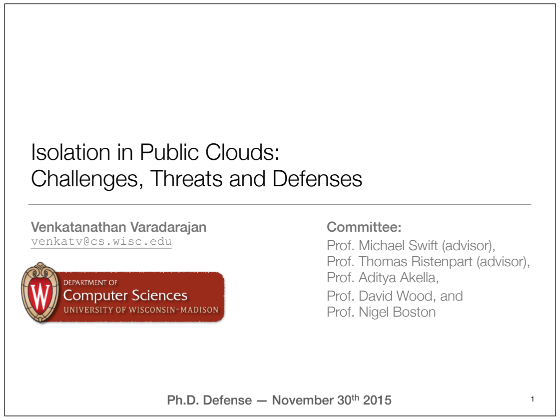
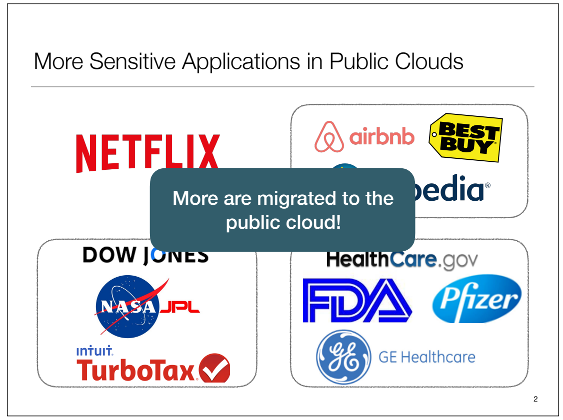


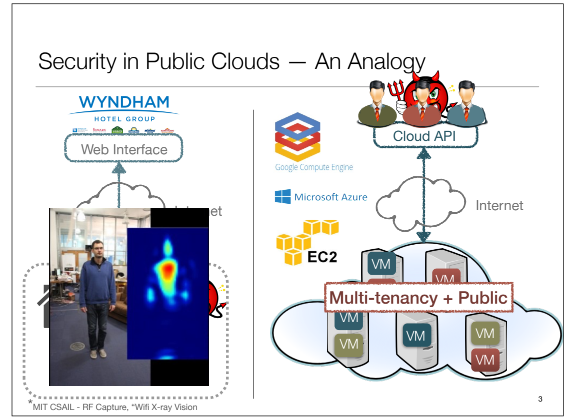
Isolation in Public Clouds:
Challenges, Threats and Defenses
Venkatanathan Varadarajan
venkatv@cs.wisc.edu
Department of Computer Sciences
University of Wisconsin–Madison
Committee:
Prof. Michael Swift (advisor),
Prof. Thomas Ristenpart (advisor),
Prof. Aditya Akella,
Prof. David Wood, and
Prof. Nigel Boston
Ph.D. Defense — November 30th 2015
1

1



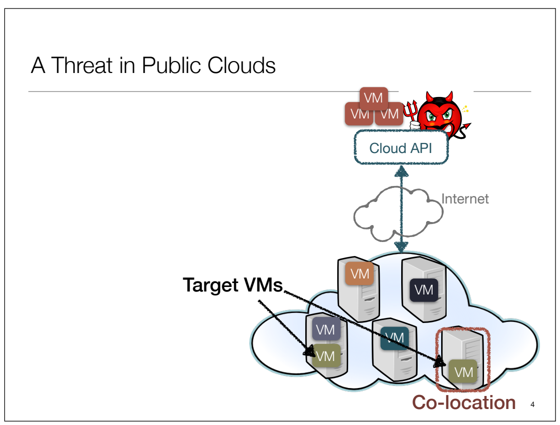
More Sensitive Applications in Public Clouds
NETFLIX
airbnb
BEST BUY
More are migrated to the public cloud!
DOW JONES
NASA JPL
intuit TurboTax
HealthCare.gov
FDA
Pfizer
GE Healthcare
2

2



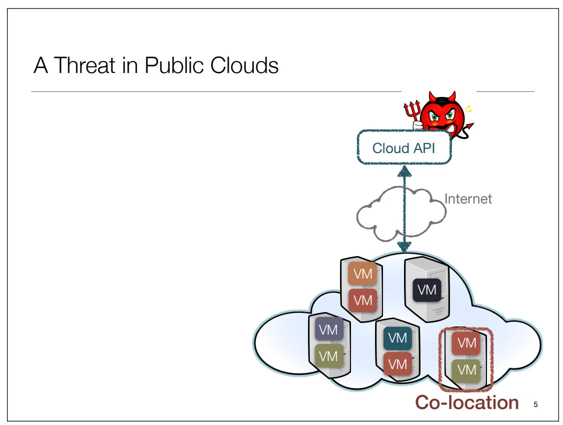
Security in Public Clouds — An Analogy
WYNDHAM
HOTEL GROUP
Web Interface
Cloud API
Google Compute Engine
Microsoft Azure
EC2
Internet
VM
Multi-tenancy + Public
MIT CSAIL - RF Capture, "Wifi X-ray Vision
3

3



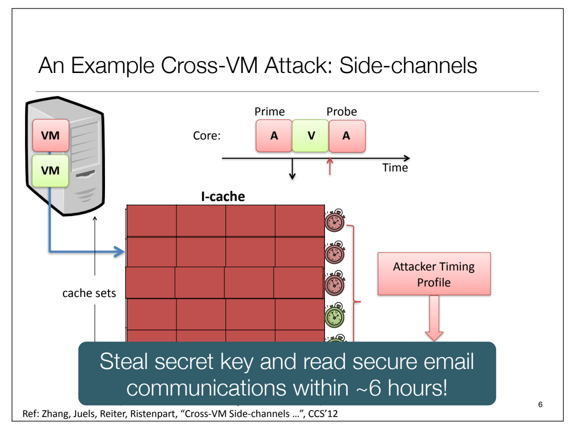
A Threat in Public Clouds
VM
Cloud API
Internet
Target VMs
Co-location
4

4



A Threat in Public Clouds
Cloud API
Internet
VM
Co-location
5

5



An Example Cross-VM Attack: Side-channels
VM
Core:
Prime
Probe
A
V
A
Time
I-cache
cache sets
Attacker Timing Profile
Steal secret key and read secure email communications within ~6 hours!
Ref: Zhang, Juels, Reiter, Ristenpart, "Cross-VM Side-channels ...", CCS'12
6

6

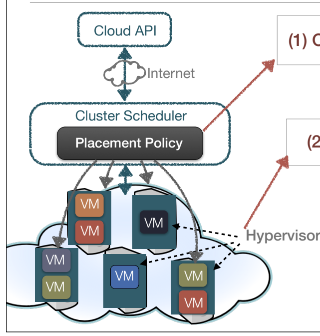
Problem: Cross-VM Attacks

Two steps:

1. *Place* VM on the same host as victim, and
2. *Exploit* sharing and lack of isolation to,
   - Steal *secrets*,
     - e.g., side-channel attacks – L1 D/I cache, TLB, Branch Predictors, LLC, memory …
   - Affect *performance*,
     - e.g., performance degradation, DoS attacks.

