Self-service Cloud Computing

Vinod Ganapathy
vinodg@cs.rutgers.edu

Department of Computer Science
Rutgers University
The modern computing spectrum

- The Cloud
- Web browsers and other apps
- Smartphones and tablets
Security concerns are everywhere!

Can I trust Gmail with my personal conversations?  
Can I trust my browser with my saved passwords?  
Is that gaming app compromising my privacy?
Today’s talk

The Cloud

Web browsers and other apps

Smartphones and tablets
Self-service Cloud Computing

Shakeel Butt
Vinod Ganapathy

H. Andres Lagar-Cavilla
Abhinav Srivastava

Rutgers University

AT&T
What is the Cloud?

A distributed computing infrastructure, managed by 3rd-parties, with which we entrust our code and data.
What is the Cloud?

A distributed computing infrastructure, managed by 3\textsuperscript{rd}-parties, with which we entrust our code and data.

- Comes in many flavours: *-aaS

- Many economic benefits
  - No hardware acquisition/maintainence costs
  - Elasticity of resources
  - Very affordable: a few $\$/hour

• Many new companies & services rely exclusively on the cloud, e.g., Instagram, MIT/Harvard EdX [NYTimes, “Active in Cloud, Amazon Reshapes Computing,” Aug 28, 2012]
Virtualized cloud platforms

Management VM (dom0)  Work VM  Work VM  Work VM

Hypervisor

Hardware

Examples: Amazon EC2, Microsoft Azure, OpenStack, RackSpace Hosting
Embracing the cloud

Let's do Cloud
Embracing the cloud

Trust me with your code & data

Client  Cloud Provider

You have to trust us as well

Cloud operators

Problem #1

Client code & data secrecy and integrity vulnerable to attack

Google Fires Employee Accused Of Spying On Kids

By Phil Villareal on September 16, 2010 9:15 AM
Embracing the cloud

Problem #1: Client code & data secrecy and integrity vulnerable to attack
Embracing the cloud

**Problem #2**

Clients must rely on provider to deploy customized services

---

I need customized malware detection and VM rollback

For now just have checkpointing …
Why do these problems arise?

Management VM (dom0)

Work VM

Work VM

Work VM

Hypervisor

Hardware
Example: Malware detection

[Example: Gibraltar -- Baliga, Ganapathy, Iftode, ACSAC’08]
Problem: Clients must rely on provider to deploy customized services.

Hypervisor

Client's VM

Code

Data

Checking daemon

Management VM

Sec. Policy

Checking

Process the page

Resume guest

Alert user

Clients must rely on provider to deploy customized services.
Problem

Client code & data secrecy and integrity vulnerable to attack

Client’s VM

<table>
<thead>
<tr>
<th>Code</th>
<th>Data</th>
</tr>
</thead>
</table>

Malicious cloud operator

Hypervisor

Resume guest
Alert user

Checking daemon
Process the page

Sec. Policy

Client code & data secrecy and integrity vulnerable to attack

Problem

Client’s VM

<table>
<thead>
<tr>
<th>Code</th>
<th>Data</th>
</tr>
</thead>
</table>

Malicious cloud operator

Hypervisor

Resume guest
Alert user

Checking daemon
Process the page

Sec. Policy

Client code & data secrecy and integrity vulnerable to attack
Problem

Client code & data secrecy and integrity vulnerable to attack

Client’s VM

<table>
<thead>
<tr>
<th>Code</th>
<th>Data</th>
</tr>
</thead>
</table>

Management VM

Checking daemon

Process page

Sec.

Policy

EXAMPLES:

• CVE-2007-4993. Xen guest root escapes to dom0 via pygrub
• CVE-2007-5497. Integer overflows in libext2fs in e2fsprogs.
• CVE-2008-0923. Directory traversal vulnerability in the shared folders feature for VMWare.
• CVE-2008-1943. Buffer overflow in the backend of XenSource Xen paravirtualized frame buffer.
• CVE-2008-2100. VMWare buffer overflows in VIX API let local users execute arbitrary code in host OS.

