Resource-Freeing Attacks: Improve Your Cloud Performance (at Your Neighbor's Expense)

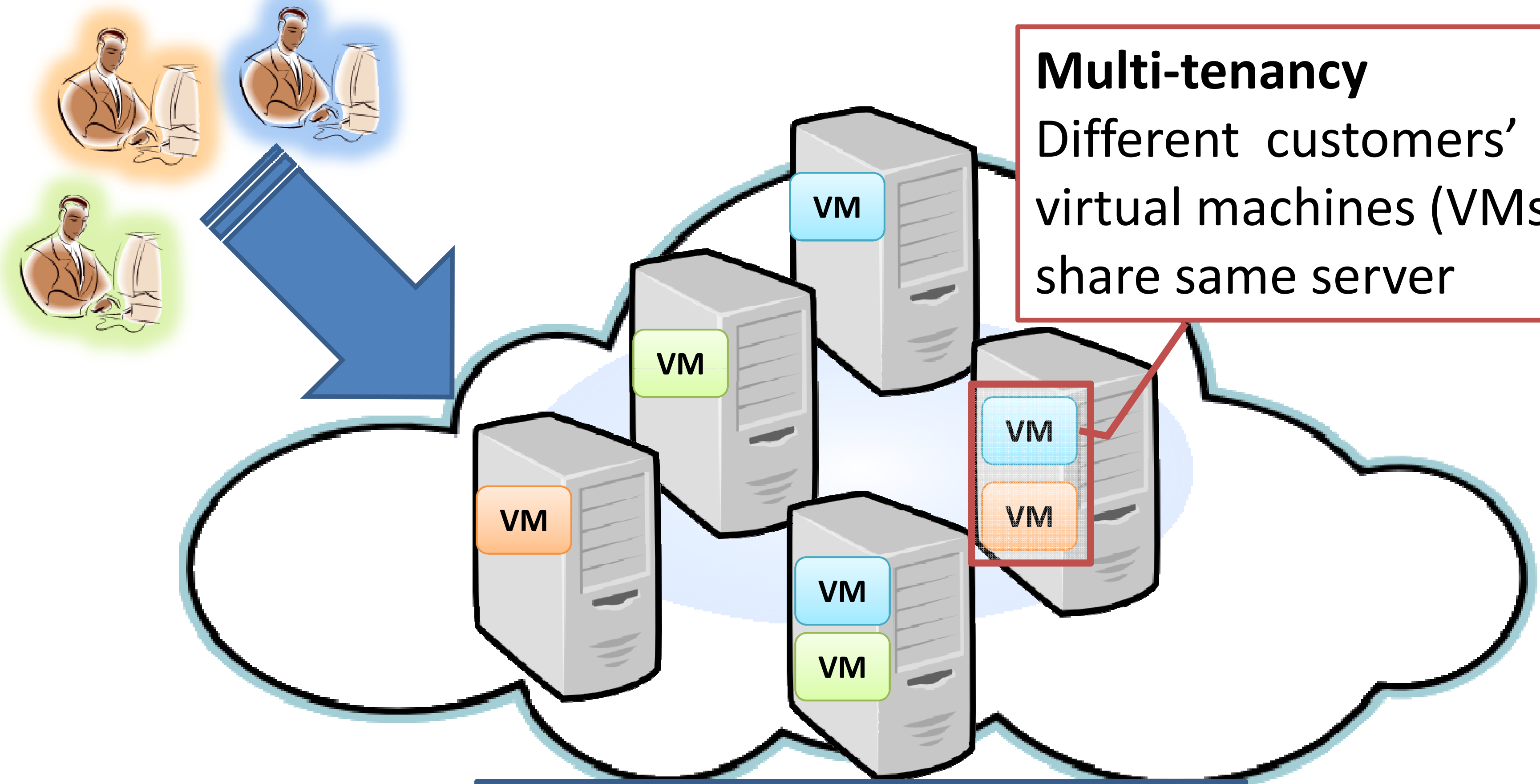
Venkata Varadarajan, Thawan Kooburat, Benjamin Farley,
Thomas Ristenpart and Michael M. Swift

In Conference on Computer and Communications Security (CCS 2012)

