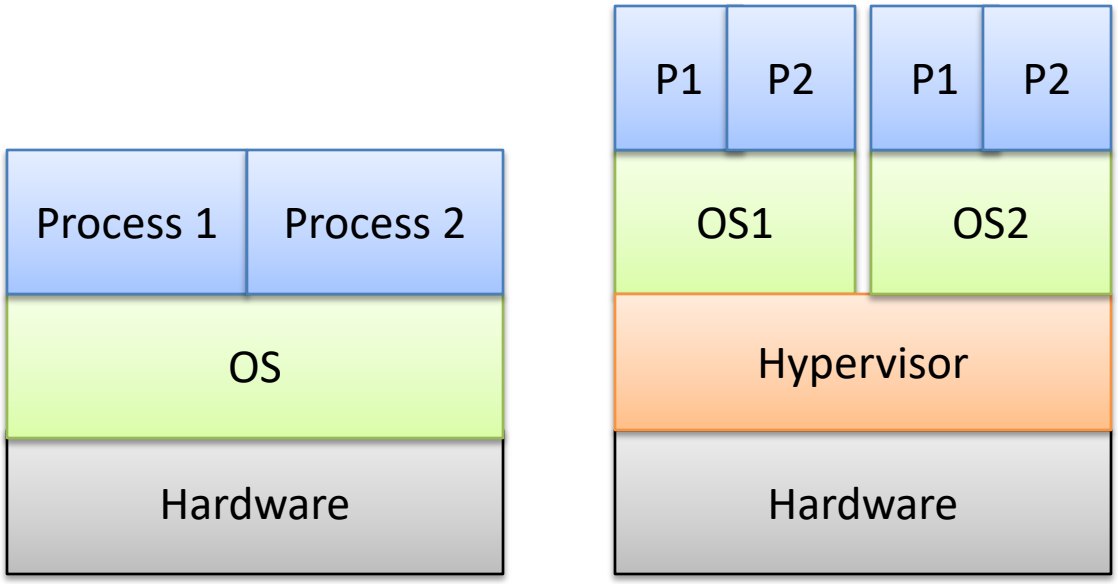
Virtualization security

CS 5450

Tom Ristenpart
https://rist.tech.cornell.edu

Virtualization

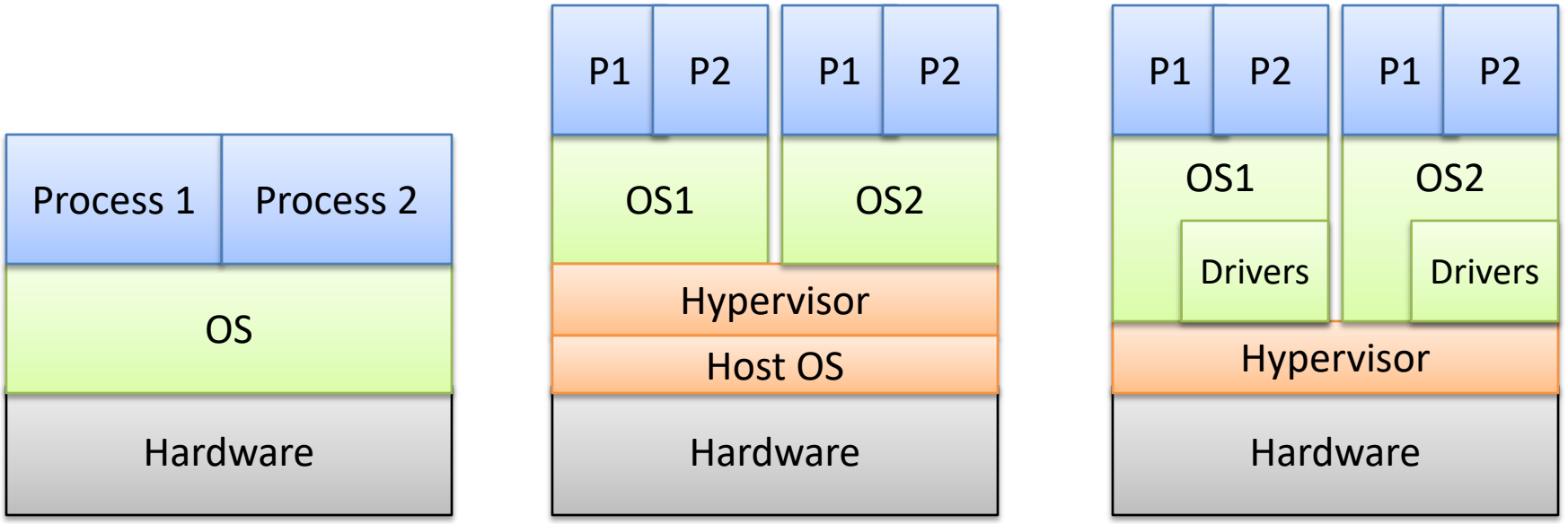


No virtualization

Full virtualization

Type-1: Hypervisor runs directly on hardware

Virtualization



No virtualization

Full virtualization

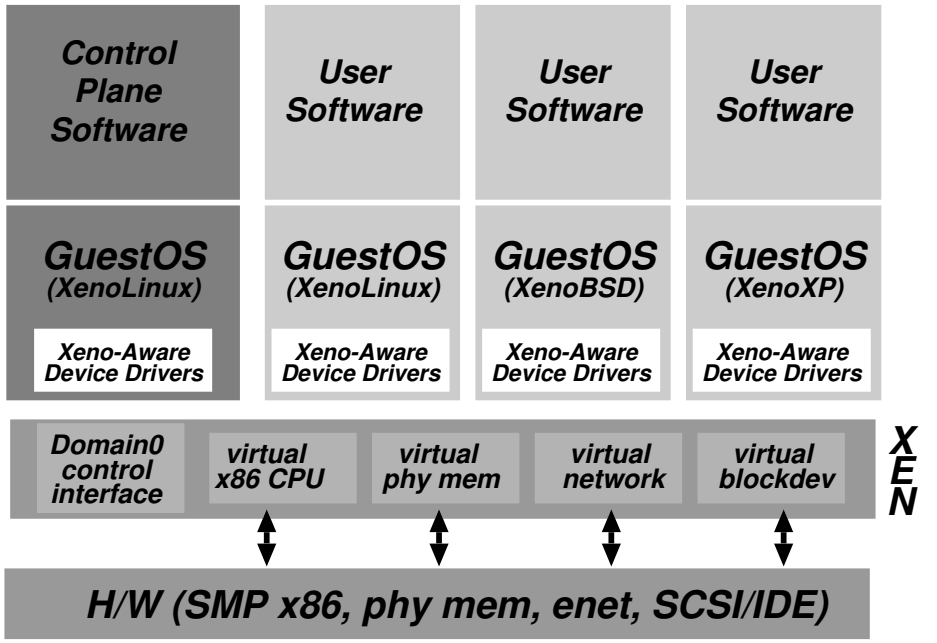
Paravirtualization

Type-1: Hypervisor runs directly on hardware

Type-2: Hypervisor runs on host OS

Xen

- 2003: academic paper
  - "Xen and the Art of Virtualization"
- Paravirtualization
  - Hypercalls vs system calls
  - Modified guest OS
  - Each guest given 1 or more VCPUs
- Why paravirtualization?



