



# **Security** and **Privacy** in **Cloud Computing**

**Ragib Hasan**

Johns Hopkins University  
en.600.412 Spring 2010

**Lecture 3**  
02/15/2010

# Mapping/topology Attacks

## Lecture Goal

- Learn about mapping attacks
- Discuss different techniques and mitigation strategies
- Analyze the practicality and impact

## Reading:

Hey, You, Get Off of My Cloud: Exploring Information Leakage in Third-Party Compute Clouds, Ristenpart et al., CCS 2009

# Why Cloud Computing brings new threats?

Traditional system security mostly means keeping bad guys out

The attacker needs to either compromise the auth/access control system, or impersonate existing users



# Why Cloud Computing brings new threats?

But clouds allow **co-tenancy** :



Multiple independent users share the same physical infrastructure

So, an attacker can legitimately be in the same physical machine as the target

# Challenges for the attacker

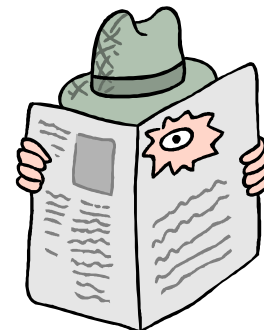
How to find out **where** the target is located



How to be **co-located** with the target in the same (physical) machine



How to **gather information** about the target



## Hey, You, Get Off of My Cloud: Exploring Information Leakage in Third-Party Compute Clouds, Ristenpart et al., CCS 2009

- First work on **cloud cartography**
- Attack launched against commercially available **“real” cloud** (Amazon EC2)
- Claims up to 40% success in co-residence with target VM

# Strategy

- **Map** the cloud infrastructure to find where the target is located
- Use various **heuristics** to determine co-residency of two VMs
- Launch **probe VMs** trying to be co-resident with target VMs
- Exploit **cross-VM leakage** to gather info about target

# Threat model

## Attacker model

- Cloud infrastructure provider is trustworthy
- Cloud insiders are trustworthy
- Attacker is a malicious third party who can legitimately use the cloud provider as a client

## Assets

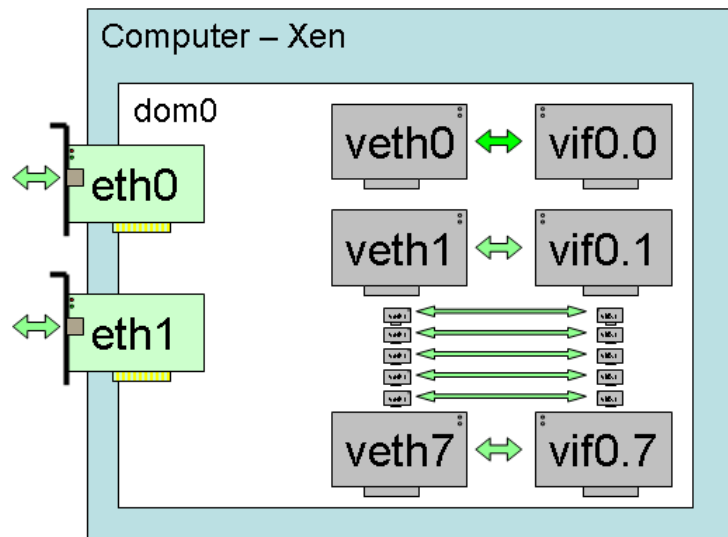
- Confidentiality aware services run on cloud
- Availability of services run on cloud

# Tools of the trade

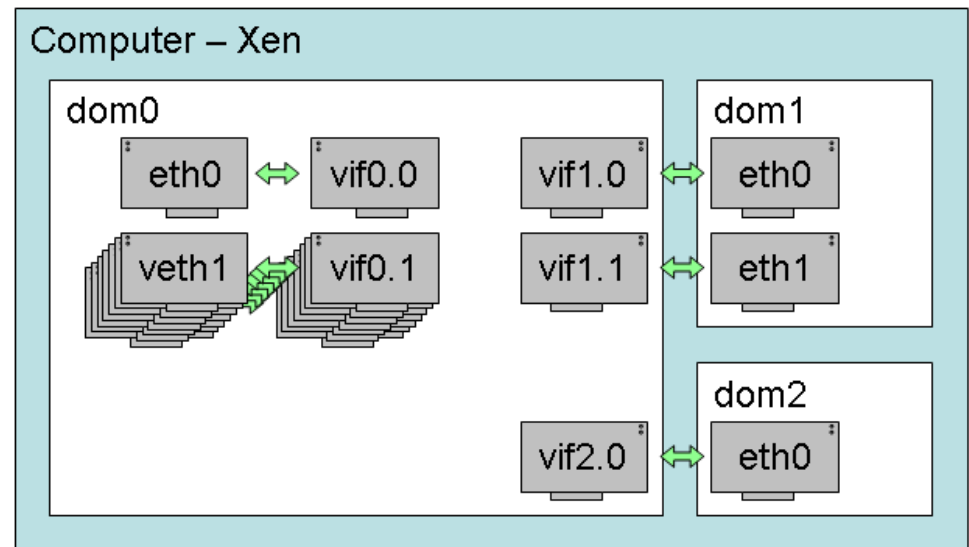
- Nmap, hping, wget for network probing
- Amazon EC2's own DNS to map dns names to IPs