Presented by

Amiya Maji

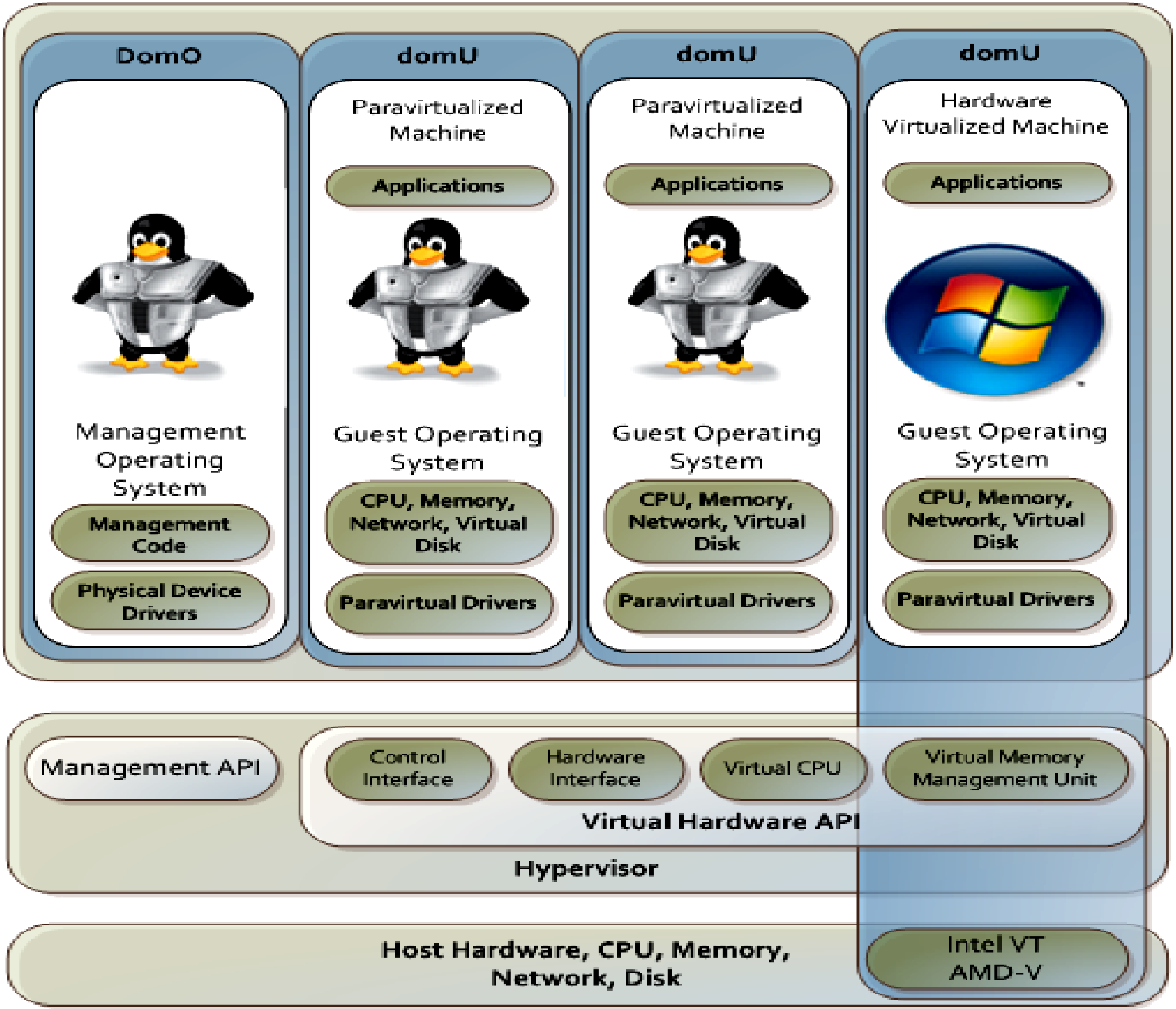
Public Clouds (EC2, Azure, Rackspace, …)



Multi-tenancy
Different customers' virtual machines (VMs) share same server

Why multi-tenancy?
Improved resource utilization

† Some slides are taken from authors' CCS'12 presentation



http://docs.oracle.com/cd/E11081_01/doc/doc.21/e10898/ovmserver.htm

Advantages and Disadvantages of Cloud

- Advantages
  - Better resource utilization
  - Low cost
  - On-demand provisioning
  - Scalability
- Disadvantages
  - Interference due to multi-tenancy
  - Side-channel attacks

Performance Isolation in Cloud

- Utilization and isolation are competing objectives
- Goal of hypervisor
  - Isolation and fair sharing of
    - CPU, Memory, Network, Disk, Cache, Mem Bandwidth
- Perfect isolation not practical for cache, memory bandwidth

What Happens due to Imperfect Isolation?