Other modern VM solutions

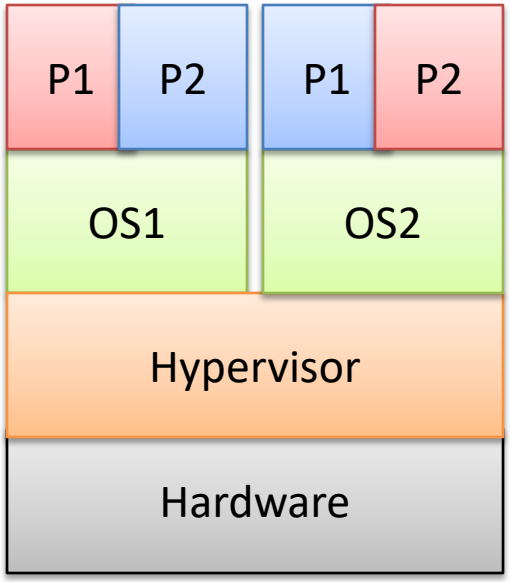
- VMWare
- Virtual Box
- KVM

Example VM Use Cases

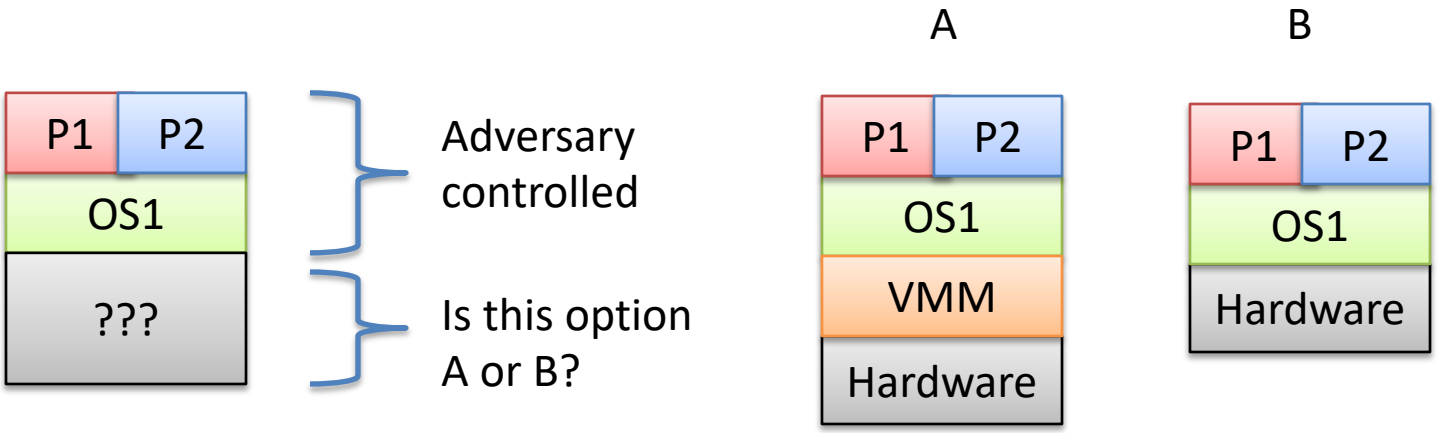
- Legacy support
- Development
- Server consolidation
- Cloud computing Infrastructure-as-a-Service
- Sandboxing / containment

Study of malware

- Researchers use VMs to study malware
- Example of VM sandboxing
  - Hypervisor must contain malicious code
- Introspection (making sense of guest behavior from hypervisor) can be challenging
- How would you evade analysis as a malware writer?
  - split personalities



VMM Transparency



- Adversary can detect if:
  - Paravirtualization
  - Logical discrepancies
    - Expected CPU behavior vs virtualized
    - Red pill (Store Interrupt Descriptor Table instr)
  - Timing discrepancies
    - Slower use of some resources

Garfinkel et al. "Compatibility is not transparency: VMM Detection Myths and Reality"

Detection of VMWare

```
MOV EAX,564D5868 <-- "VMXh"
MOV EBX,0
MOV ECX,0A
MOV EDX,5658 <-- "VX"
IN EAX,DX <-- Check for VMWare
CMP EBX,564D5868
```

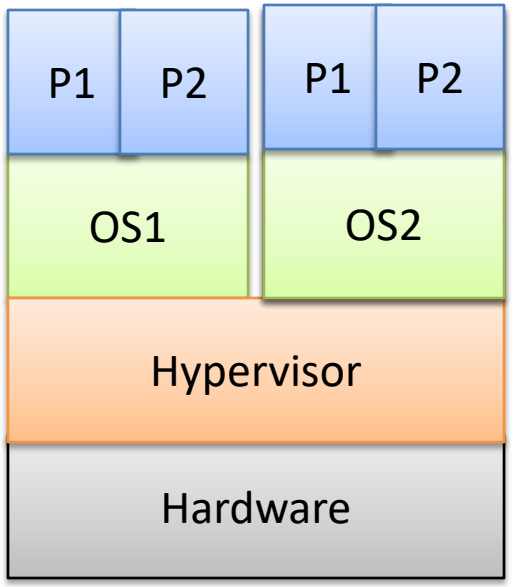
IN instruction used by VMWare to facilitate host-to-guest communication

VMWare:
    places VMXh in EBX
Physical:
    processor exception

From
http://handlers.sans.org/tliston/ThwartingVMDetection_Liston_Skoudis.pdf

Server consolidation

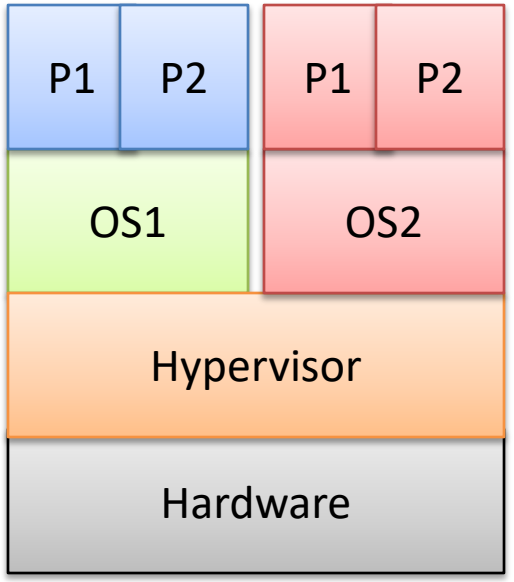
- Consolidation
  - Use VMs to optimize use of hardware
  - Pack as many VMs onto each server as possible
  - Turn off other servers



- Threat model?
  - Containment
  - Isolation
  - Assume guests are/can be compromised

Violating containment

- Escape-from-VM
  - Vulnerability in VMM or host OS (e.g., Dom0)
  - Somewhat rare, but exist and very high value



VMware vulnerability allows users to escape virtual environment

By Joab Jackson Feb 28, 2008

A new vulnerability found in some VMware products allows users to escape their virtual environments and muck about in the host operating system, penetration testing software firm Core Security Technologies announced earlier this week.

This vulnerability (CVE Name: CVE-2008-0923) could poise significant risks to enterprise users who are deploying VMware software as a secured environment.

NEWS

Xen hypervisor faces third highly critical VM escape bug in 10 months

The Xen paravirtualization mode is proving to be a constant source of serious vulnerabilities, allowing attackers to escape from virtual machines

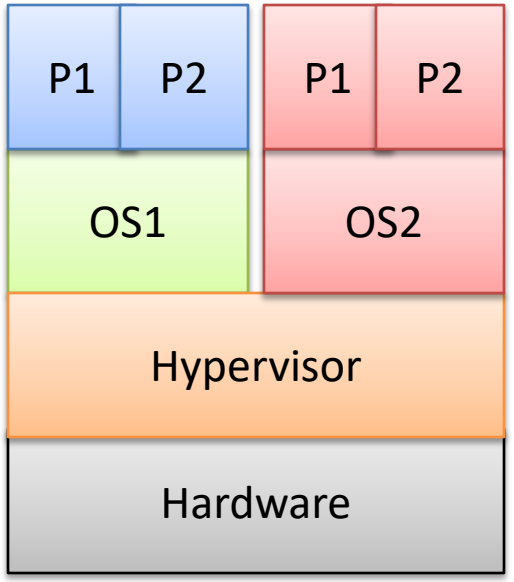
https://blog.quarkslab.com/xen-exploitation-part-2-xsa-148-from-guest-to-host.html#id17

XSA-148 vulnerability

- Hypervisor must ensure guests can't read/write arbitrary physical memory addresses
- Xen uses ***direct paging***: guest VMs use virtual machine to physical address page tables
  - Must use hypercalls to update page tables
  - ***Security critical invariants must be enforced by hypervisor***
- XSA-148: Logic error in page table mapping checks, allowing guest to build page table mapping virtual addresses to ***any*** physical address
- ***Arbitrary read/write access to all of system memory***

Server consolidation

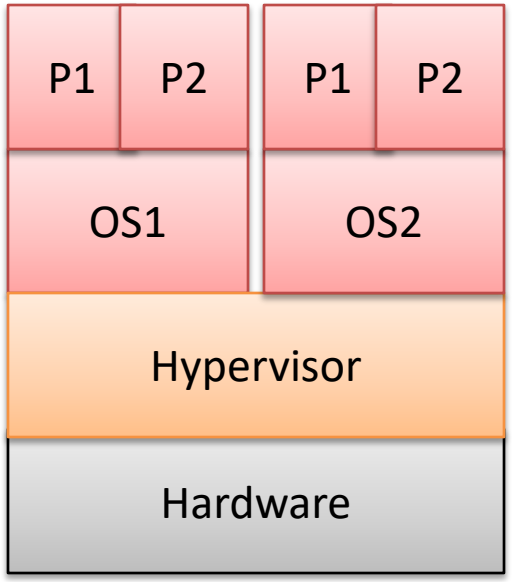
- Consolidation
  - Use VMs to optimize use of hardware
  - Pack as many VMs onto each server as possible
  - Turn off other servers