Isolation:
"two user tasks are isolated if one cannot know about the execution of the other due to resource sharing."

Placement & Isolation in Public Clouds



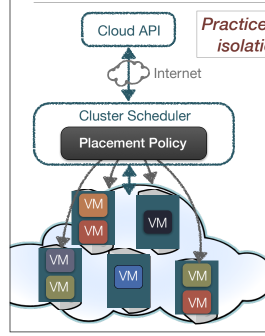
(1) Controls co-location

- **Co-location Attack** [Ristenpart'09, Zhang'14, Varadarajan'15, Xu'15]

(2) Provides isolation

- **Cross-VM side- & covert-channels** [Ristenpart'09, Xu'11, Wu'12, Zhang'12, Yarom'14, Zhang'14, Liu'15, Inci'15]
- **Resource-stealing, DoS** [Zhou'11, Varadarajan'12]

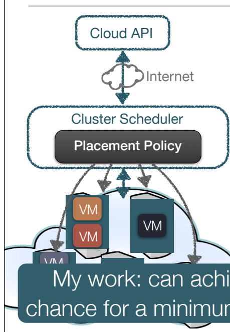
In my dissertation



Practice of multi-tenancy demands stronger isolation between VMs in public clouds.

1. Is co-location practical in modern clouds?
2. Are there unique opportunities for malicious users to exploit the lack of isolation for performance gains?
3. Can isolation be improved without compromising the efficiency?

1. Is Co-location Practical in Modern Clouds?



Let's assume 50,000 servers and a randomized placement policy, With 30 target VMs.

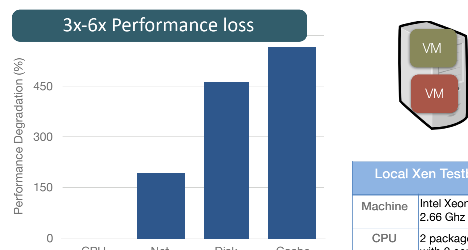
How many VMs do you need to launch to achieve a successful co-location 50% of the time?

~ 1000 to 3000 VMs

Think about its cost ($$ and time)

My work: can achieve co-location with > 90% chance for a minimum of *14* cents with 10-30 VMs!

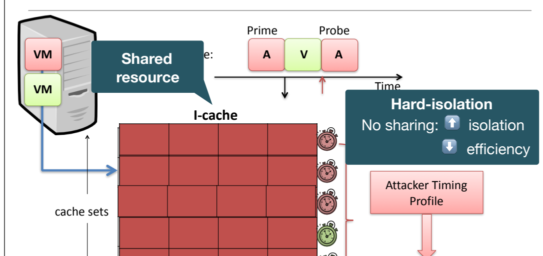
2. Malicious use of Performance Interference



Local Xen Testbed	
Machine	Intel Xeon E5430, 2.66 Ghz
CPU	2 packages each with 2 cores
	…ckage

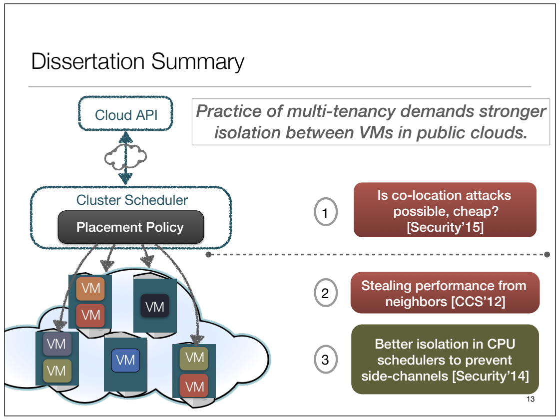
My work: A greedy customer could improve his performance by interfering with neighbors

3. Improving Isolation w/o Compromising Efficiency

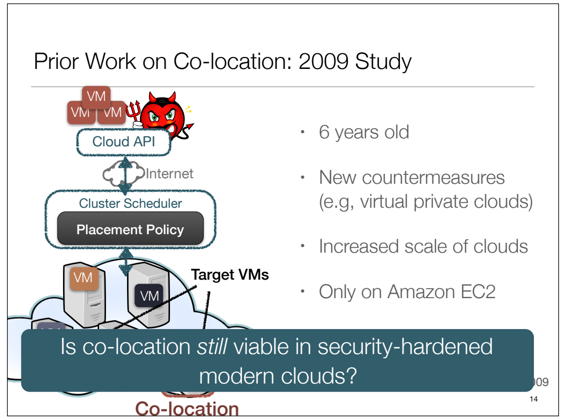


My work: Design principle *soft-isolation* to defend against cross-VM side-channels

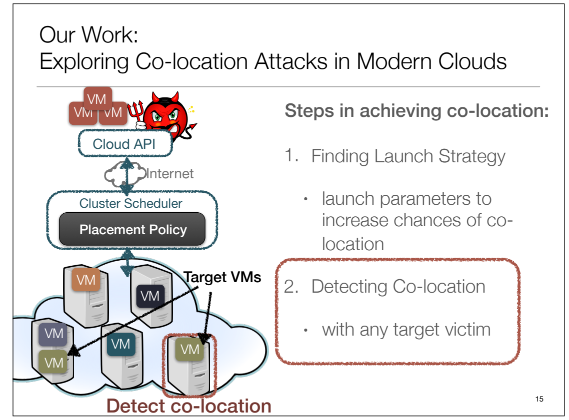
*Zhang, Juels, Reiter, Ristenpart, "Cross-VM Side-channels …", CCS'12