- Variable application performance
- How much interference?
  - Upto 7x increase in runtime for cache intensive workload
- And new security threats
  - **Resource-Freeing Attacks**

What is Resource-Freeing Attack?

- Modify workload of a victim in such a manner that frees up resources for attacker (also the beneficiary)
- Objective: Recover performance loss of beneficiary due to interference of victim
- Example
  - Victim: Web Server A (WS-A)
  - Beneficiary: Web Server B (WS-B)
  - Contending resource: Network bandwidth
  - How: Send CPU intensive requests to WS-A
  - Result: WS-B can get upto 85% share of network

Evaluating Impact of Resource Contention

- Run two VMs on the same machine under 3 configurations
  - VMs are pinned to the same core
  - Pinned to the same package (i.e. share LLC)
  - Pinned to different packages (i.e. no cache sharing)

Benchmarks

Workload	Description
CPU	Solving the N-queens problem for $N = 14$.
Net	Lightweight web server hosting 32KB static web pages cached in memory, 5000 requests per second from a separate client.
Diskrand	Requests for randomly selected 4KB chunk in 1 GB span.
Memrand	Randomly request 4B from every 64B of data from a 64MB buffer.
LLC	Execute LLCProbe, which sequentially requests 4B from every 64B of data within an LLC-sized buffer using cache coloring to balance access across cache sets.

Increase in Run-times

Same core	CPU	Net	Diskrand	Memrand	LLC
CPU	-	5	-	-	-
Net	-	194	-	-	-
Diskrand	-	-	455	-	-
Memrand	-	6	-	-	-
LLC	8	539	72	38	34
Same package	CPU	Net	Diskrand	Memrand	LLC
CPU	-	-	-	-	-
Net	-	198	-	-	-
Diskrand	-	-	461	-	-
Memrand	-	-	17	-	-
LLC	20	448	55	566	566
Diff. package	CPU	Net	Diskrand	Memrand	LLC
CPU	-	20	-	-	-
Net	-	100	-	-	-
Diskrand	-	-	462	-	-
Memrand	-	35	-	-	-
LLC	6	699	11	15	15

Xen Scheduling Overview

- Credit scheduler
  - Allocates credits for each vCPU (typically 30ms time slices) at predefined rate
  - If out of credit, suspend and wait for new credit
  - For IO interrupt,
    - if vCPU has more credits
      - Allow it to have **boost** (preempt any VM as soon as IO is ready)
    - else no boost (treated as regular suspended VM)