.... [AND MANY MORE]
Our solution

**SSC**: Self-service cloud computing

- Management VM
- Client’s VMs
- Hypervisor
- Hardware
Outline

• Disaggregation and new privilege model
• Technical challenges:
  – Balancing provider’s and client’s goals
  – Secure bootstrap of client’s VMs
• Experimental evaluation
• Future directions and other projects
Duties of the management VM

Manages and multiplexes hardware resources

Manages client virtual machines

Management VM (Dom0)
Main technique used by SSC

Disaggregate the management VM

System-wide Mgmt. VM (SDom0)

- Manages hardware
- No access to clients' VMs

Solves problem #1

Per-Client Mgmt. VM (UDom0)

- Manages client's VMs
- Allows clients to deploy new services

Solves problem #2
Embracing first principles

Principle of separation of privilege

System-wide Mgmt. VM (SDom0)

Per-Client Mgmt. VM (UDom0)
Embracing first principles

Principle of least privilege

System-wide Mgmt. VM (SDom0)

Per-Client Mgmt. VM (UDom0)
An SSC platform

SSC Hypervisor

Hardware

Equipped with a Trusted Platform Module (TPM) chip

SDom0

UDom0

Client’s meta-domain

Work VM

Service VM

Work VM
 SSC’s privilege model

Privileged operation

Self-service hypervisor

Is the request from client’s Udom0?

YES

DOES requestor have privilege (e.g., client’s service VM)

YES

ALLOW

NO

DENY

NO

ALLOW
Key technical challenges

1. Providers want some control
   – To enforce regulatory compliance (SLAs, etc.)
   – **Solution**: Mutually-trusted service VMs

2. Building domains in a trustworthy fashion
   – Sdom0 is not trusted
   – **Solution**: the Domain Builder

3. Establishing secure channel with client
   – Sdom0 controls all the hardware!
   – **Solution**: Secure bootstrap protocol
Providers want *some* control

- Udom0 and service VMs put clients in control of their VMs
- Sdom0 cannot inspect these VMs
- Malicious clients can misuse privilege
- Mutually-trusted service VMs
Trustworthy regulatory compliance

SDom0

UDom0

Mutually-trusted Service VM

Work VM

Work VM

SSC Hypervisor

Hardware
Bootstrap: the Domain Builder

SSC Hypervisor

Hardware

SDom0

Bootstrap: the Domain Builder

UDom0

Domain Builder

Work VM

Service VM
Bootstrap: the Domain Builder

Must establish an encrypted communication channel

SSC Hypervisor

Hardware
Secure bootstrap protocol

- **Goal**: Build Udom0, and establish an SSL channel with client
- **Challenge**: Sdom0 controls the network!
- **Implication**: Evil twin attack
An evil twin attack
1. Udom0 image, Enc (1, 0)
DomB builds domain
DomB installs key, nonce

Enc (key, nonce)

Domain Builder

UDom0

SSC Hypervisor

Hardware
Client gets TPM hashes
5 Udom0 sends to client
6 Client sends Udom0 SSL key
SSL handshake and secure channel establishment
Can boot other VMs securely
Client meta-domains

SSC hypervisor

Hardware

Udom0

Mutually-trusted Service VMs

- Regulatory compliance
- Trustworthy metering

Service VMs

- Storage services
- Firewall and IDS
- Malware detection

Computation

Work VM

Work VM

Work VM

Work VM
Case studies: Service VMs

• Storage services: Encryption, Intrusion detection
• Security services:
  – Kernel-level rootkit detection
  – System-call-based intrusion detection
• Data anonymization service
• Checkpointing service
• Memory deduplication
• And compositions of these!
Evaluation

• Goals
  – Measure overhead of SSC

• Dell PowerEdge R610
  – 24 GB RAM
  – 8 XEON cores with dual threads (2.3 GHz)
  – Each VM has 2 vCPUs and 2 GB RAM

• Results shown only for 2 service VMs
  – Our ACM CCS’12 paper presents many more
Storage encryption service VM

Sdom0

Backend Block device

Client’s work VM

Frontend Block device
Storage encryption service VM

<table>
<thead>
<tr>
<th>Platform</th>
<th>Unencrypted (MB/s)</th>
<th>Encrypted (MB/s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Xen-legacy</td>
<td>81.72</td>
<td>71.90</td>
</tr>
<tr>
<td>Self-service</td>
<td>75.88</td>
<td>70.64</td>
</tr>
</tbody>
</table>
Checkpointing service VM
Checkpointing service VM