# Sidenote: EC2 configuration

EC2 uses **Xen**, with up to 8 instances per physical machine



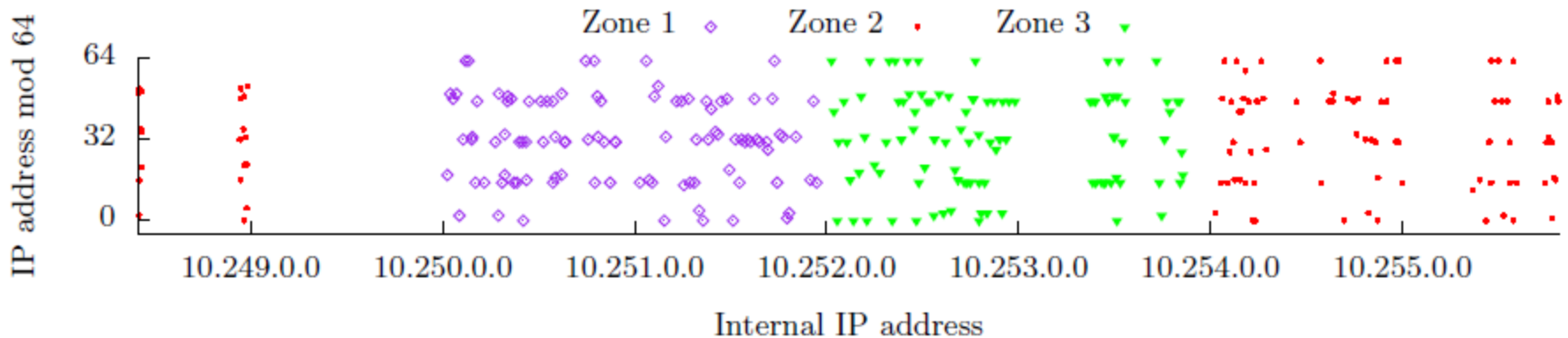
Dom0 is the first instance on the machine, connected to physical adapter



All other instances route to external world via dom0

[Figures from Xen Wiki]

# Task 1: Mapping the cloud



Different availability zones use different IP regions.

**Each instance has one internal IP and one external IP. Both are static.**

For example:

External IP: *75.101.210.100*

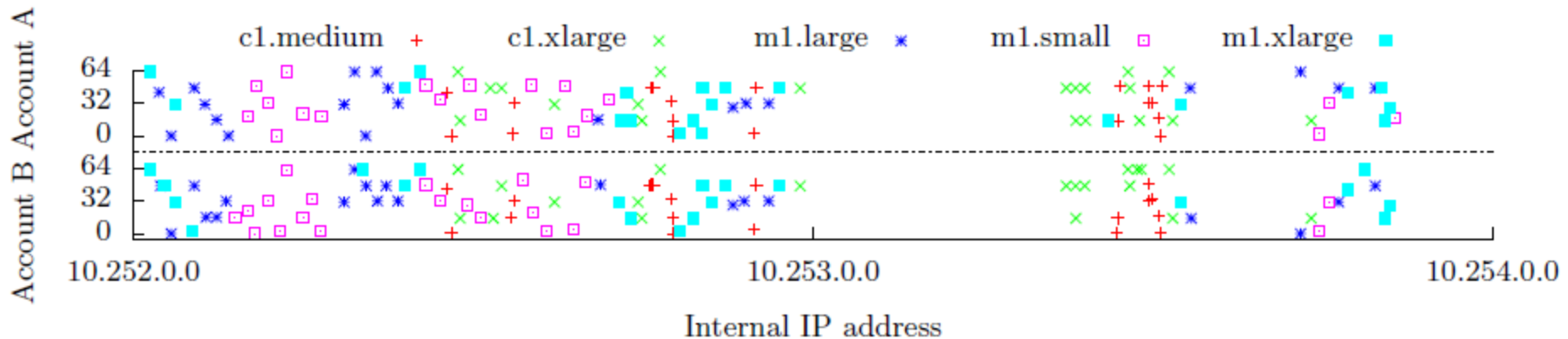
External Name: *ec2-75-101-210-100.computer-1.amazonaws.com*

Internal IP: *10.252.146.52*

Internal Name: *domU-12-31-38-00-8D-C6.computer-1.internal*

**Reverse engineering** the VM placement schemes provides useful heuristics about EC2's strategy

# Task 1: Mapping the Cloud



**Finding:** same instance type within the same zone = similar IP regions

## Reverse engineered mapping decision heuristic:

A /24 inherits any included sampled instance type.

A /24 containing a Dom0 IP address only contains Dom0 IP address.

All /24's between two consecutive Dom0 /24's inherit the former's associated type.