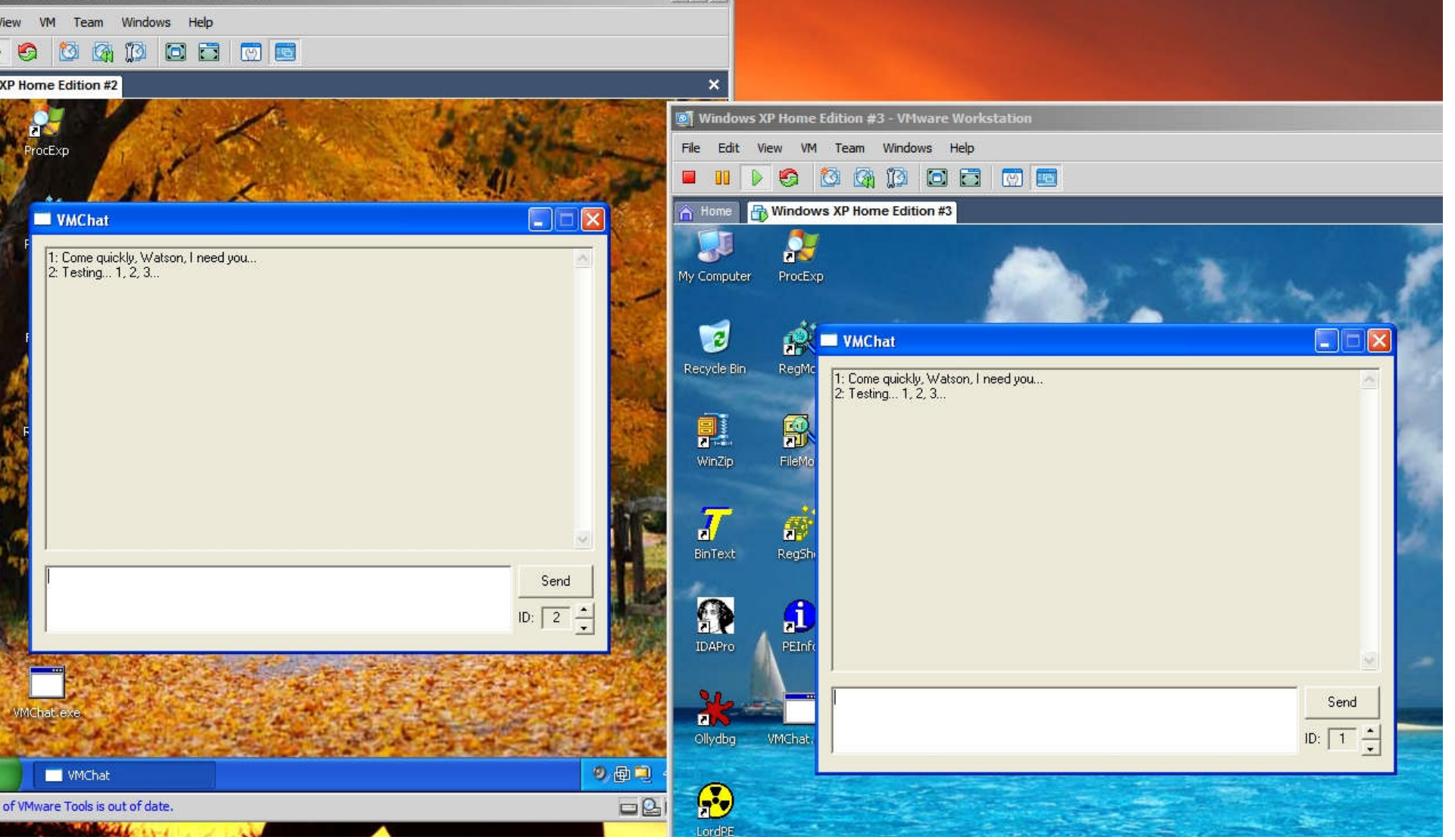


- Threat model?
  - Containment
  - Isolation
  - Assume guests are/can be compromised

Violating isolation

- Covert channels between VMs circumvent access controls
  - Bugs in VMM
  - Side-effects of resource usage

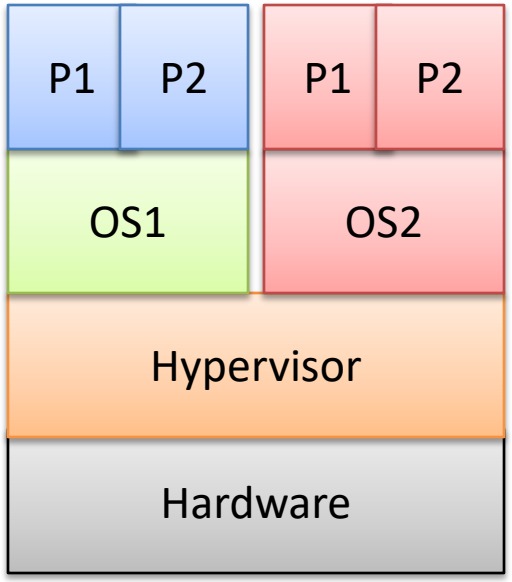




http://handlers.sans.org/tliston/ThwartingVMDetection_Liston_Skoudis.pdf

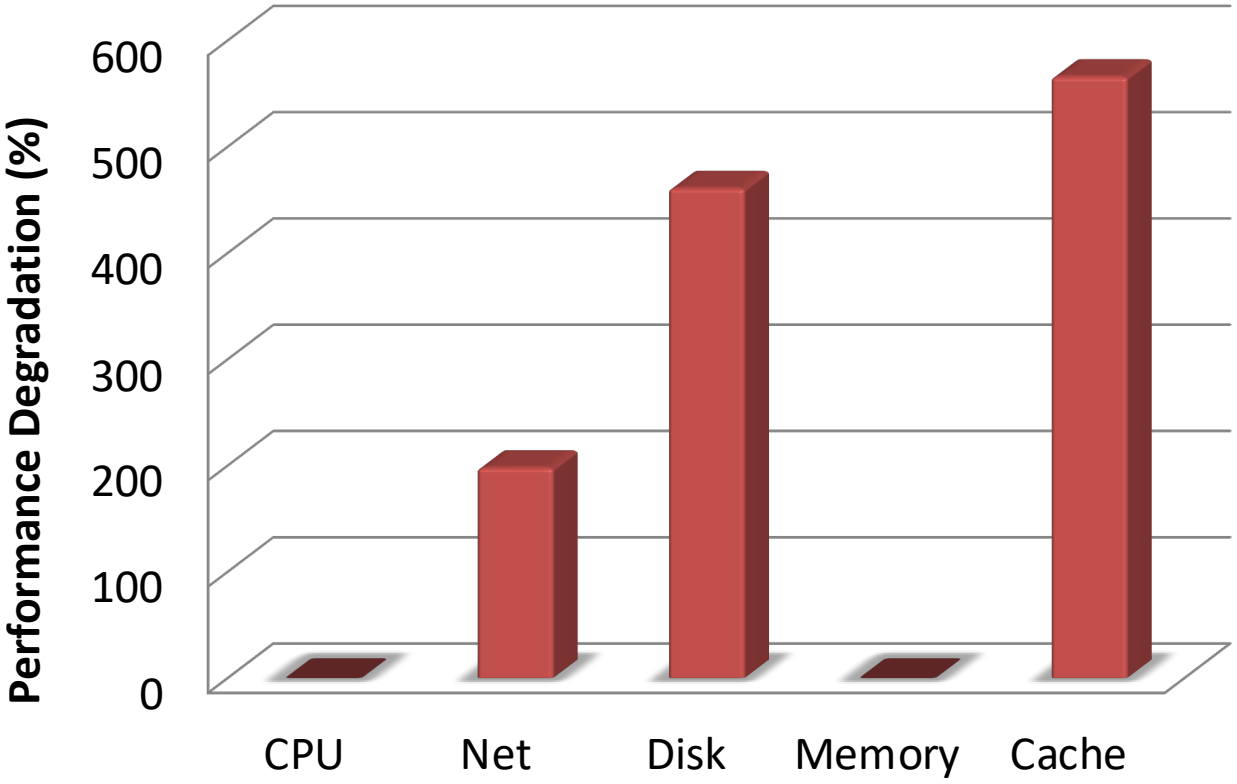
Violating isolation

- Covert channels between VMs circumvent access controls
  - Bugs in VMM
  - Side-effects of resource usage
- Degradation-of-Service attacks
  - Guests might maliciously contend for resources
  - Xen scheduler vulnerability (https://arxiv.org/abs/1103.0759)



