CSC/ECE 774 Advanced Network Security

Topic 7.2 Infrastructure Security for Virtual Cloud Computing

Outline

• Background
• Security threats to virtual cloud computing
• Security of virtual cloud computing infrastructures
  – Security architecture
• One recent result
  – HyperSentry: Stealthy and in-context measurement of hypervisor integrity
• Conclusion

What is Cloud Computing

• Wikipedia
  – Cloud computing is a paradigm of computing in which dynamically scalable and often virtualized resources are provided as a service over the Internet
  – Users need not have knowledge of, expertise in, or control over the technology infrastructure in the "cloud" that supports them
• Virtualization is a key to cloud computing
  – Scalability
  – Ease of use
  – Affordable pricing
Virtual Cloud Computing

- Virtual cloud computing is emerging as a promising solution to IT management
  - Reduction in hardware, operational, and administrative costs

- Examples
  - Amazon’s Elastic Compute Cloud (EC2)
  - IBM Research Compute Cloud (RC2)
  - Microsoft Azure
  - NCSU Virtual Computing Lab (VCL)

Industry Example:
Amazon Elastic Compute Cloud (EC2)

Academic Example:
NC State Virtual Computing Lab (VCL)
Security of Virtual Cloud Computing

• Need for security
  – Customers
    • Their workloads will not be subject to attacks
  – Cloud service providers
    • Their cloud services will work normally
  – Other users
    • Compute clouds will not be used as stepping stones to attack them

Security Threats in Virtual Cloud Computing

- External threats
- Guest-to-guest threats
- Guest-to-cloud threats
- Cloud-to-guest threats

Security of Virtual Cloud Computing

• Virtual cloud computing presents
  – New security threats, and also
  – New opportunities

• Our proposal: An architecture for secure virtual cloud computing
  – Addition of security architecture components
  – Arising research opportunities and challenges
Virtualization-based Security Services (Cont’d)

- Ongoing research
  - General direction
    - Goals
      - System integrity
      - Application data confidentiality
    - Strategy
      - Exploit the isolation and higher privilege of the hypervisor to protect the guest OS
  - Examples
    - Lares, VMWatcher, OverShadow, Storage Capsule, SIM, HookSafe
- Technical problems being addressed
  - Semantic gap
  - Overhead of “world changes”
Problem Not Addressed Yet

• Integrity of dynamic objects
  – E.g., stack, heap, global variables
  – Inherent to overall system integrity

• Challenges
  – Non-trivial to identify and derive the known good states of dynamic objects
  – Many objects are transient; integrity violations could be transient
    • Verification needs to be done continuously
  – Potentially a large number of dynamic objects

VM Image Security Services

Why VM Image Security Services

• Reason #1: Convenient place to provide security services
• Reason #2: It is not just convenience; it is necessary
  – Imagine a dormant VM image put to sleep last year …
  – Imagine a VM image repository with thousands of dormant VM images put to sleep at different times …
Possible VM Image Security Services

- Security management
  - Offline virus scanning
  - Offline patching
- Protection of sensitive data in VM images
  - Access control
  - Sensitive data identification and removal
- Integrity protection
  - Integrity evidence over VM image updates
  - VM image provenance

Hypervisor Integrity Services

Why Hypervisor Integrity Services?

- Many virtualization-based security mechanisms assume the hypervisor is trusted
  - Examples: Lares, SIM, HookSafe, Patagonix, HIMA, ...
- Hypervisors cannot be blindly trusted
  - Example #1
    - Two backdoors in Xen [BlackHat 2008]
  - Example #2
    - VM Ware ESX 3.x: 50 Secunia advisories; 368 vulnerabilities; 10% Secunia advisories not patched (Visited on 10/14/10)
    - Existing hypervisor’s code base is growing
      - More vulnerabilities are likely
- It is necessary to ensure hypervisor integrity
Possible Hypervisor Integrity Services

- Integrity measurement of hypervisor
  - Easy to bootstrap, but hard to perform at runtime
- Hypervisor self-protection
  - Make hypervisor more resilient to runtime attacks
- Challenges
  - Hypervisor is the highest privileged software
  - Hypervisor controls the HW and SW

Isolated Execution to Bypass Hypervisor Control

Why Bypassing Hypervisor Control?