<table>
<thead>
<tr>
<th>Platform</th>
<th>Unencrypted (sec)</th>
<th>Encrypted (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Xen-legacy</td>
<td>1.840</td>
<td>11.419</td>
</tr>
<tr>
<td>Self-service</td>
<td>1.936</td>
<td>11.329</td>
</tr>
</tbody>
</table>
## Related projects

<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Protect client VM data from Dom0 using a thin, bare-metal hypervisor</td>
<td>Allow clients to have their own Dom0s on commodity clouds using a thin shim</td>
</tr>
</tbody>
</table>

![Diagram showing the relationship between Dom0, Client VM, Nested Hypervisor, CloudVisor, and XenBlanket.]
SSC is a cloud model that …

- Improves security and privacy of client code and data
- Enhances client control over their VMs
- Imposes low runtime performance overheads
- Provides a rich source of problems for follow-up work 😊
Future vision for SSC

• **Cloud app** markets:
  – Marketplaces of service VMs.
  – **Research problems**: Ensuring trustworthiness of apps, enabling novel mutually-trusted apps, App permission models.

• **Migration-awareness**:
  – Policies and mechanisms for VM migration in SSC.
  – **Research problems**: Prevent exposure of cloud infrastructure details to competitors, TPM-based protocols that are migration-aware.
Other research projects
The Cloud

The browser

The smartphone
Smartphone rootkits

New techniques to detect OS kernel-level malware

• Rootkits operate by maliciously modifying kernel code and data

RESULTS:
• New techniques to detect data-oriented rootkits [ACSAC’08] 🏆
• Exploring the rootkit threat on smartphones [HotMobile’10] 📢
• Security versus energy tradeoffs in detecting rootkits on mobile devices [MobiSys’11]
Securing Web browsers

Studying information leakage via 3\textsuperscript{rd} party browser addons

- Addons are untrusted, privileged code
  - All major browsers support addons
  - Can leak sensitive information

RESULTS:

- Information flow tracking-enhanced browser [ACSAC’09]
- Static capability leak analysis for Mozilla Jetpack [ECOOP’12]
- New bugs found in Mozilla extensions
And many more …

• The Cloud (and other software systems)
  [CCS08, ACSAC08a, ACSAC09a, RAID10, TDSC11, CCS12a, CCS12b, ANCS12]
  – Security remediation using transactional programming
  – Fast, memory-efficient network intrusion detection

• The browser (and the Web)
  [ACSAC08b, ACSAC09b, ECOOP12a, ECOOP12b]
  – Secure mashup Web applications
  – Integrating the Web and the cloud
  – Isolation as a first-class JavaScript feature

• The smartphone (and other mobile devices)
  [UbiComp09, SACMAT09, HotMobile10, MobiSys11]
  – Location privacy in mobile computing
  – Secure remote access to enterprise file systems
Looking into the future...
Active ongoing projects

SSC++

Improving browser extension security

Improving mobile app security
Collaborators and students

And many other camera-shy folks!
Fast and memory-efficient NIDS

Using ordered binary decision diagrams (OBDDs) to address time/space tradeoff in regexp matching

- Regexp matching a basic primitive in many NIDS and firewalls
- Fundamental time/space tradeoff:
  - DFAs are fast but memory intensive
  - NFAs are memory efficient but slow

**MAIN RESULT:**
- Encoding NFAs using OBDDs
- Obtains NFA-like memory consumption with DFA-like speed

[RAID’10, COMNET’11, ANCS’12]
Transactional introspection

Security using transactional programming and machinery

```javascript
dispatch_request (){
    transaction [ principal ]{
        ...
        perform_request ( );
        ...
    } /* Commits only if all authorization succeeds */
}
```

- Enforcing authorization policies with stronger guarantees [CCS’08]
- Detecting data structure corruptions [RV’11]
- Sandboxing untrusted JavaScript code using transactions [ECOOP’12]

**BENEFIT:** Security remediation for free