Active Systems Management

William Arbaugh

Virgil Gligor

University of Maryland

College Park

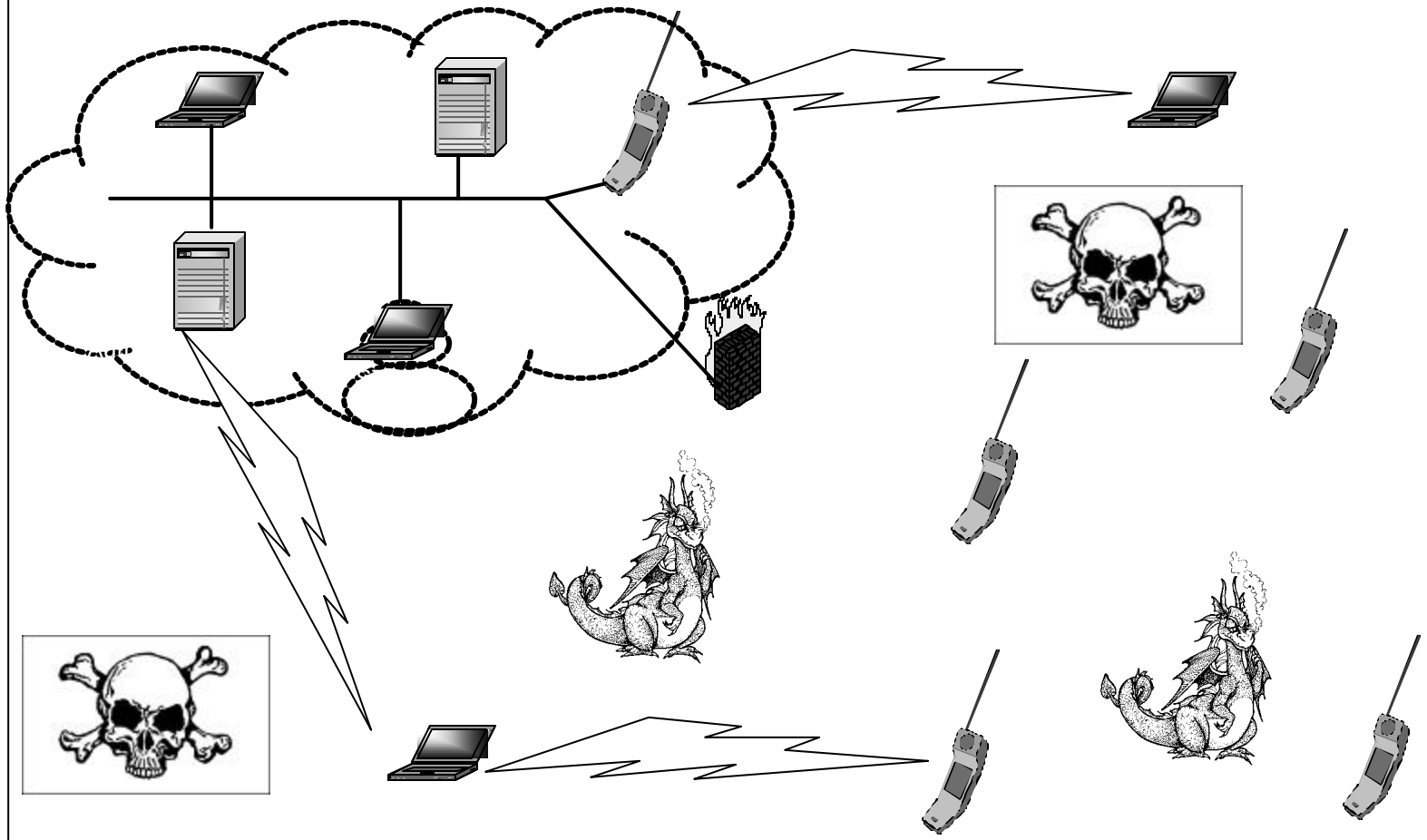
Future Environment

- Devices may require multiple management sources
  - A handset may need to receive updates from the manufacturer,
  - The developers of installed applications, and
  - Receive user and/or organizational data

Future Environment

- Management will become significantly more difficult
  - Separation of management instructions is a MUST,
  - Many organizations will want to be “in the loop” on all management instructions,
  - Devices are “always on”

The Future



What to do?

- Reevaluate the role of the firewall
- Obviously- need better management (easier said than done)

Reevaluation of the firewall

- In the future ubiquitous “always on” world- every device MUST be able to protect itself.
- Further- the mobility of many of the devices will make centralized management difficult if not impossible.

How do we improve management?