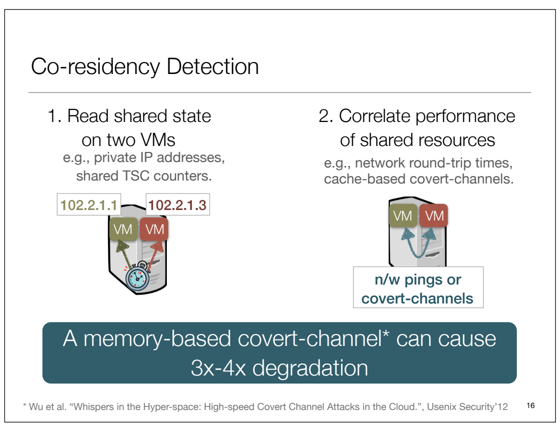
13



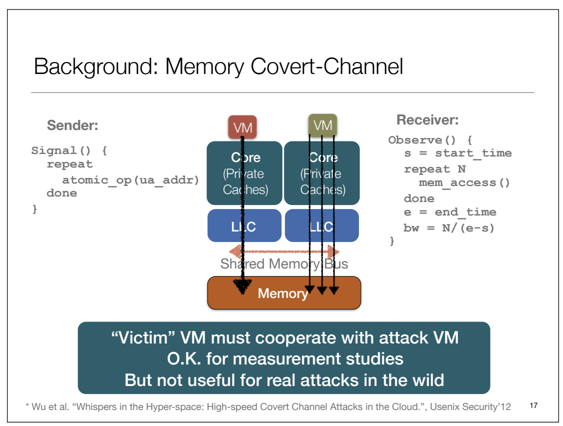
14



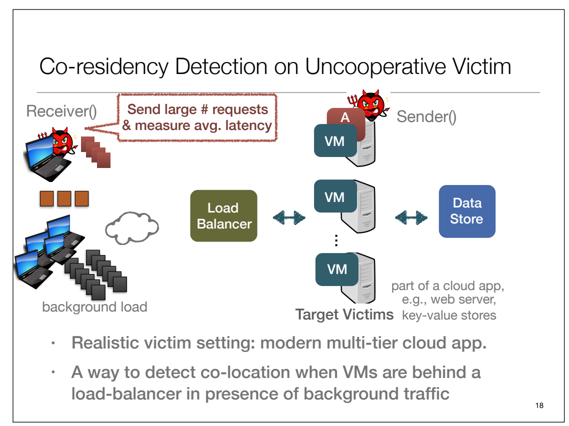
15



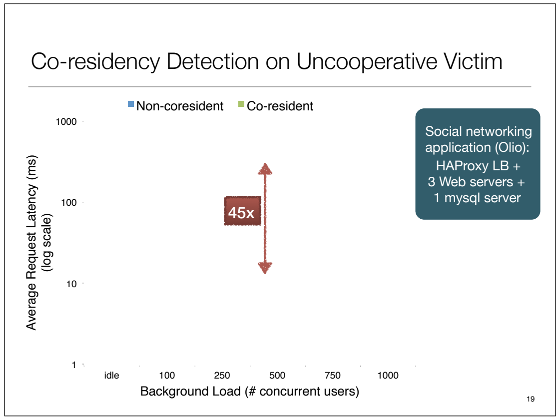
16



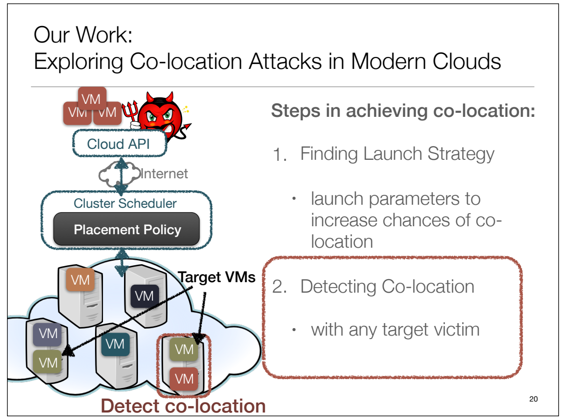
17



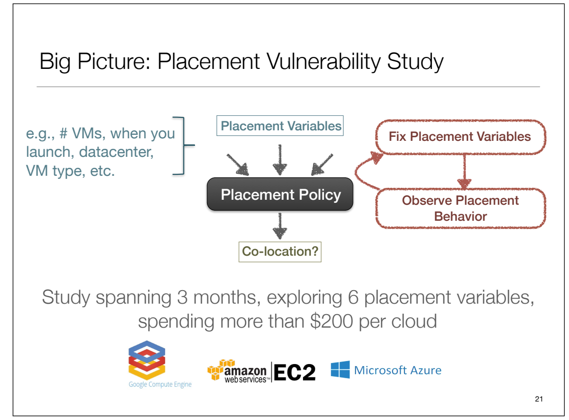
18



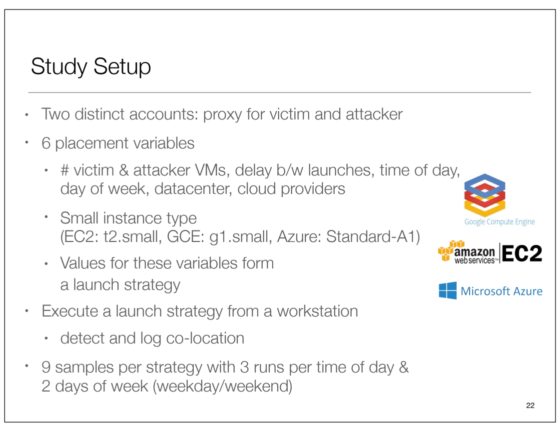
19



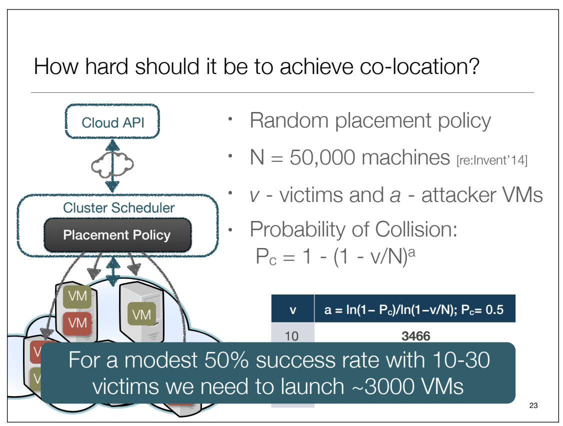
20



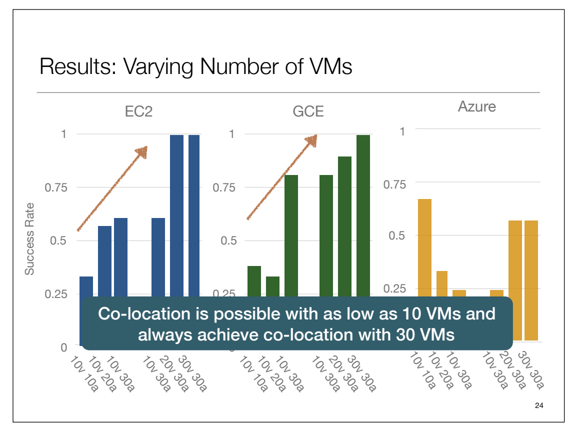