Measuring Resource Contention

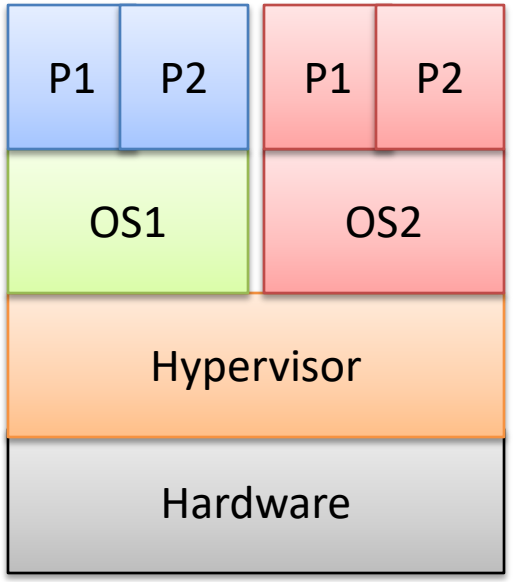
- Contention for the same resource



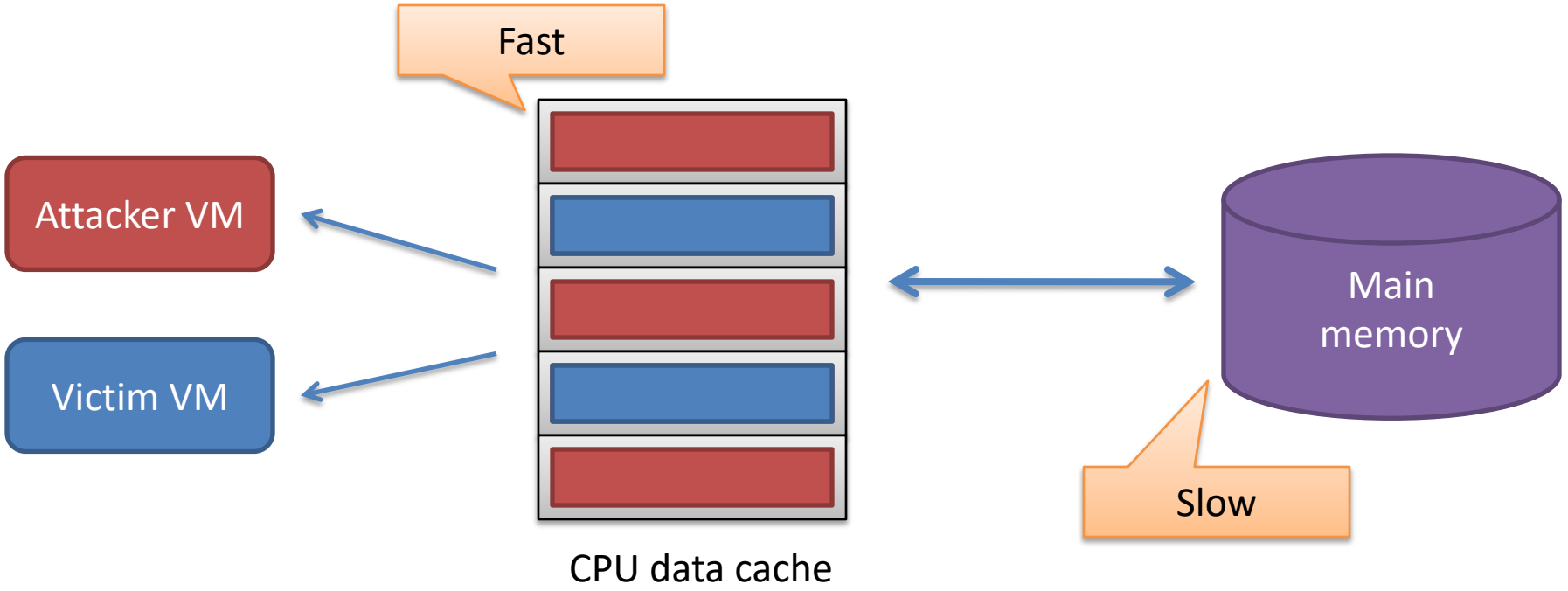
Local Xen Testbed	
Machine	Intel Xeon E5430, 2.66 Ghz
Packages	2, 2 cores per package
LLC Size	6MB per package

Violating isolation

- Covert channels between VMs circumvent access controls
  - Bugs in VMM
  - Side-effects of resource usage
- Degradation-of-Service attacks
  - Guests might maliciously contend for resources
  - Xen scheduler vulnerability (https://arxiv.org/abs/1103.0759)
- Side channels
  - Spy on other guest via shared resources



Cross-VM side channels using CPU cache contention



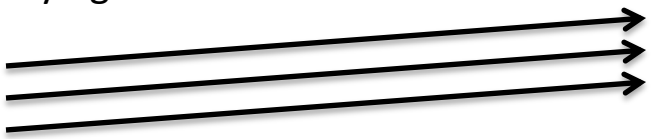
1) Read in a large array (fill CPU cache with attacker data)

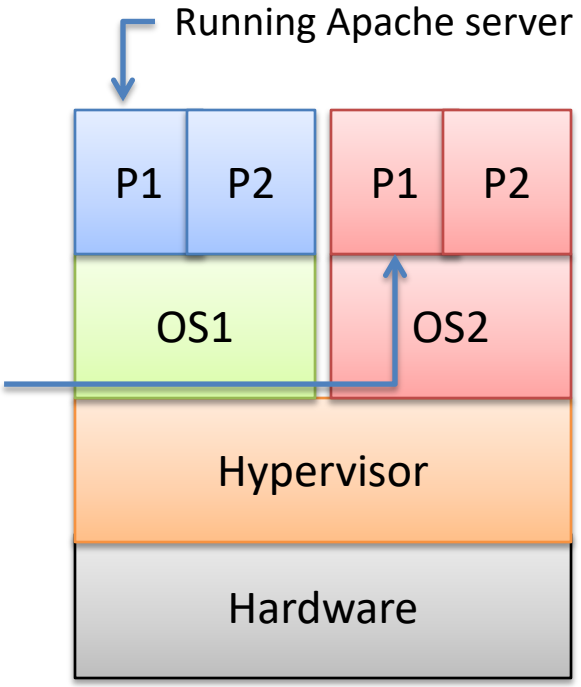
2) Busy loop (allow victim to run)

3) Measure time to read large array (the load measurement)

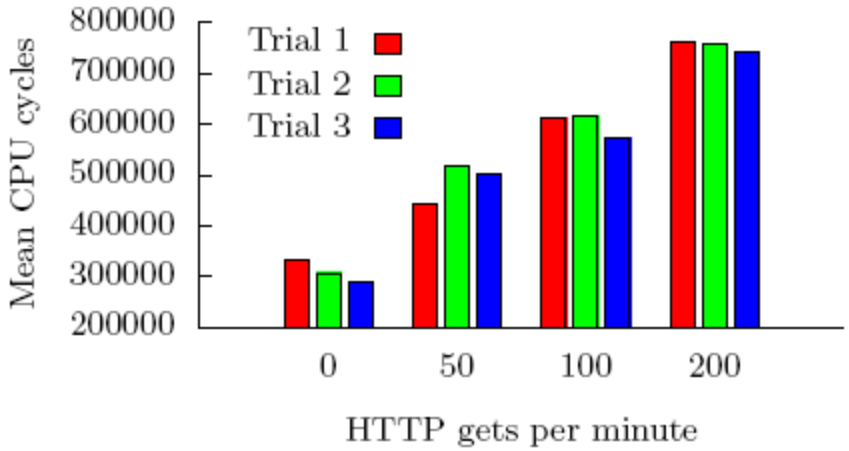
Cache load is correlated with Apache traffic volume