- Unfortunately, too little research has been done on systems management
- Our approach: Active Systems Management
  - Formalize the problem
    - Host state model
    - Active systems management process
  - Build and evaluate experimental systems
    - Independent Audit
    - Enforcement
    - Communication

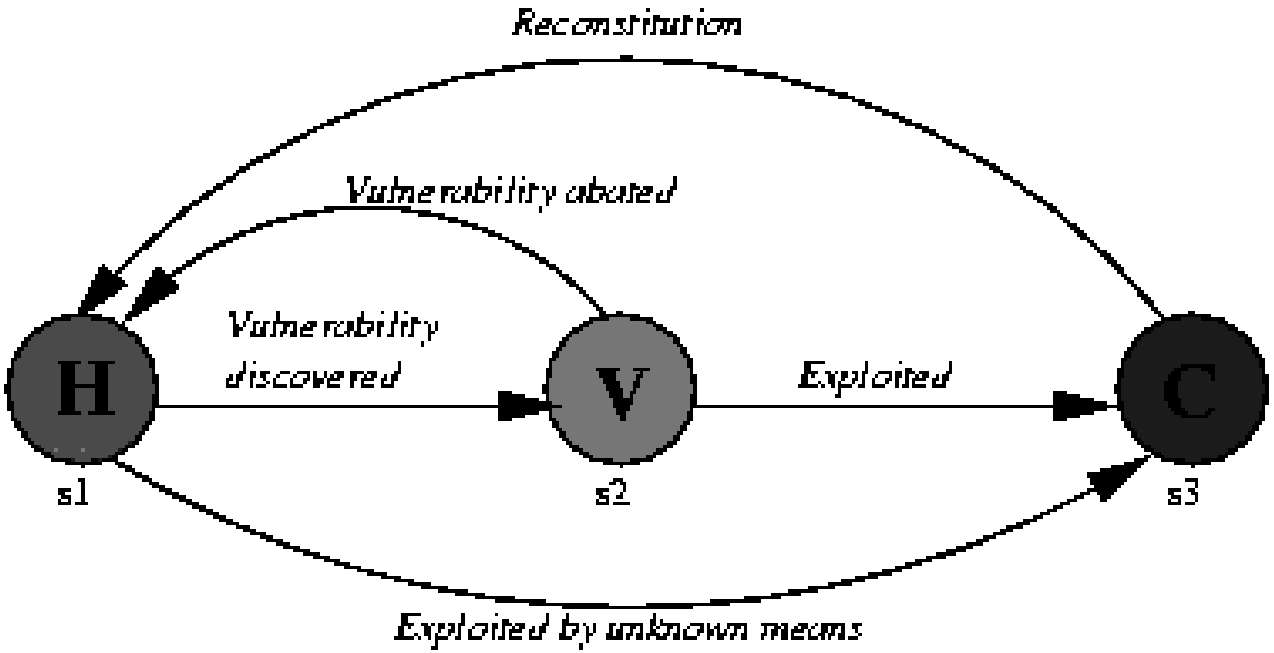
Active Systems Management

- Since most all devices will be highly mobile- configuration and management instructions MUST be mobile as well so that devices can receive instructions in a timely fashion.
- Every device MUST be able to protect and reconstitute itself in an OS independent fashion.
- Investigate historical evidence to gain a broader understanding of the threat.

