Module: Cloud Computing Security

Professor Trent Jaeger
Penn State University
Cloud Computing Is Here

Why not use it?
What’s Happening in There?
Overview

- **Cloud computing** replaces physical infrastructure

- Is it safe to **trust** these services?
From Data Center to Cloud

Traditional Data Center
- Proprietary, customized
- Full Control
- Most Secure
- Dedicated

vs.

Cloud Computing Service
- Standardized
- Partial Control
- Secure
- Shared

Other enterprises

Enterprise

Local Network

Cloud (Internet)

From http://blogs.zdnet.com/Hinchcliffe
Reasons to Doubt

- History has shown they are **vulnerable to attack**
  - SLAs, audits, and armed guards offer few guarantees
  - **Insiders** can subvert even hardened systems

**Data Loss Incidents**

<table>
<thead>
<tr>
<th>Year</th>
<th>Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>‘06</td>
<td>641</td>
</tr>
<tr>
<td>‘07</td>
<td>770</td>
</tr>
<tr>
<td>‘08</td>
<td>986</td>
</tr>
<tr>
<td>‘09</td>
<td>695</td>
</tr>
<tr>
<td>‘10</td>
<td>678</td>
</tr>
<tr>
<td>‘11</td>
<td>903</td>
</tr>
</tbody>
</table>

**Incident Attack Vector**

- **External**: 54%
- **Accidental**: 23%
- **Insider**: 16%
- **Unknown**: 7%

*Credit: The Open Security Foundation datalossdb.org*
Cloudy Future

• New problem or new solution?
  ‣ New challenges brought on by the cloud (plus old ones)
  ‣ Utility could provide a foundation for solving such challenges
Cloudy Future

- Improve on data centers? On home computing?
  - Seems like a low bar
What is Cloud Computing?

• Cloud vendor provides managed computing resources for rent by customers

• What do you want to rent?
  ‣ (Virtualized) Hosts (Infrastructure as a Service)
    • Rent cycles: Amazon EC2, Rackspace Cloud Servers, OpenStack
  ‣ Environment (Platform as a Service)
    • Rent instances: Microsoft Azure, Google App Engine
  ‣ Programs (Software as a Service)
    • Rent services: Salesforce, Google Docs

• Other variations can be rented
What is Cloud Computing?

- **IaaS** (Infrastructure as a Service)
- **PaaS** (Platform as a Service)
- **SaaS** (Software as a Service)

Value Visibility to End Users

End Users

Application Developers

Network Architects
IaaS Platform: OpenStack
PaaS Platform: Google App

• Platform for deploying language-specific apps
  ‣ Java, Python, PHP, etc.

• **Vendor** provides OS and middleware
  ‣ E.g., Web server, interpreters

• **Customers** deploy their customized apps
  ‣ You focus on custom code

• **Clients** use these apps
  ‣ Analogously to IaaS
How to Build an IaaS Cloud?

- **Vendors** obtain **hardware resources** for
  - Various **cloud services**: API, Messages, Storage, Network, ...
  - **Compute nodes** for running customer workloads

- **Install** your hardware
  - Need to choose software configurations specific for services and compute nodes

- **Start** your hosts
  - Join the cloud - services and available compute nodes

- **Now your cloud is running**
  - **Have fun!** Customers are ready to use your services and nodes
How to Use an IaaS Cloud?

• **Customers choose an OS distribution**
  ‣ These are published by the cloud vendor and others
  ‣ Obtain cloud storage necessary to store these and your data

• **Configure your instance (VM)**
  ‣ Prior to starting - enable you to login and others to access the instance’s services

• **Start your instance**
  ‣ Boots the chosen OS distribution with the configurations

• **Now your instance is running**
  ‣ **Have fun!** Login via SSH or ready for your clients
Multiple Stakeholders

- Are my data protected?
- Are my services running correctly?
- Is my platform secure?
Cloud Complexity

• Cloud environment challenges
  ‣ Opaque, Complex, Dynamic
  ‣ Insiders, Instances, Co-hosting
What Could Go Wrong?

- What do customers depend on from the cloud?
  - Trust Model
  - Are those parties worthy of our trust?

- Who are potential adversaries in the cloud?
  - Threat Model
  - Are customers protected from their threats?

- What would be ideal from a security standpoint?
  - Ideal Security Model
  - How many trusted parties and how many threats?
Consumers use published instances

Provider

Publisher

Consumer

Who do you trust? What are threats?
• Publisher left an SSH user authentication key in their AMI

• Fortunately, Amazon agreed that this is a violation
  ‣ Unfortunately, it was not an isolated problem
    • 30% of 1100 AMIs checked contained such a key
  ‣ Also, pre-configured AMIs had SSH host keys
    • Thus, all instances use the same host key pair
    • Implications?
Security Configuration

- Zillions of security-relevant configurations for instances
  - Do you have the right code and data installed?
  - Are you running the expected code?
  - Discretionary access control
  - Firewalls
  - Mandatory access control
    - SELinux, AppArmor, TrustedBSD, Trusted Solaris, MIC
  - Application policies (e.g., Database, Apache)
  - Pluggable Authentication Modules (PAM)
  - Application configuration files
- Plus new configuration tasks for the cloud - e.g., storage
Cloud Service Vulnerabilities

- **Vulnerabilities** have been found in cloud services
  - E.g., OpenStack identity service, web interface, and API service
- Adversaries who compromise such services may launch a variety of attacks
  - E.g., Key Injection Attack

![Diagram of Key Injection Attack]

`nova keypair-add mykey`

**Step 1**

`nova boot --key-name mykey`

**Step 2**

**API Service**

**Database**

**API Service**

**Compute Service**

- `mykey : ssh-rsa ABC`
Although the vendor may have a good reputation, not every employee may
Insider Threats

• May trust the cloud vendor company
  ‣ But, do you trust all its employees?

