent to security of most vulnerable component
- Terra provides:
  - Strong isolation between VMs
  - Ability to run application-specific OS
  - Attestation to ensure applications only interact with trusted peers
- Assurance of Terra is equivalent to assurance of the OS (TVMM)

# Distributed Computation



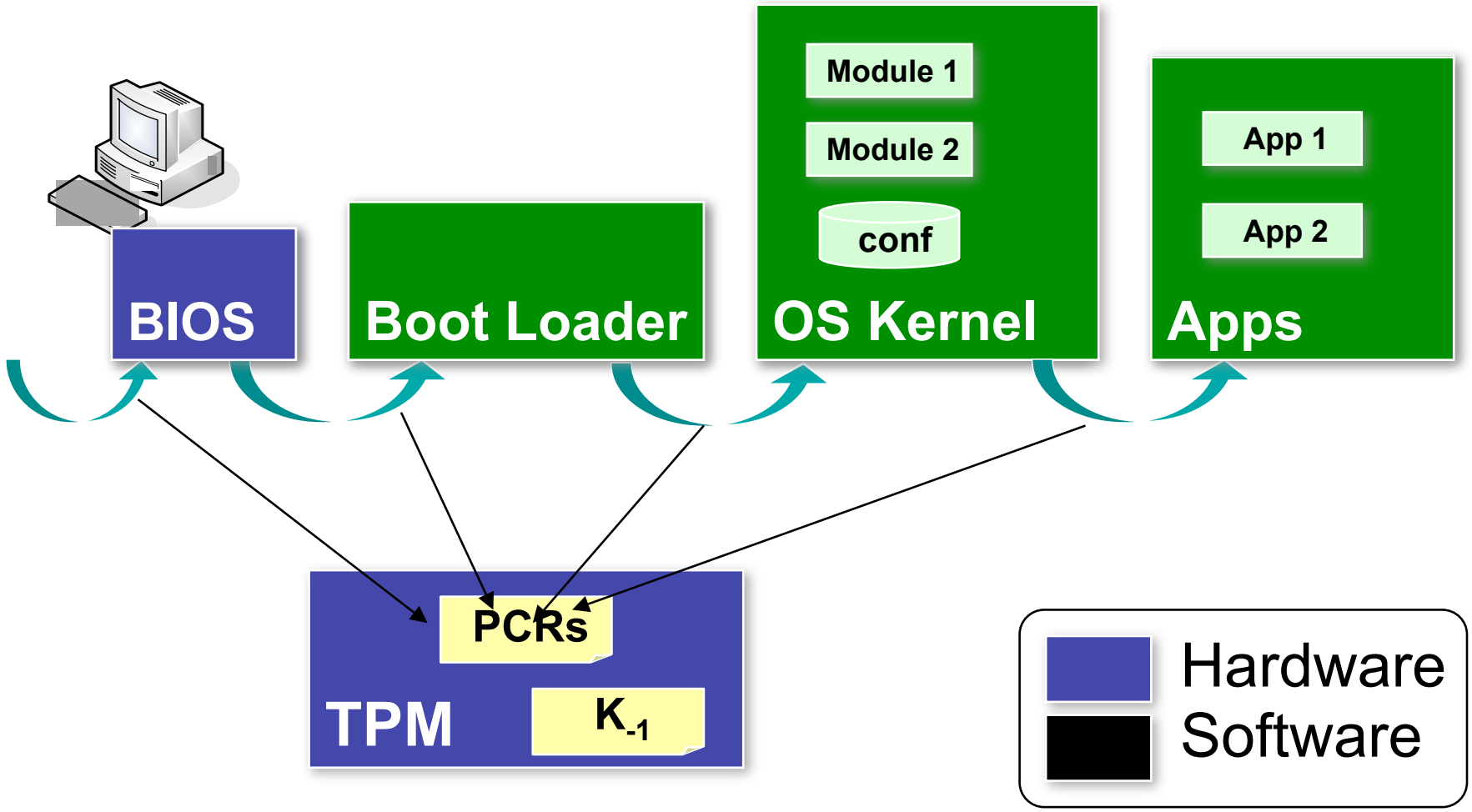
# TCG Trusted Platform Module (TPM)