# Task #2: Determining co-residence



- **Co-residence:** Check to determine if a given VM is placed in the same physical machine as another VM
- Network based check:
  - Match Dom0 IP addresses, check packet RTT, close IP addresses (within 7, since each machine has 8 VMs at most)
  - Traceroute provides Dom0 of target
  - No false positives found during experiments

# Task #3: Making a probe VM co-resident with target VM

## Brute force scheme

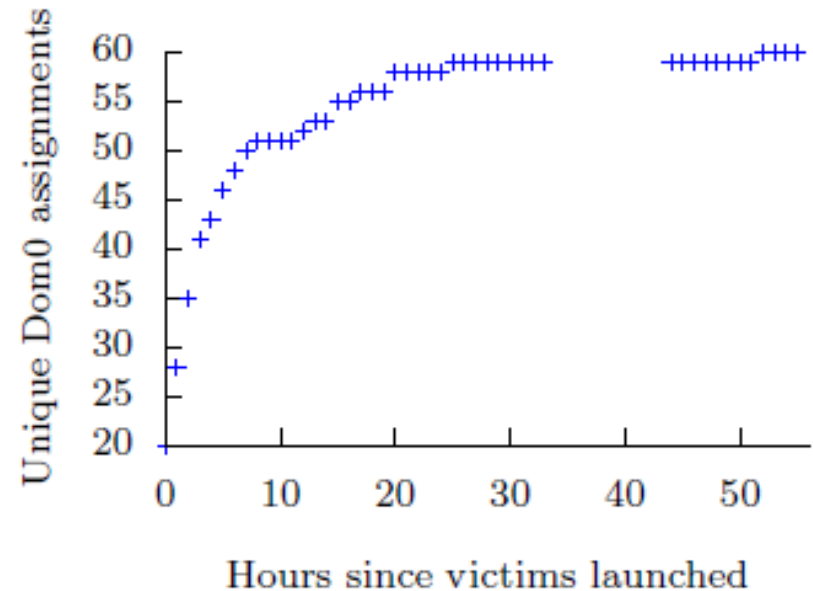
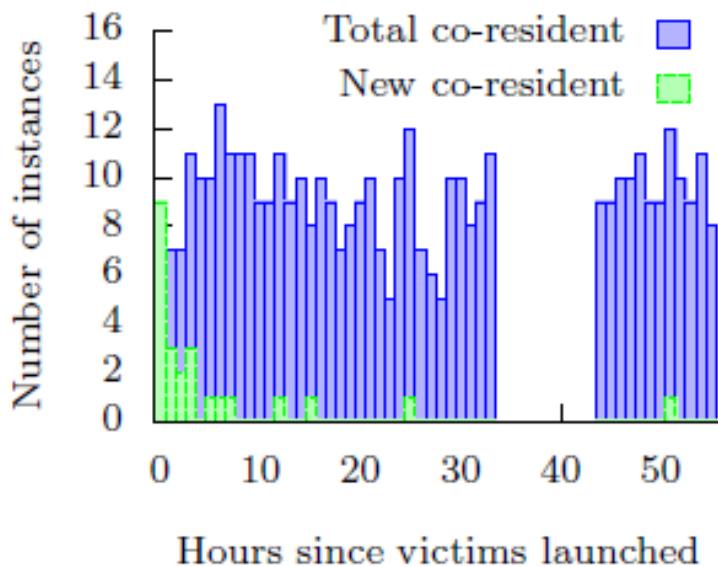
- **Idea:** figure out target's availability zone and type
- Launch many probe instances in the same area
  
- Success rate: 8.4%

# Task #3: Making a probe VM co-resident with target VM

## **Smarter strategy: utilize locality**

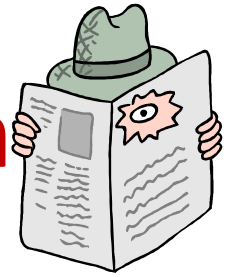
- **Idea:** VM instances launched right after target are likely to be co-resident with the target
- Paper claims 40% success rate

# Task #3: Making a probe VM co-resident with target VM



Window of opportunity is quite large, measured in days

# Task #4: Gather leaked information



Now that the VM is co-resident with target, what can it do?

- Gather information via side channels
- Perform DoS

# Task 4.1: Gathering information

If VM's are separated and secure, the **best** the attacker can do is to gather information

- Measure latency of cache loads
- Use that to determine
  - Co-residence
  - Traffic rates
  - Keystroke timing

# Mitigation strategies #1: Mapping

- Use a randomized scheme to allocate IP addresses
- Block some tools (nmap, traceroute)

# Mitigation strategies #2: Co-residence checks

- Prevent traceroute (i.e., prevent identification of dom0)

# Mitigation strategies #3: Co-location

- Not allow co-residence at all
  - Beneficial for cloud user
  - Not efficient for cloud provider

# Mitigation strategies #4: Information leakage

- Prevent cache load attacks?

# Discussion

- How is the problem different from other attacks?
- What's so special about clouds?

# Discussion

## Cons

- Are the side channels *\*really\** effective?



## Further Reading

### **Amazon downplays report highlighting vulnerabilities in its cloud service**

Hypothetical example described in report much harder to pull off in reality, company says  
TechWorld, Oct 29, 2009. <http://bit.ly/dvxEZp>