• Insiders can control platform
  ‣ Determine what software runs consumers’ code

• Insiders can monitor execution
  ‣ Log instance operation from remote

• Insiders may have physical access
  ‣ Can monitor hardware, access physical memory, and tamper secure co-processors
Co-Hosting Threats

• An instance co-hosted on the same physical platform could launch attacks against your instance

• Co-hosted instances share resources
  ‣ Computer
    • CPU, Cache, Memory, Network, etc.

• Shared resources may be used as side channels to learn information about resource or impact its behavior
Resource Freeing Attacks

• Setup

• Victims
  ‣ One or more VMs with public interface

• Beneficiary
  ‣ VM whose performance we want to improve (content over target resource)

• Helper
  ‣ Mounts attack using public interface
Resource Freeing Attacks

• Resource contention over the CPU
  ‣ Schedule beneficiary more frequently

• Attack: shift resource usage via public interface
  ‣ Normally, victim is scheduled and pollutes the cache
  ‣ Approach lower scheduling priority
    • Make victim appear CPU-bound

![Graph showing performance impact with and without RFA]

- RFA intensities – time in ms per second
- 60% Performance Improvement
- 196% slowdown
- 86% slowdown
Preventing Vulnerabilities

• How would you prevent these threats?
  ‣ Misconfigured instances
  ‣ Untrusted cloud services
  ‣ Insiders
  ‣ Side channels
  ‣ (Attacks to cloud platform also)
Verifiable Computation

• Your services are black boxes - to the cloud!
  ‣ Send a program and encrypted data
  ‣ Program computes over encrypted data
  ‣ Scheme: KeyGen (for Program), Compute (Program), Verify

Depends on heavy crypto - homomorphic encryption
Pinocchio [Oakland 2013]

- New cryptographic protocol for **general-purpose public verifiable computation** with support for zero-knowledge arguments
- Big advance: **Performance**
- Encoding in “**quadratic programs**”; signature depends only on security constant
  - Idea behind quadratic arithmetic programs: each multiplication gate is a “small expression”. Construct polynomials that encode the equations, such that if the evaluation is correct, then $D(z) / P(z)$. Then the protocol just checks divisibility randomly
- **Beats local C execution** (for verification)
Integrity Monitor Concept

• **Integrity monitor** similar to a reference monitor
  ‣ Mediate access to service based on integrity criteria

• **Challenges**
  ‣ Where do we measure integrity-relevant events?
  ‣ How do we verify ongoing integrity?
  ‣ How can we deploy this in a cloud environment?
• **Policy-sealed data** [*USENIX Sec 2012b*]
  ‣ Do not release my data to the cloud until that cloud satisfies my requirements
  ‣ **Customer-chosen policy**

• **How to ensure that only nodes that satisfy customer-chosen policy get data?**
  ‣ **Attribute-based encryption**
  ‣ Encrypt data using ABE description of load-time configuration
  ‣ A verifiable monitor is trusted to delegate correct credentials to nodes (using hardware-based attestations - e.g., via TPM)
Excalibur Approach

- Check node configurations
  - Monitor attests nodes in background
- Scalable policy enforcement
  - CP-ABE operations at client-side lib

From Nuno Santos’ slides
Runtime Monitoring

- Excalibur does not address runtime issues with instance
  - Customers may want to ensure that clients of their services only receive communications from satisfactory instances
  - Customer may want to take remediative actions
**Integrity Verification Proxy**

- Clients specify criteria to be enforced by a channel mediator *[TRUST 2012]*

- Set of measurement modules verifies the criteria
  - Loadtime modules measure VM components
  - VM Introspection to examine runtime criteria
    - E.g., Binaries/data loaded, enforcement disabled, policy changes, kernel data (binary handler), etc.

![Diagram showing the process of Integrity Verification Proxy with steps (1) Register criteria, (2) Verify Monitor / Node, (3) Verify VM, (4) Connect, and (5) Report Violation.]}
Cloud Anchor [CCSW 2010, TrustCom 2012] +IVP in OpenStack [CSAW 2013]

Client monitors CV and cloud criteria
CV monitors cloud node
IVP monitors cloud instance

Client provides criteria
Client criteria sent to IVP

Cloud Verifier

IVP Node

Cloud Instance

Client stops using Cloud
Disable Cloud Node
Block connection at the Cloud Node
Customer-Driven Monitoring

- CV/IVP Limitation
  - IVP must be trusted by cloud vendor
  - Part of management VM

- What if you need to perform monitoring that the cloud vendors will not support?
Self-Service Clouds

- Customizable cloud platform stack \([\text{CCS 2012}]\)

Why do these problems arise?

- Management VM (dom0)
- Work VM
- Work VM
- Work VM

Hypervisor

Hardware

Slides courtesy of Vinod Ganapathy
Self-Service Clouds

- Customizable cloud platform stack [CCS 2012]
Self-Service Clouds

- Customizable cloud platform stack \[\text{CCS 2012}\]
  - UDom0 boots customer-defined Service VMs

UDom0 boots customer-defined Service VMs

Equipped with a Trusted Platform Module (TPM) chip
Take Away

• Cloud computing is here to stay
  ‣ In some form

• May be a solution or a problem or both
  ‣ Introduces new types of vulnerabilities into systems we ran on data centers - which had vulnerabilities to begin with

• Ultimately, have to improve service providers’ jobs
  ‣ Make it easy to ensure that systems perform as expected

• Two possible methods
  ‣ Verifiable computation and instance monitoring