Varying rates of web traffic



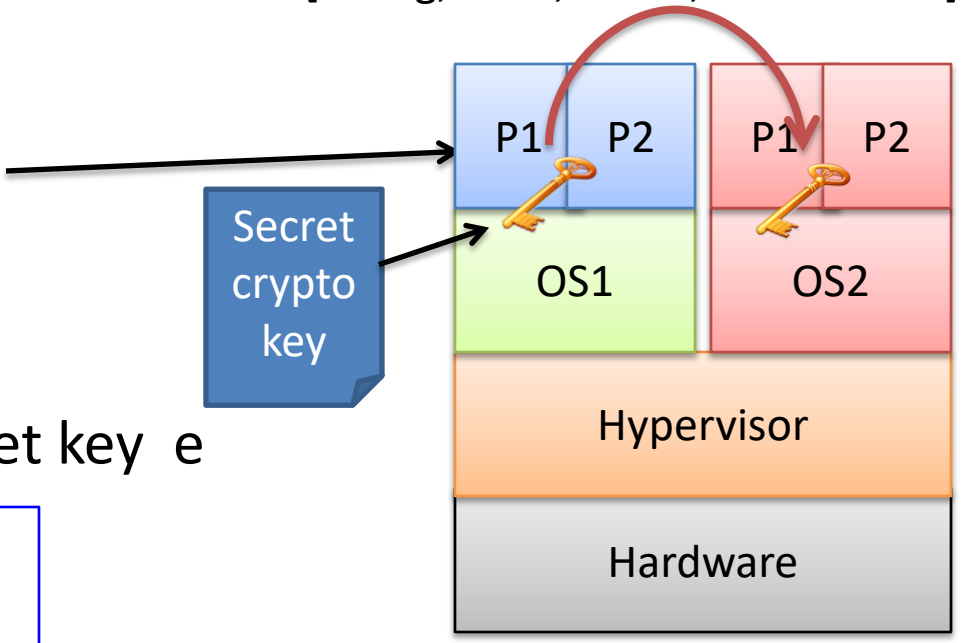


3 trials with 1 pair of EC2 instances:
1,000 cache load measurements during
0, 50, 100, or 200 **HTTP gets** (3 Mbyte page) per minute for ~1.5 mins



Cross-VM cryptographic side-channel attacks

[Zhang, Juels, Reiter, R. – CCS `12]



Target is 4096-bit ElGamal secret key e

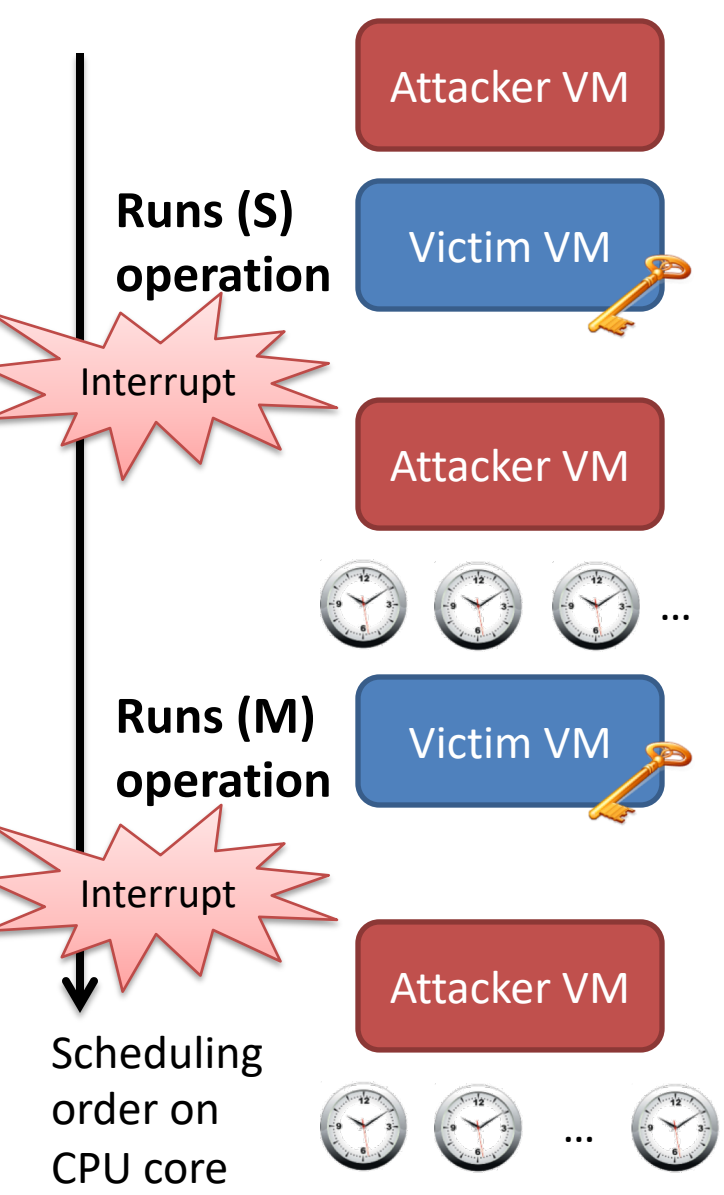
Modular Exponentiation (x, e, N):

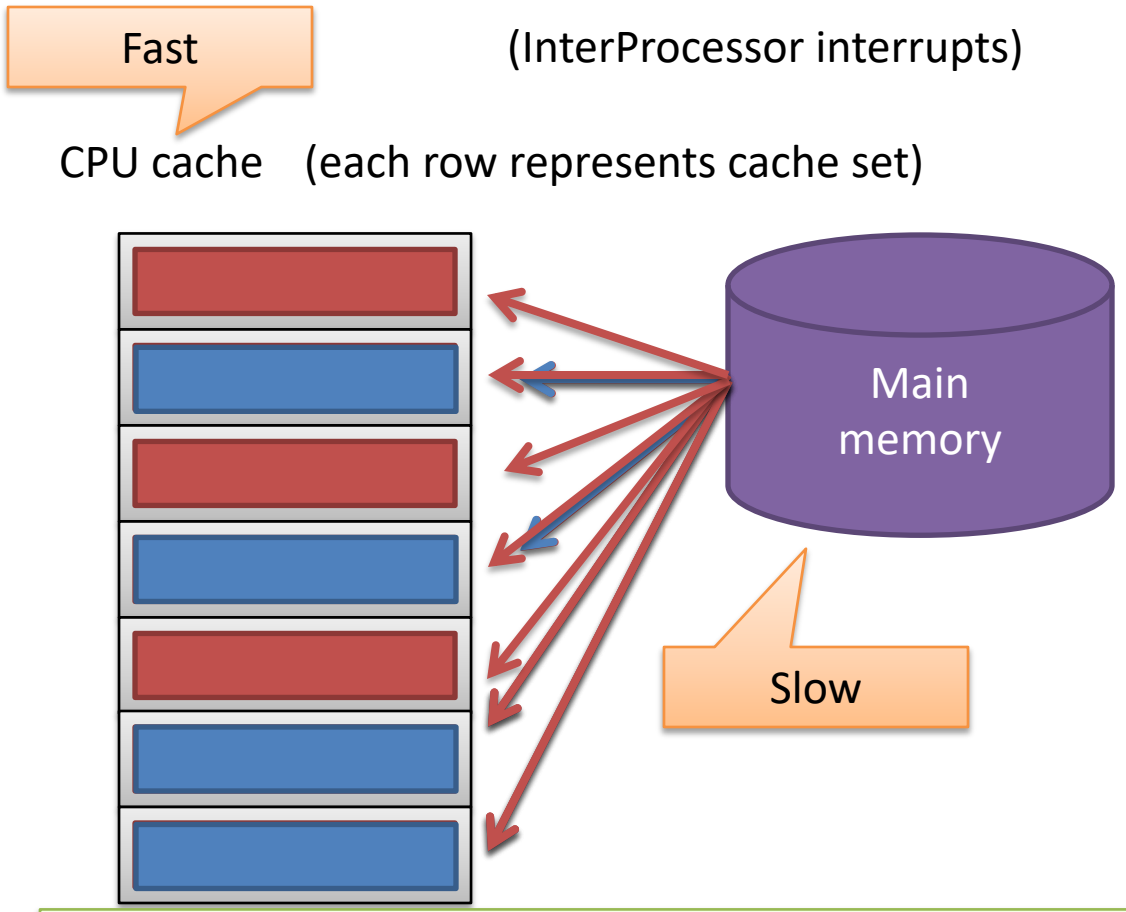
let $e_n \ldots e_1$ be the bits of e
$y \leftarrow 1$
for e_i in $\{e_n \ldots e_1\}$
  $y \leftarrow$ **Square**(y) **(S)**
  $y \leftarrow$ **Reduce**(y, N) **(R)**
  if $e_i = 1$ then
    $y \leftarrow$ **Multi**(y, x) **(M)**
    $y \leftarrow$ **Reduce**(y, N) **(R)**
return y // $y = x^e \bmod N$

$e_i = 1 \rightarrow$ "SRMR"
$e_i = 0 \rightarrow$ "SR"