21



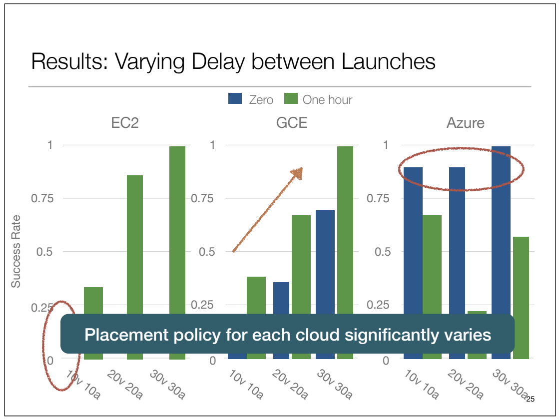
22



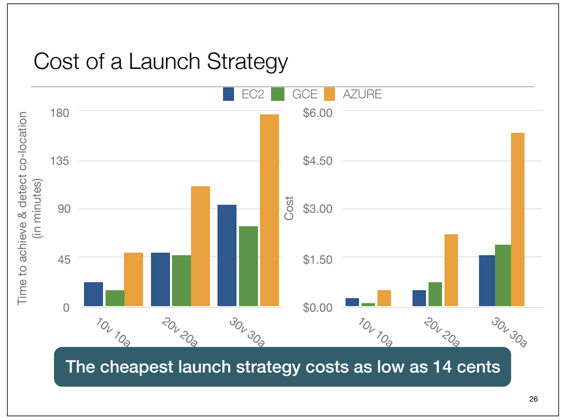
23



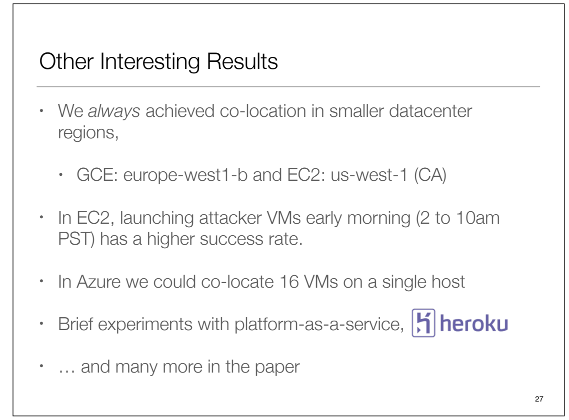
24



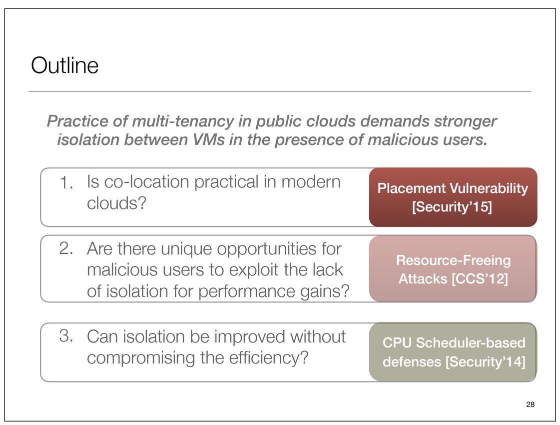
25



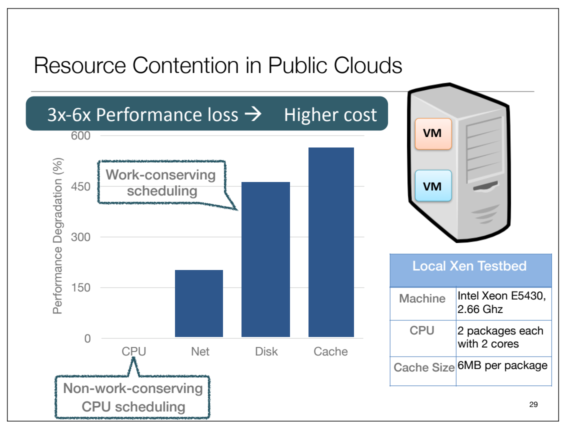
26



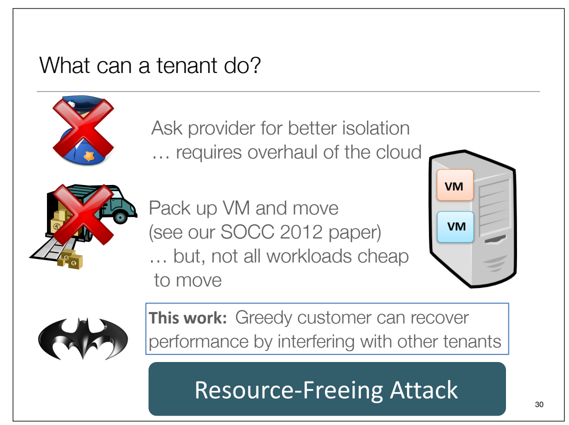
27



28



29



30

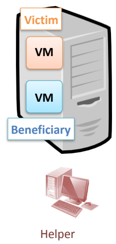
The Setting

Victim:
- One or more VMs
- Public interface (eg, http)