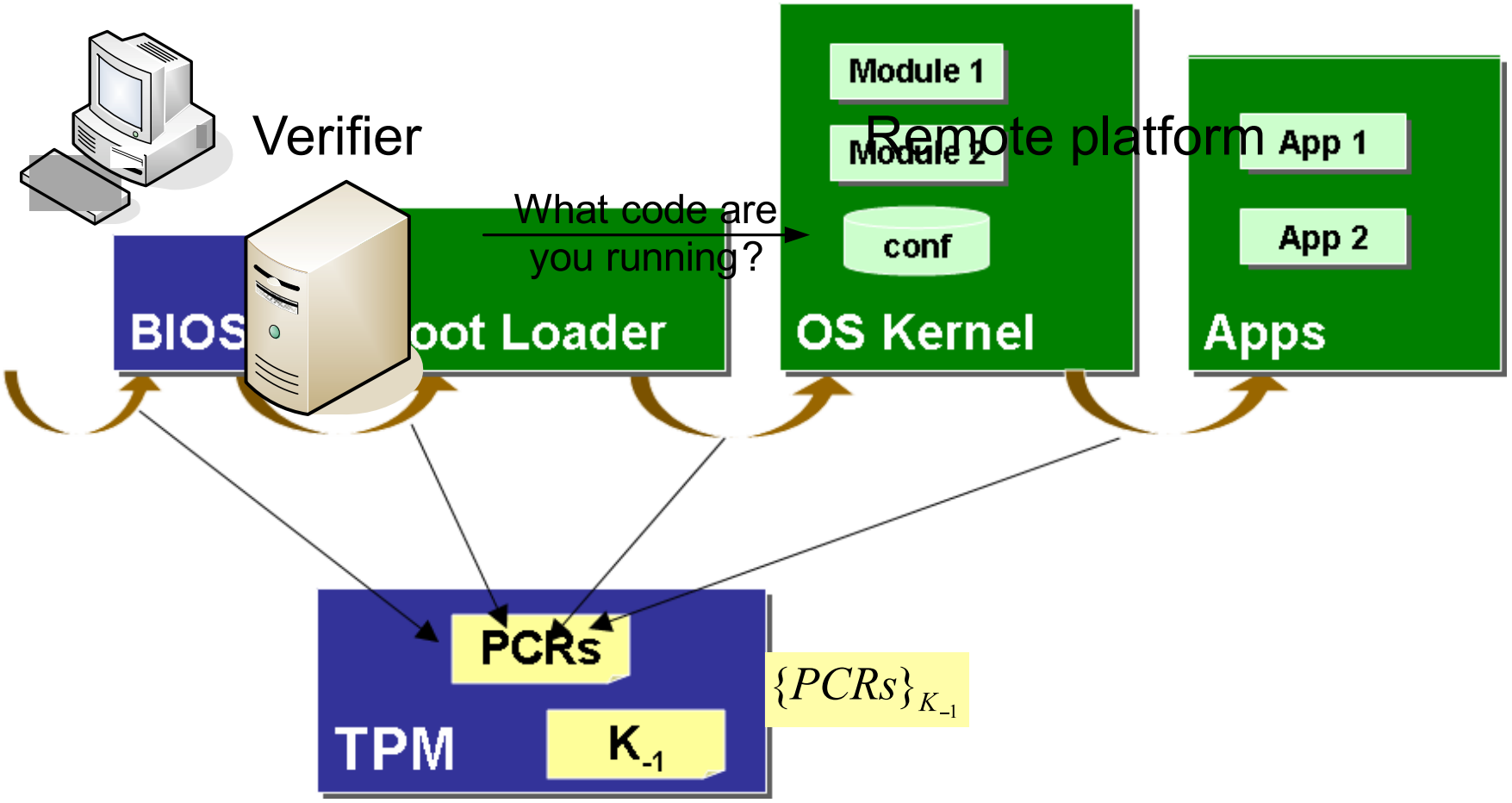
# Basic TPM Functionality

- TPM contains 16 program configuration registers (PCRs) to store integrity measurements
- Operations on PCRs
  - TPM_Extend(N, S): $PCR_N = SHA\text{-}1(PCR_N \mid S)$
  - TPM_Read(N): Return contents of $PCR_N$
- TPM contains private key to sign attestations and manufacturer certificate
  - Tamper resistant storage for private key $K^{-1}_{TPM}$
  - Manufacturer certificate, for example $\{K_{TPM}\}_{K^{-1}_{IBM}}$

# Ahead-of-Time (offline) Attestation



# Ahead-of-Time (offline) Attestation



# Application – Trusted Quake

- Quake – multi-player online game vulnerable to client cheating
- Terra provides:
  - Secure communication
  - Client integrity
  - Server integrity
  - Isolation
- Terra can't prevent:
  - Bugs and undesirable features
  - DoS attacks
  - Covert channels

# Discussion

- Limited TVMM implementation
  - Do not emulate underlying TCPA hardware (no TPM)
  - No trusted path (lack of hw)
  - Bulky TVMM (VMware GSX Server)
  - No high assurance guarantees (Debian/VMware)
- Some experiences implementing trusted quake and trusted access points
- Tons of discussion and material, much of it based on yet unreleased or alpha technologies
- Lots of we're sorry but we…
  - Don't have special hardware
  - Didn't have source code
  - Didn't implement this or that
- Great deal of foresight into future technologies
- Trusted computing technologies are a available today
  - Terra could be realized almost as predicted

# Open Research ?s

- How to build secure systems using TPM?
  - Attestation is potentially ugly!
    - Must attest/trust every version of windows with every combination of patches?!
    - Or do you force WinXP sp2 with IE7 and patches 1, 5, 9, 10?
  - Alternate approch: Gun Sirer's "Nexus" OS
    - Labels that attest to *properties*