Sequence of function calls reveals secret key

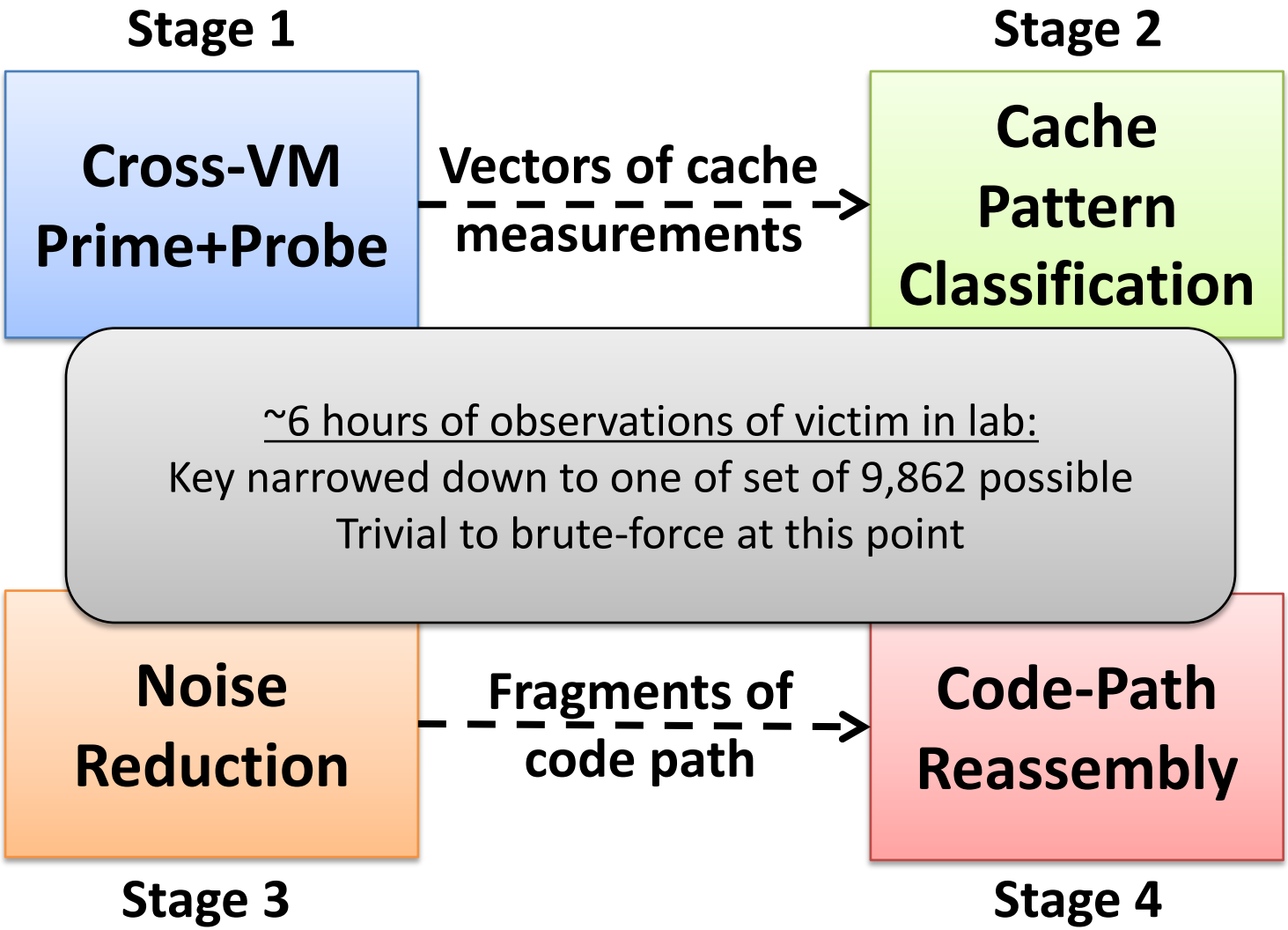
Prime+Probe protocol using IPIs





- Timings correlated to (distinct) cache usage patterns of S, M operations
- Can spy frequently (every ~16 µs) by exploiting scheduling

Full attack pipeline



Cross-VM side channel attacks

- Active research on new attacks and defenses for side-channels
- Most recent attacks can steal secrets quickly
  - Almost exclusively in lab settings
  - One paper demonstrated RSA secret key recovery across VMs on EC2 [Inci et al. 2016]
- Unclear if used by real attackers in the wild

New class of vulnerabilities

- Meltdown/Spectre CPU vulnerabilities discovered 2017, made public early 2018
- Memory access controls bypassed due to how speculative execution works
- Allow reading victim memory

Speculative execution

Modern CPUs allow out-of-order execution of instructions to speed things up

Use branch prediction to attempt to guess outcome of conditional statements

```
(1)     if (x < array1_size)
(2)         y = array2[array1[x] * 4096];
```

Normal execution:

Fetch array1_size from memory, compare to x

If condition holds, execute (2), otherwise do not

Speculative execution

Modern CPUs allow out-of-order execution of instructions to speed things up

Use branch prediction to attempt to guess outcome of conditional statements

```
(1)     if (x < array1_size)
(2)         y = array2[array1[x] * 4096];
```

Speculative execution:

- Say array1_size must be fetched from memory (very slow!)
- Checkpoint state of CPU registers
- If branch prediction guesses that x < array1_size then run line (2) before determining x < array1_size
- Now check if x < array1_size, commit if so, otherwise rollback state

Line (2) gets executed as ***transient instructions*** whether or not condition is true

New class of vulnerabilities

Say this snippet of code in hypercall path

x is controlled by malicious attacker

```
(1)     if (x < array1_size)
(2)         y = array2[array1[x] * 4096];
```

array1[x] secret value (e.g., one byte of a cryptographic secret)

Transient execution fetches
k = array1[x]
and array2[k*4096]

Here k is a secret!

Attacker checks what value of k has
array2[k*4096]
loaded in cache via side-channel

New class of vulnerabilities

Say this snippet of code in hypercall path

x is controlled by malicious attacker

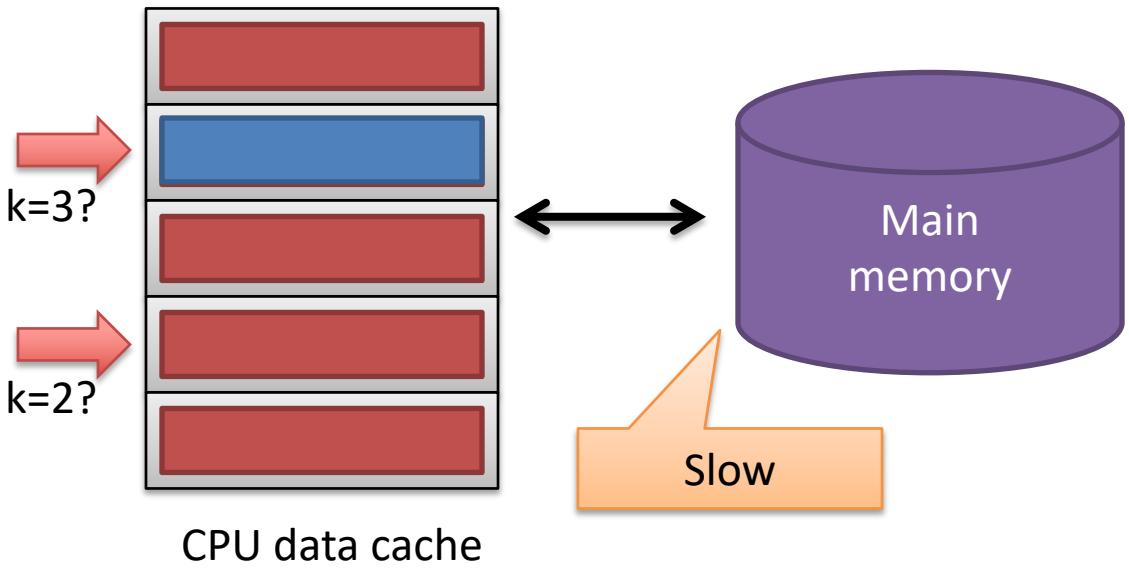
```
(1)     if (x < array1_size)
(2)         y = array2[array1[x] * 4096];
```

array1[x] secret value (e.g., one byte of a cryptographic secret)

Transient execution fetches
k = array1[x]
and array2[k*4096]

Here k is a secret!