Beneficiary:
- VM whose performance we want to improve

Helper:
- Mounts the attack

Beneficiary and victim fighting over a *target resource*

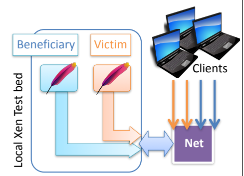


Example: Network Contention

Beneficiary & **Victim**

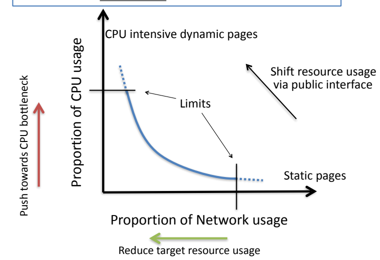
- Apache webservers hosting static and dynamic (CGI) web pages
- **Target Resources:** Network Bandwidth
- Work-conserving scheduler
  - *half* the bandwidth

What can you do?



Recipe for a Successful RFA

Shift resource away from the *target resource* towards the *bottleneck* resource



An RFA in Our Example

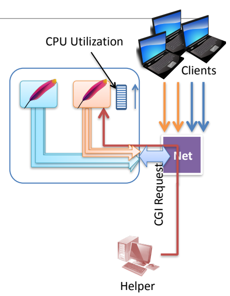
In our testbed:

Increases beneficiary's share of bandwidth

No RFA: 1800 page requests/sec

W/ RFA: 3026 page requests/sec

50% →85% share of bandwidth



Summary: Resource Freeing Attacks

1) Send targeted requests to victim
2) Shift resources use from target to a bottleneck

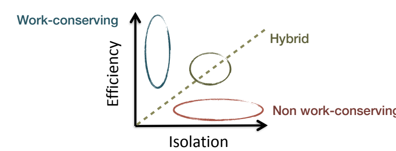
Similar RFA on other resources exists, e.g. CPU Cache Bandwidth

On EC2, we reduced contention by 66.5% = performance improvement of 3-13%

"Resource-Freeing Attacks: Improve Your Cloud Performance (at Your Neighbor's Expense)", ACM CCS 2012

What did we learn from RFAs?

1. Work-conserving vs. Non-work-conserving



2. Effects of simple pay-per-hour pricing model
   - $ per unit time != $ per useful work-done

Outline

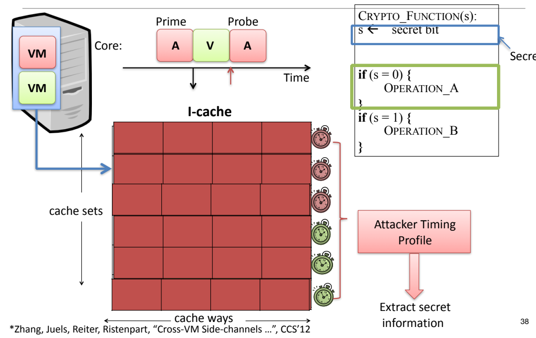
Practice of multi-tenancy in public clouds demands stronger isolation between VMs in the presence of malicious users.

1. Is co-location practical in modern clouds? — Placement Vulnerability [Security'15]
2. Are there unique opportunities for malicious users to exploit the lack of isolation for monetary gains? — Resource-Freeing Attacks [CCS'12]
3. Can isolation be improved without compromising the efficiency? — CPU Scheduler-based defenses [Security'14]

37

37

Problem: Cache-based Side-channels*

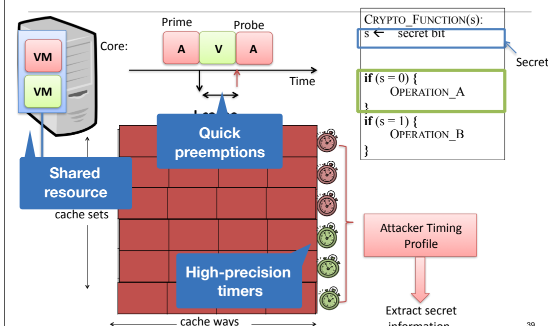


*Zhang, Juels, Reiter, Ristenpart, "Cross-VM Side-channels ...", CCS'12

38

38

Requirements for Successful Side-channel



*Zhang, Juels, Reiter, Ristenpart, "Cross-VM Side-channels ...", CCS'12

39

39

Defenses against Side-channels

1. **Sharing**
   - Resource Partitioning [NoHype'10]
   - Specialized Hardware [RPcache'07]
   - Software-based partitioning [StealthMem'12]
2. **Access to high-resolution timers**
   - Reduce resolution [TimeWarp'12]
   - Removing timing channel [StopWatch'13]

No countermeasures deployed by providers!

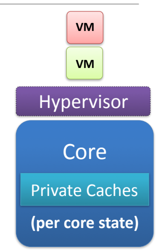
3. **Quick cross-VM preemptions**
   - No prior work!

40

40

Our Solution: Soft Isolation

Allow sharing but limit frequency of dangerous cross-VM interactions



Goals:

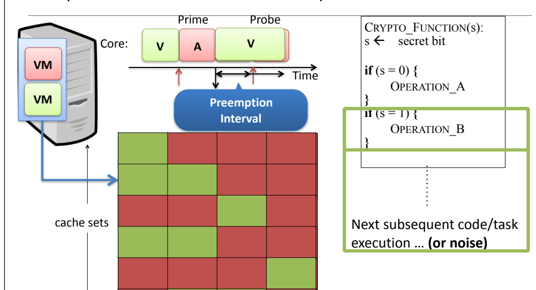
1. *Secure*: Controlled information leakage
2. *Commodity*: Easy to adopt
3. *Efficient*: Allow sharing, low overhead