      - e.g., "Media player will not copy; will allow only N plays of video"
      - Media can be played by any player that makes those guarantees (some cert. auth. has to sign for them...)

- This is ongoing research
  - Definitely don't know the answers yet!
- What does TPM let us do differently?
  - Where would you draw security bounds differently?
  - How much trust should you export to "trusted" client?
    - Still vulnerable to...
      - maybe: Rogue DMA hardware? RDMA network card??
      - bus analyzer? CPU interposer?
      - government/org. crime with STEM?

# Examples to consider

- Fairness / congestion control in networks (most people don't care enough to break;  rewards small)
- DDoS prevention (hardware owner probably doesn't want computer being used to launch DDoS)
- Virus scanning (benefits owner of computer)
- Cheating prevention in games (stakes aren't that high...)
- Secure RDMA-like access to NFS with access control performed by trusted local proxy (earlier papers)
- Updating bank balance / securely handling e-cash
- Voting?
- Where to draw the line between {on trusted server, on trusted client, on untrusted client}?  What changes?

# Building Secure Distributed Systems

- Challenge: Build trustworthy service based on distributed set of potentially untrusted hosts
- Approaches
  - Software security community has proposed mechanisms to harden software to prevent exploits [Prevention]
  - Intrusion detection community has proposed mechanisms for detecting specific attacks or anomalies [Detection and Recovery]
  - Distributed systems community has designed protocols to provide property if up to 1/3 of hosts are compromised (Byzantine hosts) [Resilience]
- Attestation
  - Provide guarantee that correct code is executing on remote host
  - Vendors embed trusted HW in devices providing attestation
  - Exciting new directions for building secure systems