Attacker checks what value of k has array2[k*4096] loaded in cache via side-channel



Fallout of Meltdown/Spectre

- Software mitigations for some exploits
- VM vendors released patches
- Cloud providers had to roll out patches quickly
- Intel had to ship new microcode for CPUs
  - Performance degradation serious concern
- 3 class-action lawsuits against Intel

VM Containment & Isolation Lessons

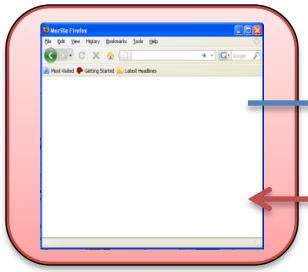
- VMMs provide better containment and isolation than:
  - Processes, containers (e.g., Docker)
- But still not perfect:
  - VMM transparency
  - Vulnerabilities in hypervisor and CPU
  - Side channels exist
- Securing guest OS and host OS still very important for defense-in-depth

Virtual Machine lifecycle management

- Snapshots
  - Volume snapshot / checkpoint
    - persistent storage of VM
    - must boot from storage when resuming snapshot
  - Full snapshot
    - persistent storage and ephemeral storage (memory, register states, caches, etc.)
    - start/resume in between (essentially) arbitrary instructions
- VM image is a file that stores a snapshot

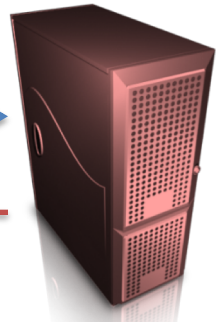
Virtual machines and secure browsing

"**Protect Against Adware and Spyware:** Users protect their PCs against adware, spyware and other malware while browsing the Internet with Firefox in a virtual machine."
[http://www.vmware.com/company/news/releases/player.html]



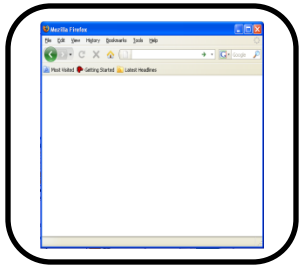
http://www.freesoftware.com/

browser exploit



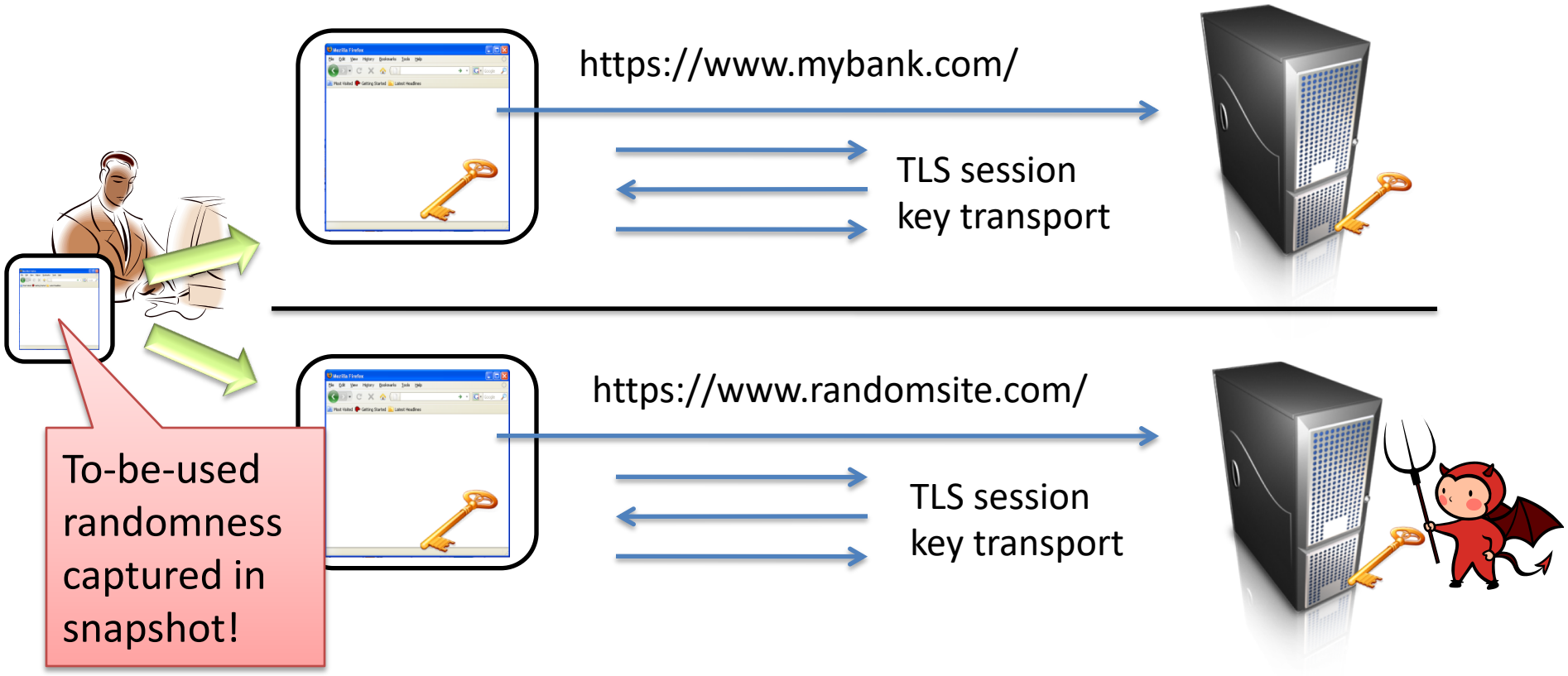
Clean snapshot of VM with browser running

Virtual machine compromised, but not host OS



Resetting to snapshot removes malware

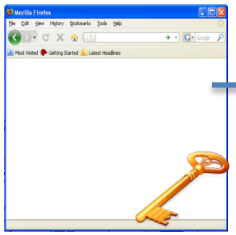
Virtual machine resets lead to RNG failures



2010 vulnerabilities:

Some versions of Firefox, Chrome allow session compromise attacks

Apache mod_ssl TLS server: server's secret key can be stolen!



https://www.mybank.com/

TLS session
key transport

A logical timeline of events

User launches browser in VM

User snapshots VM

Snapshot later run.

User requests https page

Randomness gathered by browser random number generator (RNG)