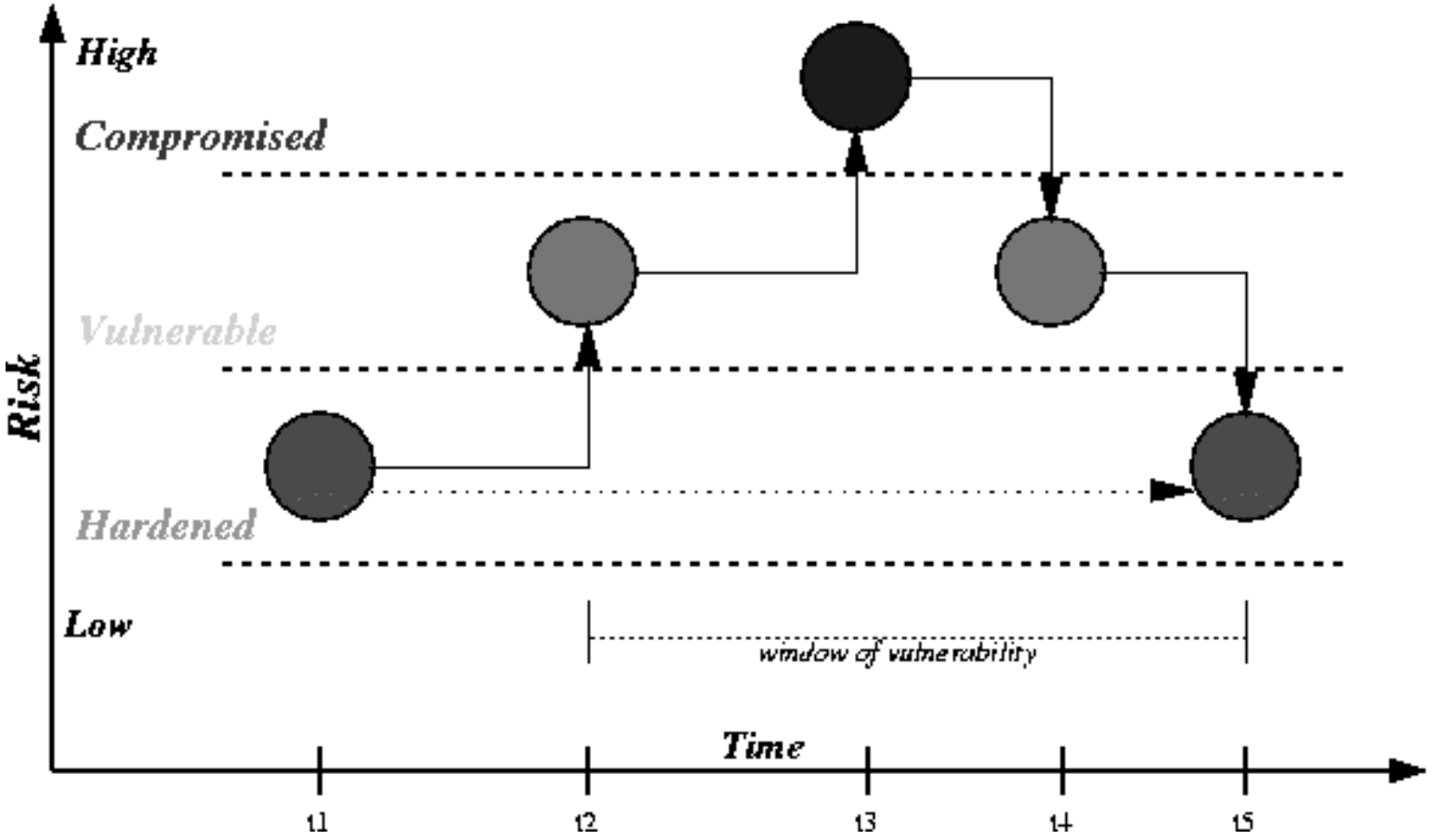
Host States

- Hardened: The host/device is patched and configured against all know vulnerabilities.
- Vulnerable: The host/device is vulnerable to at least one known attack.
- Exploited: An attacker has successfully exploited a vulnerability on the host/device.

Host State Model



Example


High
Compromised
Vulnerable
Hardened
Low
Risk
window of vulnerability
Time
t1
t2
t3
t4
t5