... with simple changes to Hypervisor's CPU scheduler

41

41

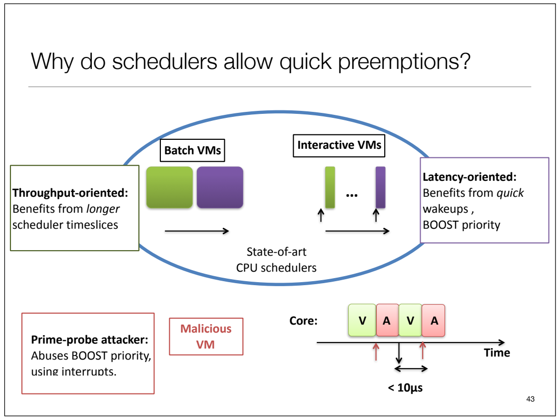
Requirement for Quick Preemptions



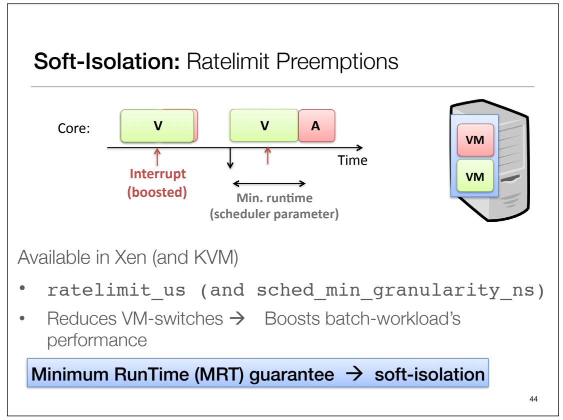
Rate of preemption > Rate of event to measure

42

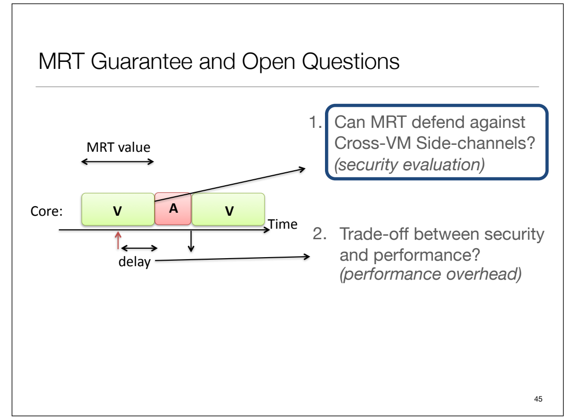
42



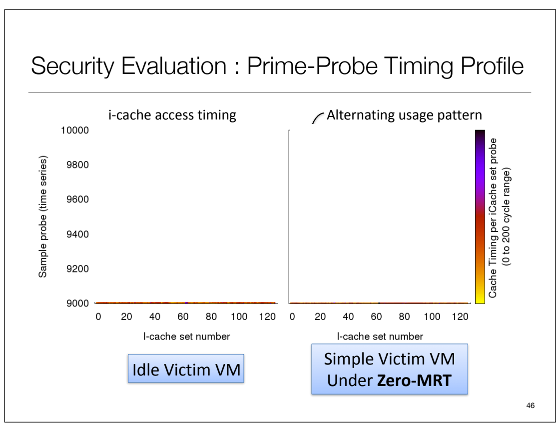
43



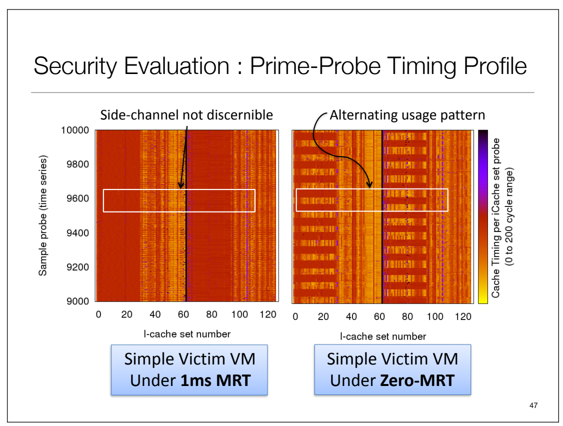
44



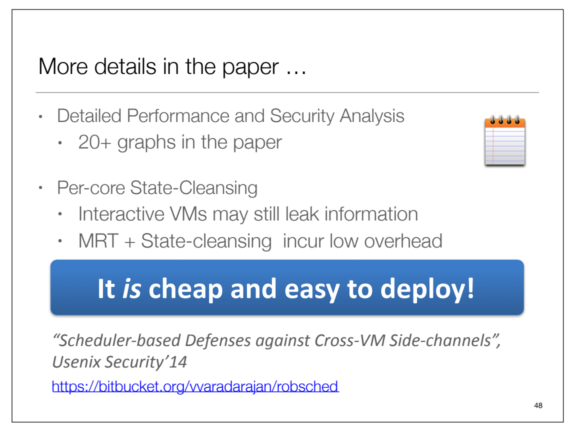
45



46



47



48

Conclusion

1. Is co-location practical in modern clouds? ***Yes, and surprisingly cheap!*** — Placement Vulnerability [Security'15]
2. Exploiting the lack of isolation for performance gains? ***Yes, reduce contention by gaming schedulers.*** — Resource-Freeing Attacks [CCS'12]

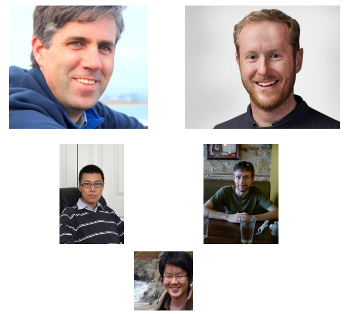
Practice of multi-tenancy in public clouds demands stronger isolation between VMs in the presence of malicious users.

3. Improving isolation w/o compromising efficiency? ***Soft-Isolation*** — CPU Scheduler-based defenses [Security'14]

49

49

Thanks to …



50

50

References

(1) **Varadarajan, V.**, Kooburat, T., Farley, B., Ristenpart, T., & Swift, M. M. *Resource-freeing attacks: improve your cloud performance (at your neighbor's expense).* ACM CCS 2012

(2) **Varadarajan, V.**, Ristenpart, T., & Swift, M. *Scheduler-based defenses against cross-vm side-channels.* In Usenix Security 2014.