Randomness used by TLS key transport

TLS client

Public key

RSA encryption

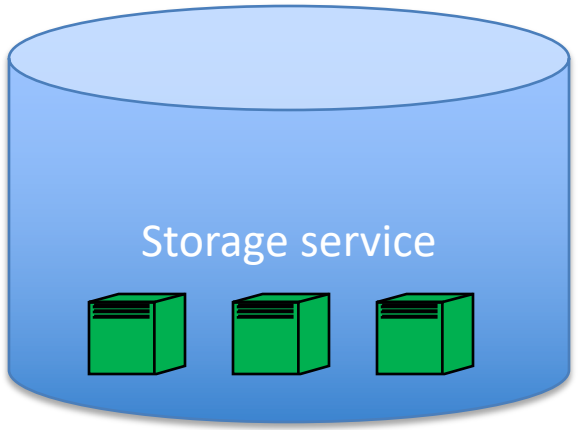
C

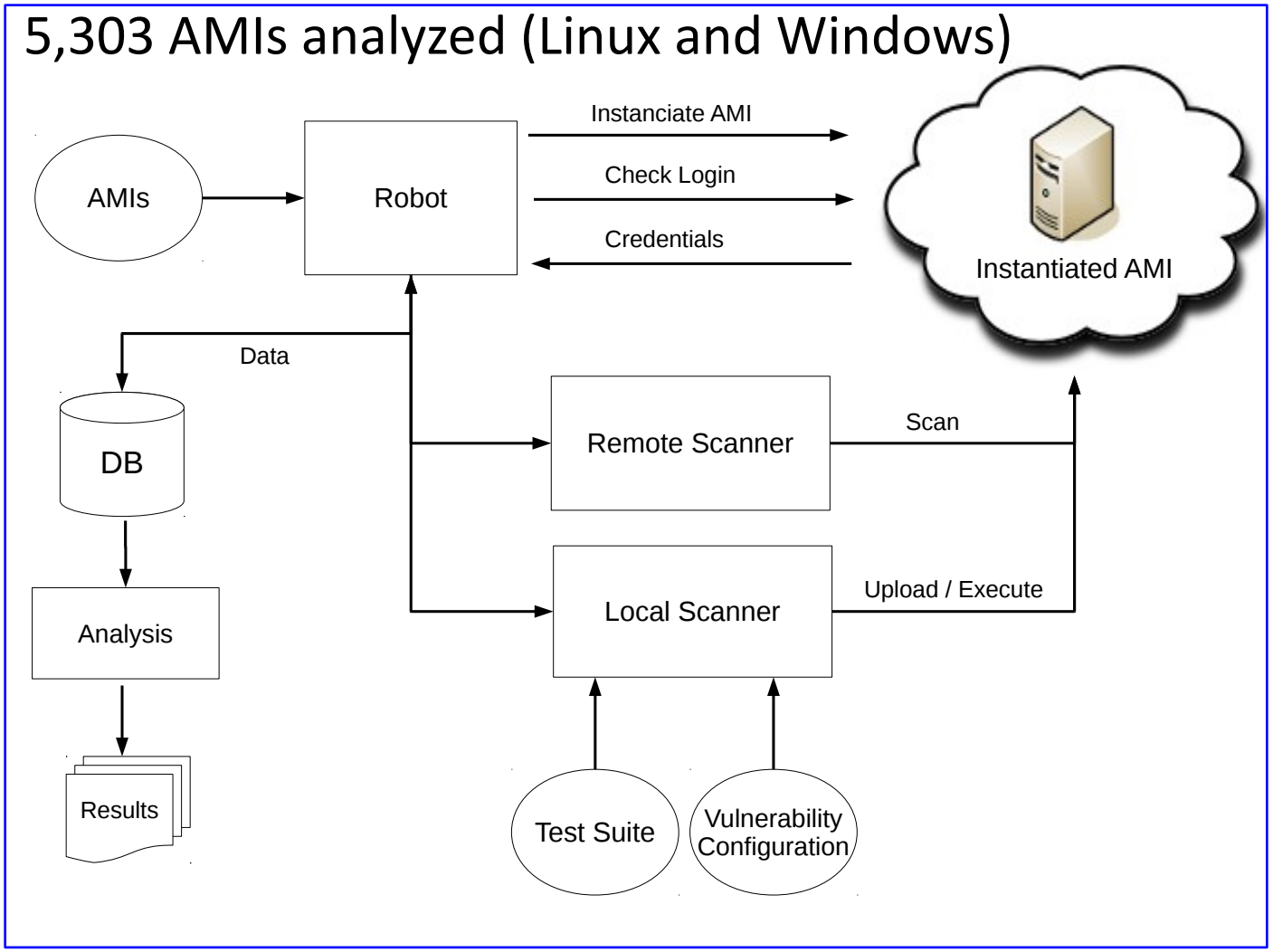
C sent to server

A second run from snapshot leads to same secret key being sent to (different) server

Amazon Machine Images (AMIs)

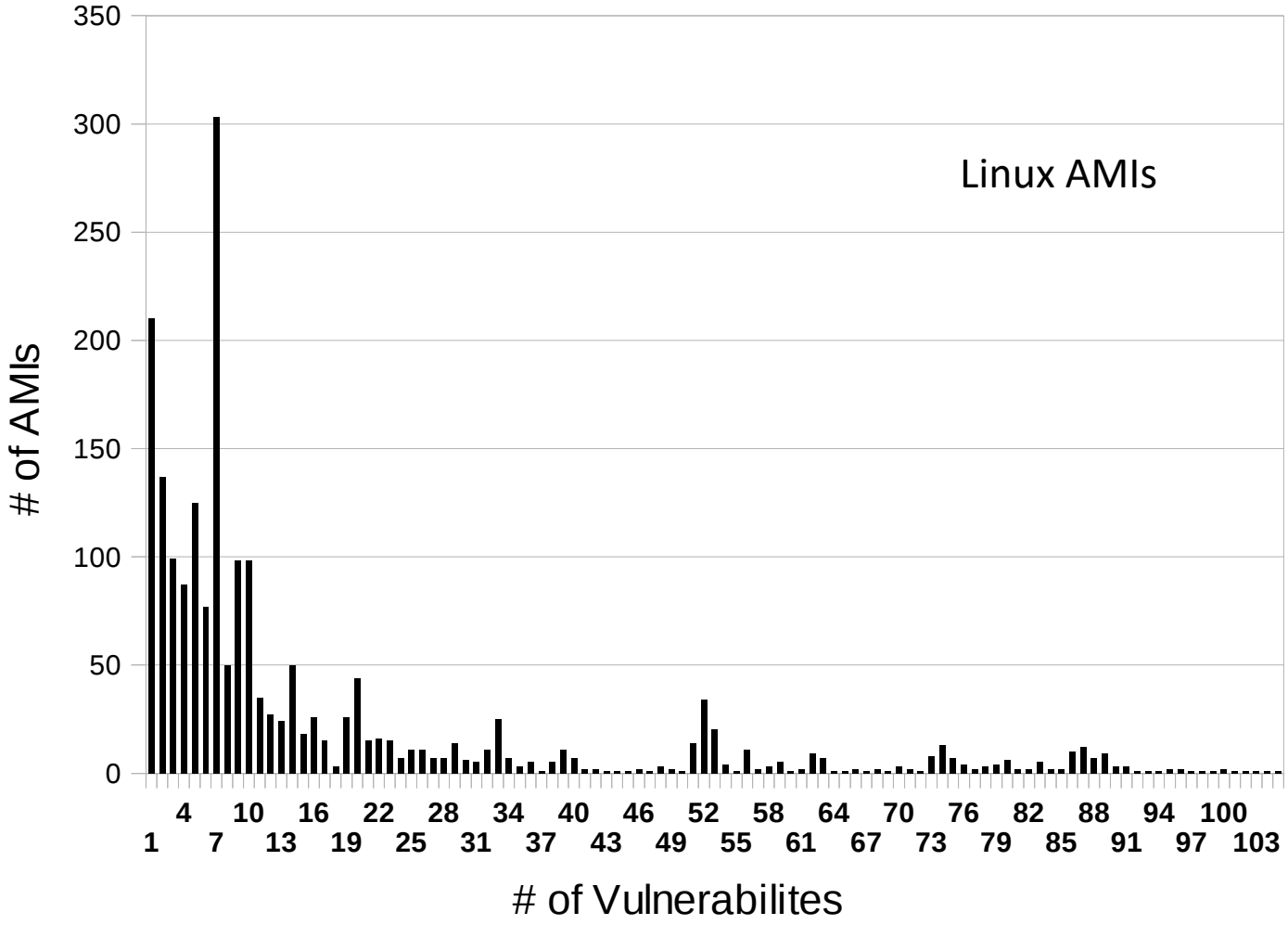
- Users set up volume snapshots / checkpoints that can then be run on the Elastic Compute Cloud (EC2)
- Can be marked as public and anyone can use your AMI





Balduzzi et al. “A Security Analysis of Amazon’s Elastic Compute Cloud Service – Long Version –”, 2011

See also Bugiel et al., “AmazonIA: When Elasticity Snaps Back”, 2011



Also: Malware found on a couple AMIs

Balduzzi et al. analysis

- Backdoors
  - AMIs include SSH public keys within authorized_keys
  - Password-based backdoors

	East	West	EU	Asia	Total
AMIs (%)	34.8	8.4	9.8	6.3	21.8
With Passwd	67	10	22	2	101
With SSH keys	794	53	86	32	965
With Both	71	6	9	4	90
Superuser Priv.	783	57	105	26	971
User Priv.	149	12	12	12	185

Table 2: Left credentials per AMI

Balduzzi et al. analysis

- Credentials for other systems
  - AWS secret keys (to control EC2 services of an account): 67 found
  - Passwords / secret keys for other systems: 56 found

Finding	Total	Image	Remote
Amazon RDS	4	0	4
dDNS	1	0	1
SQL	7	6	1
MySql	58	45	13
WebApp	3	2	1
VNC	1	1	0
Total	74	54	20

Table 3: Credentials in history files

Balduzzi et al. analysis

- Deleted files
  - One AMI creation method does block-level copying

Type	#
Home files (`/home, /root`)	33,011
Images (min. 800x600)	1,085
Microsoft Office documents	336
Amazon AWS certificates and access keys	293
SSH private keys	232
PGP/GPG private keys	151
PDF documents	141
Password file (`/etc/shadow`)	106

Table 5: Recovered data from deleted files

Response

http://www.forbes.com/sites/andygreenberg/2011/11/08/researchers-find-amazon-cloud-servers-teeming-with-backdoors-and-other-peoples-data/

- Amazon notified customers with vulnerable AMIs
- Made private AMIs of non-responsive customers
- New tutorials for bundling systems

Similar problem: git AWS exposures

- Common bad practice to include AWS credentials in source code
  - Gets checked into public git repos
- Criminals and Amazon now regularly scan github and other sites for exposed credentials
- https://github.com/awslabs/git-secrets

VM management lessons

- New software management practices & tools needed for new software deployment models

- Use tools to help prevent credential exposure, use provider's credential management tools (AWS has many)