- Cloud service provider can see everything in guest workload
  - How to process sensitive data in customers’ workloads?
- It is desirable to allow customers to have isolated execution bypassing hypervisor control
Challenges

- Protection of sensitive data
  - Encrypting sensitive data is easy, but how to handle the decryption keys
  - Need for a key installation service
- Complex transactions
  - How to handle complex transactions
- Performance overhead
  - Cost of “world changes”; much more than switches between guest VMs and hypervisors

HyperSentry: Enabling Stealthy In-context Measurement of Hypervisor Integrity

Challenges

- A fundamental problem
  - How to measure the integrity of the highest privileged software?
- Hypervisor has full control of the software system
  - Scrubbing attacks
  - Tampering with the measurement agent
  - Tampering with the measurement results
- Relying on a higher privileged software goes back to the same problem
The HyperSentry Approach

- HyperSentry
  - A generic framework to stealthily measure the integrity of a hypervisor in its context
- Key ideas
  - Allow the measurement software to gain the highest privilege temporarily
  - Measurement is triggered stealthily
    - Scrubbing attacks
  - Isolate measurement results from the hypervisor

Foundation of HyperSentry

- System Management Mode (SMM)
  - x86 operating mode for system management functions
  - SMRAM can be locked to prevent all access to it except from within the SMM
    - Hypervisor cannot access the SMRAM once locked
  - System Management Interrupt (SMI) only handled by SMI handler in SMRAM
    - SMI bypasses hypervisor’s control
  - Provides the isolation required for HyperSentry
- Main challenges
  - How to retrieve the needed context for hypervisor?
  - How to attest to the measurement output?

Foundation of HyperSentry (Cont’d)

- Out-of-band communication channel
  - Triggers a System Management Interrupt (SMI)
  - Out of the control of the hypervisor
  - Example: IPMI
    - Uses a microcontroller on the motherboard
    - Hard-wired to GPIO chip to trigger SMI
  - Not under the control of the Hypervisor
- Main challenge
  - How to prevent or detect hypervisor’s intervention (e.g., reprogram APIC)?
In-context Integrity Measurement

• Challenges
  – How to detect the intercepted CPU operation mode?
    • Hypervisor or guest VM?
  – How to retrieve the context needed for measurement?
    • E.g., CR3 and page table

• Solution
  – Inject a privileged instruction to force the CPU to fall back to the hypervisor mode
  – Run the measurement agent in the same context as the hypervisor
    • Agent runs in a protected execution environment

In-context Integrity Measurement
Stealthy Invocation

• Is out-of-band invocation sufficient to achieve stealthy invocation?
  – Unfortunately …

A Variation of Scrubbing Attack

Thwarting this Scrubbing Attack

• Can we prevent the hypervisor from blocking SMIs?
  – Not possible with existing hardware

• Solution
  – Detecting fake SMIs generated by the (compromised) hypervisor
    • Verifying status registers to ensure that the measurement is invoked by the out-of-band channel
  – Key reason: HW SMI and SW HMI are distinguishable
Stealthy Invocation

Target Platform (IBM HS21XM Blade Server)

- All status registers are non-writable
- Measurement is invoked only if all other bits are 0
- A fake SMI is easily detectable

CPU Core 0

Memory Controller (North Bridge)

SMI_STS

ALT_GPI_STS

IO Control Hub (South Bridge)

CPU Core 1

CPU Core n

Building Trust Chain

Bootstrapping

 Initialization code

 Measures

 SMRAM

Hypervisor

Measures & verifies

Runtime

TPM

Measures

Hypervisor

Measures

SMRAM

Measures

Hypervisor

Measures & verifies

Attesting to the Measurement Output

• Challenge
  – Absence of a dedicated hardware for attestation
    • The hypervisor controls the hardware most of time

• Solution
  – Providing the SMRAM with a private key
  – Using this key to attest to the measurement results
Attesting to the Measurement Output

HyperSentry Security Properties

- Stealthy Invocation
  - If configurations are not changed → guaranteed by hardware
  - If configurations change → fake SMIs are detectable
- Verifiable Behavior
  - The measurement agent is measured every time before it executes
- Deterministic Execution
  - The measurement agent possesses full control over the system
- In-context privileged measurement
  - Guarantee falling back to the hypervisor mode
  - The measurement agent runs in the same context as the hypervisor
- Attestable output
  - The measurement output is signed by a verifiable and protected key

Verifying the Integrity of Xen

- Xen Code Integrity
  - SHA-1 hash of Xen's code
  - Control flow verification
    - Hardware registers (e.g., IDTR, SYSENTER_EIP)
- An attempt on Xen’s data integrity
  - Inspecting page tables to detect unauthorized sharing of pages across guest virtual machines
**HyperSentry Performance**

- IBM HS21XM blade server
- Measuring the Xen hypervisor
  - End-to-end execution time: 35 ms
  - Periodical measurement:
    - Every 8 seconds: 2.4% overhead; every 16 seconds: 1.3% overhead

![HyperSentry Performance Graph](graph.png)

**Conclusion**

- Security of virtual cloud computing
  - Necessary for new research
  - Potentially fruitful research area
- Security architecture for virtual cloud computing
  - Hypervisor-based runtime security services
  - VM image security services
  - Integrity of hypervisors
  - Isolated execution bypassing hypervisor control
  - *Not necessarily complete*
    - Hopefully a guidance/framework for innovative ideas
- **Stay relevant!!!**