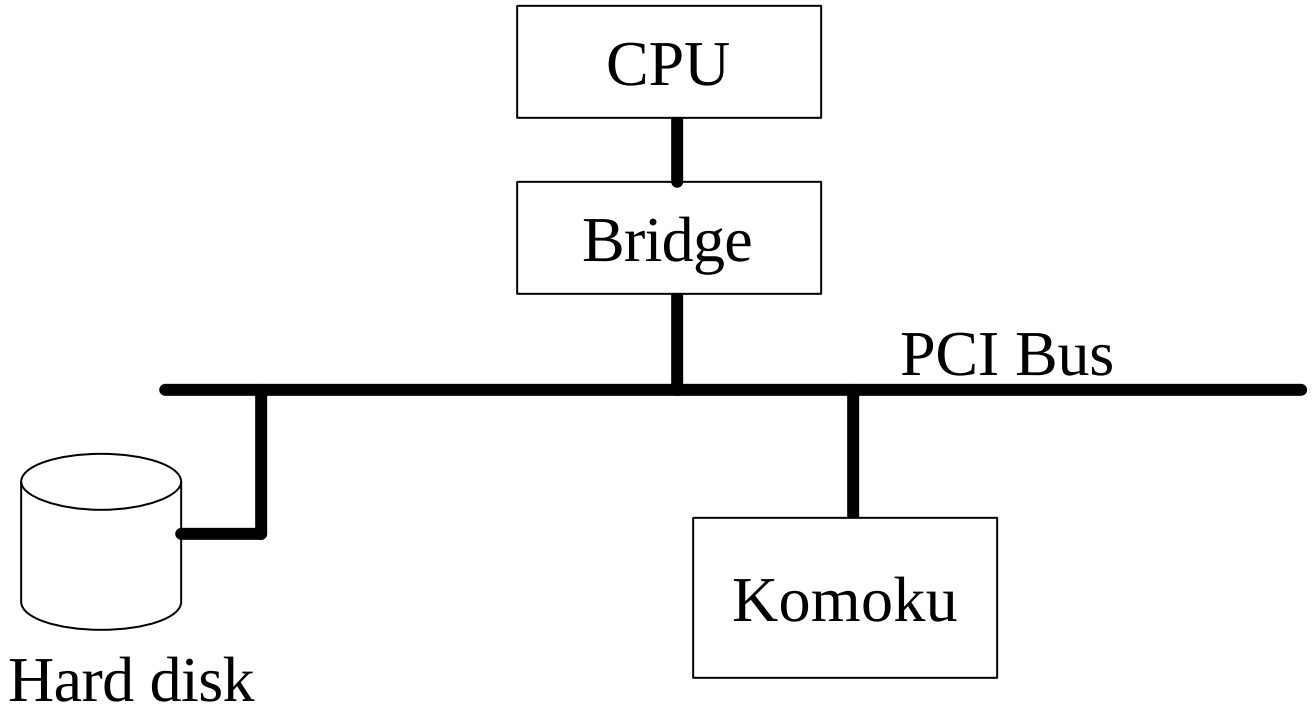
Research Questions

- Given any host, what is the current state of the host? (*Identification*)
- Given that a host is either vulnerable or exploited, what are the minimal steps required to transition the host back to the hardened state, and how do we execute them? (*Reconstitution*)

Komoku: How do we identify the current state?

- Impossible to determine state with only software because attackers modify OS to report false information
- Komoku is an add-in co-processor that serves as an independent auditor that is isolated from the host OS
- Goals were to make Komoku OS independent with absolutely no OS modifications required

First Prototype



Problems with PCI Bus

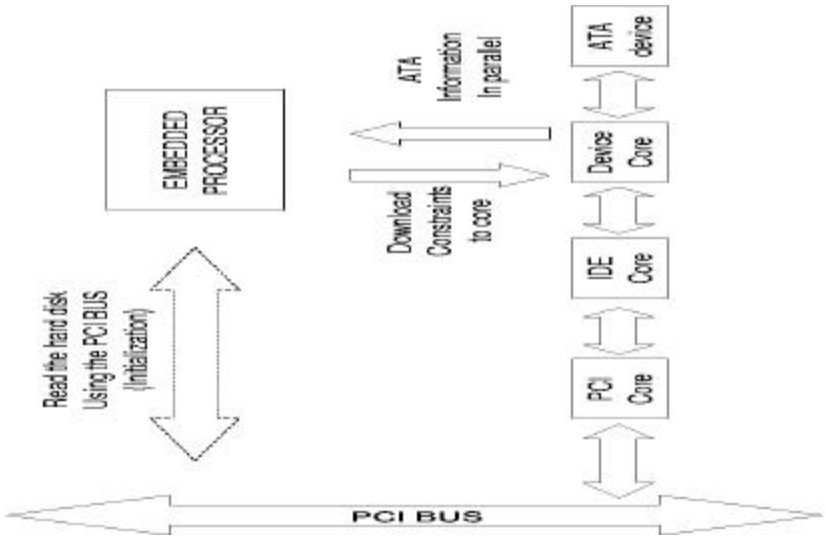
- Unfortunately, many implementations of the PCI bus DO NOT support mutual exclusion
  - Results in race condition when Komoku and OS try to read the disk at the same time
- Solution was to implement a simple MUTEX using PCI registers
  - Requires host OS support, but does not introduce any significant weakness.
  - Required writing a polled IDE driver for Komoku

Results

- Komoku has been tested with both Windows NT and Linux using AIDE (Tripwire like application) to provide integrity protection.
- Throughput to Komoku is 1.4 Mbps when Komoku has access to disk.

The Future of Komoku

- Implement Komoku as an FPGA directly along the IO path
- This permits Komoku to be in smaller devices and serve as a security and management enforcer.



Status

- Second generation prototype currently supports limited IDE functionality.
  - Used open source cores from www.opencores.org
  - Moving to commercial cores this fall since the open source cores have not worked well.

Current and Future Work

- Identified a meta vulnerability class induced by layering with three sub-classes: Session hijacking, TOCTOU, man in the middle.
  - Formalizing with BAN Logic (adding a temporal element)
  - Reducing the sub-classes to layering
  - Proving a general mitigation strategy works for all three sub-classes.