(3) **Varadarajan, V.**, Zhang, Y., Ristenpart, T., & Swift, M. *A placement vulnerability study in multi-tenant public clouds.* In Usenix Security 2015.

(4) Farley, B., Juels, A., **Varadarajan, V.**, Ristenpart, T., Bowers, K. D., & Swift, M. M. *More for your money: exploiting performance heterogeneity in public clouds.* ACM SoCC 2012.

(5) Volos, H., Nalli, S., Panneerselvam, S., **Varadarajan, V.**, Saxena, P., & Swift, M. M.. *Aerie: Flexible file-system interfaces to storage-class memory.* ACM Eurosys 2014.

51

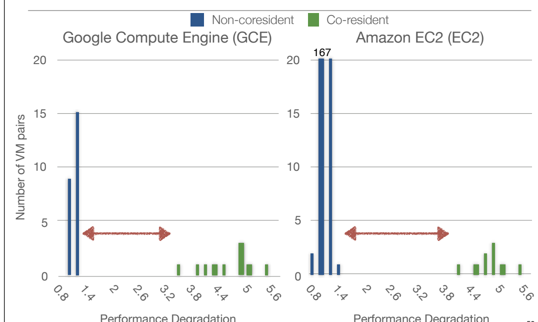
51

Backup Slides

52

52

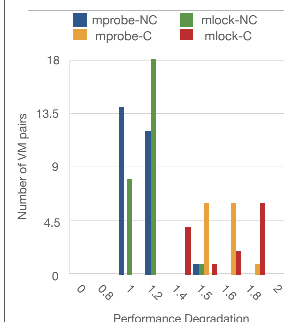
Results: Co-residency Detection in Public Clouds



53

53

Co-residency Detection in Public Clouds (contd.)



- Heterogeneity at play on **Azure AMD machines**
- Requiring two receivers to detect co-residency
- Memory-probing *or* memory-locking: *> 1.5x degradation*

54

54

Limitations

1. Although an *exhaustive* study
   – results limited to small instance type, three clouds, 9 runs per configuration, two user accounts etc.
2. Positive co-residency signal != *exploitable*
   – may share only a small set of resources (e.g., memory, network)
3. A result of an *unlucky* placement policy settings?

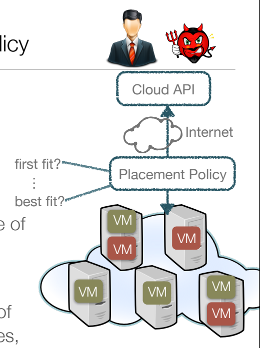
“A Placement Vulnerability Study in Multi-tenant Public Clouds”, Usenix Security 2015

55

55

Challenge 1: Placement Policy

- An *unknown* placement policy places VM on a host
- Variables may influence VM placement
  – Parameters: type of VM, time of launch, # VMs launched, datacenter region, etc.
  – Environment variables: time of day, day of week, # machines, etc.

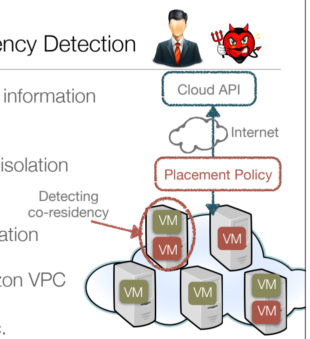


56

56

Challenge 2: Co-residency Detection

- No explicit co-residency information from cloud provider
- Multiple layers of strong isolation
  – Virtual machine — e.g. hardware virtualization
  – Network — e.g. Amazon VPC
  – Modern hardware, etc.



57

57

Some Strategies Work Better than Others

Example strategies on EC2

Launch Strategy	v x a	Cost in Cloud	Cost under Random Placement*	Success rate norm. w/ random*
Launch 10 VMs in less popular datacenter	10x10	\$0.26	\$113.87	1/0.1 (=10)
Launch 30 VMs 1 hour after victim VM launches	30x30	\$1.56	\$32.75	1/0.6 (=1.67)
Launch more than 20 VMs 4 hours after victim VM launches	20x20	\$0.52	\$53.76	1/0.33 (=3.03)

*Random Placement of VMs on N hosts,
v x a launch strategy has a probability of collision: $1 - (1 - v/N)^a$

58

58

Resource Freeing Attacks

1) Send targeted requests to victim

2) Shift resources use from target to a bottleneck

Can we mount RFAs when target resource is CPU cache?

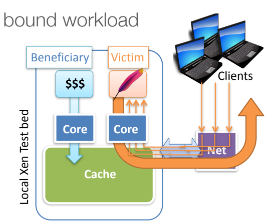
Shared CPU Cache:
- *Ubiquitous*: Almost all workloads need cache
- *Hardware controlled*: Not easily isolated via software
- *Performance Sensitive:* High performance cost!

59

59

Case Study: Cache vs. Network

- **Victim** : Apache webserver hosting static and dynamic (CGI) web pages
- **Beneficiary**: Synthetic cache bound workload (*LLCProbe*)
- **Target Resource:** Cache
- No cache isolation:
  ~3x slower when sharing cache with webserver



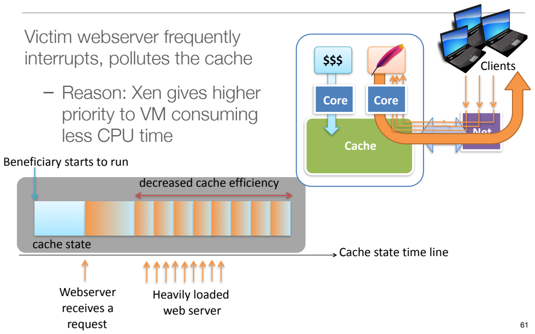
60

60

Cache vs. Network

Victim webserver frequently interrupts, pollutes the cache

– Reason: Xen gives higher priority to VM consuming less CPU time

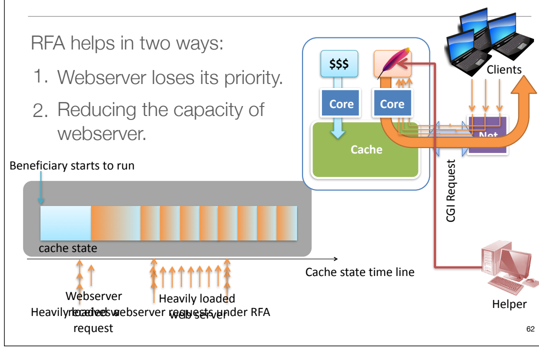


61

Cache vs. Network w/ RFA

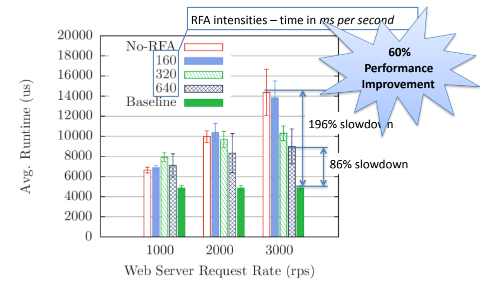
RFA helps in two ways:

1. Webserver loses its priority.
2. Reducing the capacity of webserver.



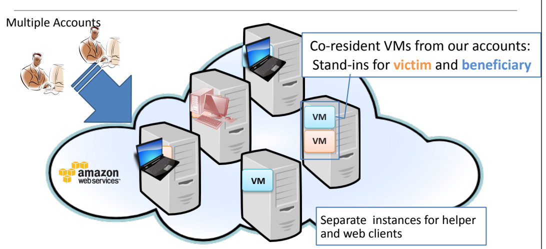
62

RFA: Performance Improvement



63

Experiments on Amazon EC2

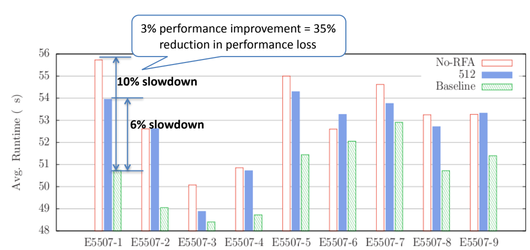


Instance type	m1.small
# of co-resident pairs	***9 (23 total instances)***
Machine type	Intel Xeon E5507 with 4MB LLC

No ***direct*** interact with any other customers
Indirect interaction akin to normal usage cases

64

mcf from SPEC-CPU



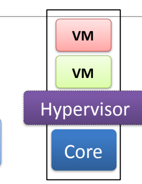
On average RFA improved performance across *all* SPEC workloads!

65

Experimental Methodology

Two VMs:

1. Attacker
2. Victim



Setting similar to public clouds (e.g. EC2)

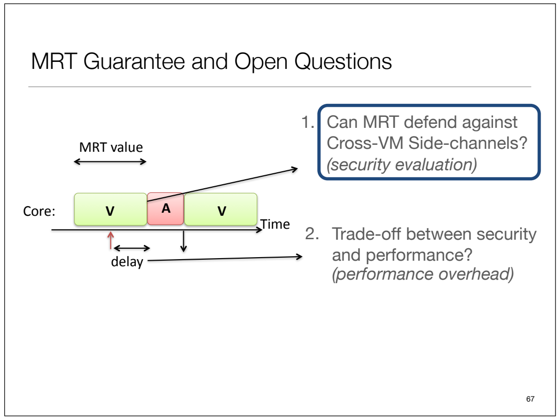
Xen Configuration

Xen Version	4.2.1
Scheduler	*Credit Scheduler 1*
Configuration (Non-work conserving)	*40% cap on DomU VCPUs with equal weight*
# VMs	*6*
# VCPUs per VM	*2*

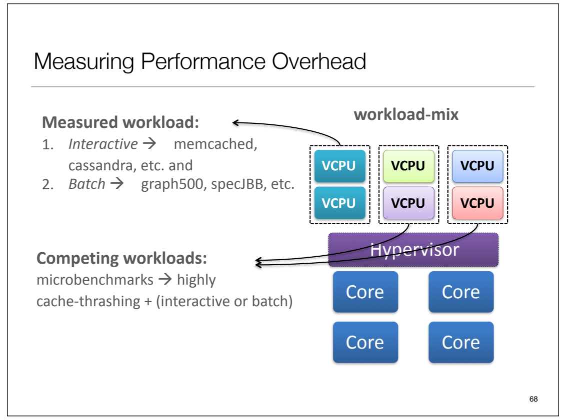
Machine Configuration

Machine	Intel Xeon E5645, 2.4GHz, 6 cores, single package
Memory Hierarchy	*Private 32KB L1 (I- and D-Cache), 256KB unified L2, 12MB shared L3 & 16GB DDR3 RAM.*

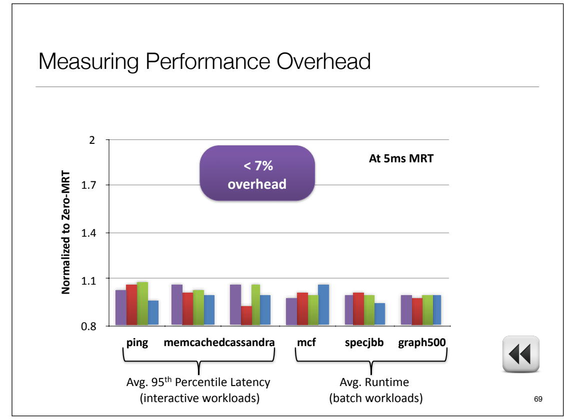
66


MRT Guarantee and Open Questions
MRT value
Core:
V
A
V
Time
delay
1. Can MRT defend against Cross-VM Side-channels? (security evaluation)
2. Trade-off between security and performance? (performance overhead)
67

67


Measuring Performance Overhead
Measured workload:
1. Interactive → memcached, cassandra, etc. and
2. Batch → graph500, specJBB, etc.
Competing workloads:
microbenchmarks → highly cache-thrashing + (interactive or batch)
workload-mix
VCPU
VCPU
VCPU
VCPU
VCPU
VCPU
Hypervisor
Core
Core
Core
Core
68

68


Measuring Performance Overhead
Normalized to Zero-MRT
2
1.7
1.4
1.1
0.8
< 7% overhead
At 5ms MRT
ping
memcached
cassandra
mcf
specjbb
graph500
Avg. 95th Percentile Latency (interactive workloads)
Avg. Runtime (batch workloads)
69

69