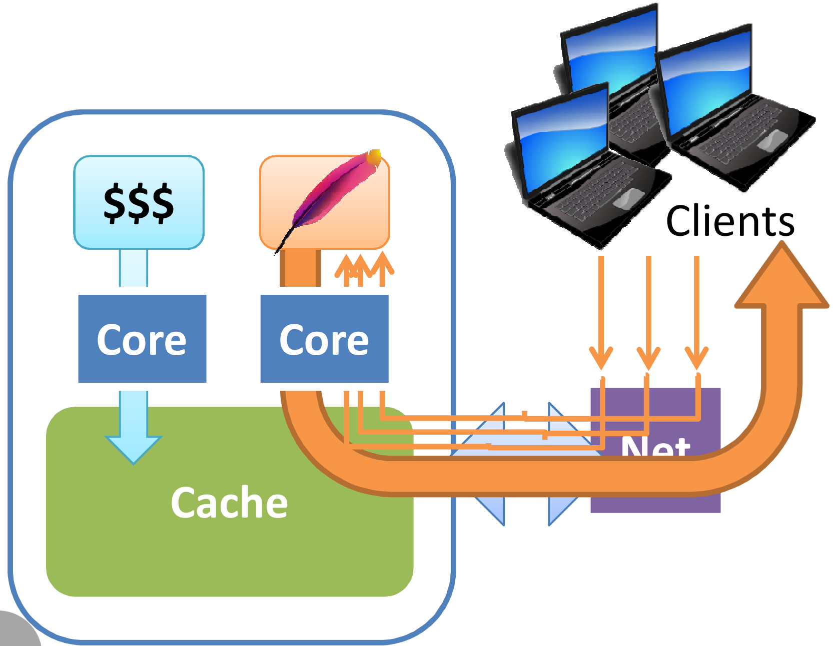
Understanding Cache/NW Contention

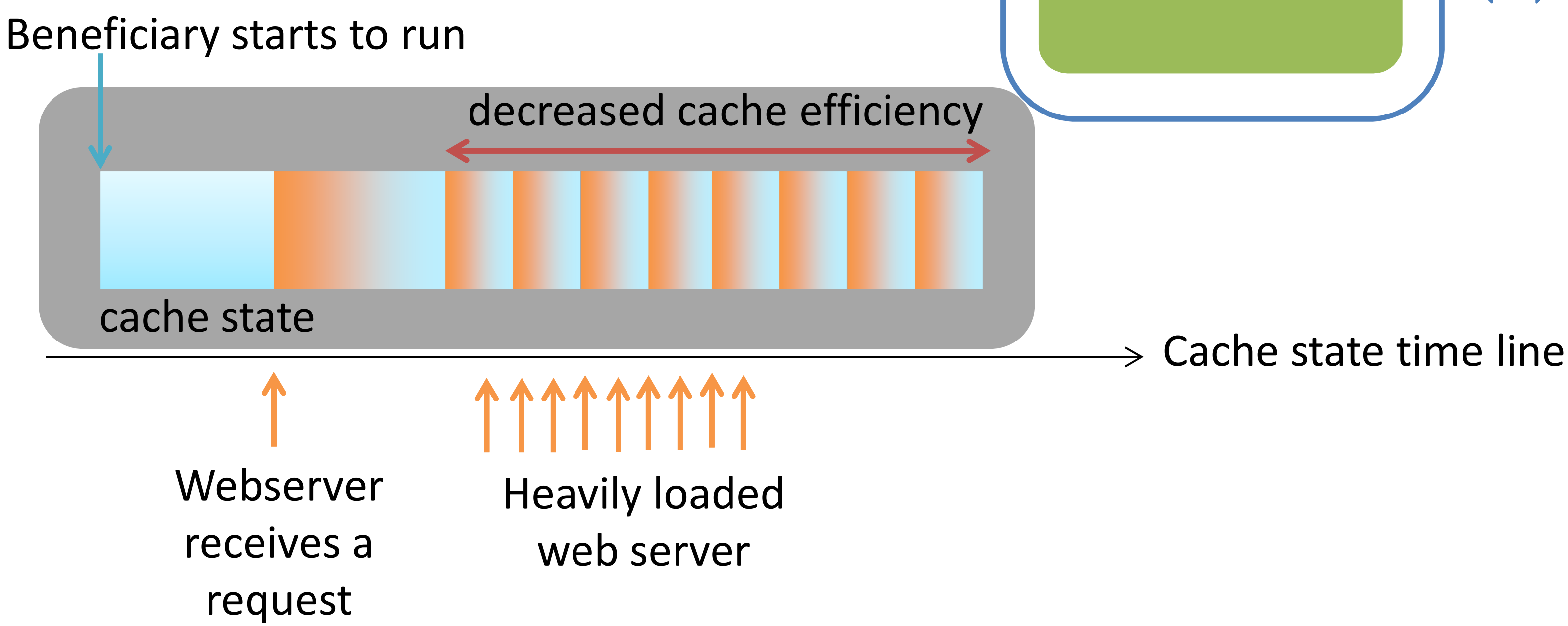
- Webserver With 3000rps
  - Runs < 1ms once it’s scheduled 80% of the time
- Webserver With 1500rps
  - Runs < 1ms once it’s scheduled 40% of the time
- Conclusion
  - Frequent interrupts give boost to WS VM
  - Preempt LLCProbe, effectively reducing its time slice

Cache vs. Network

Victim webserver frequently interrupts, pollutes the cache

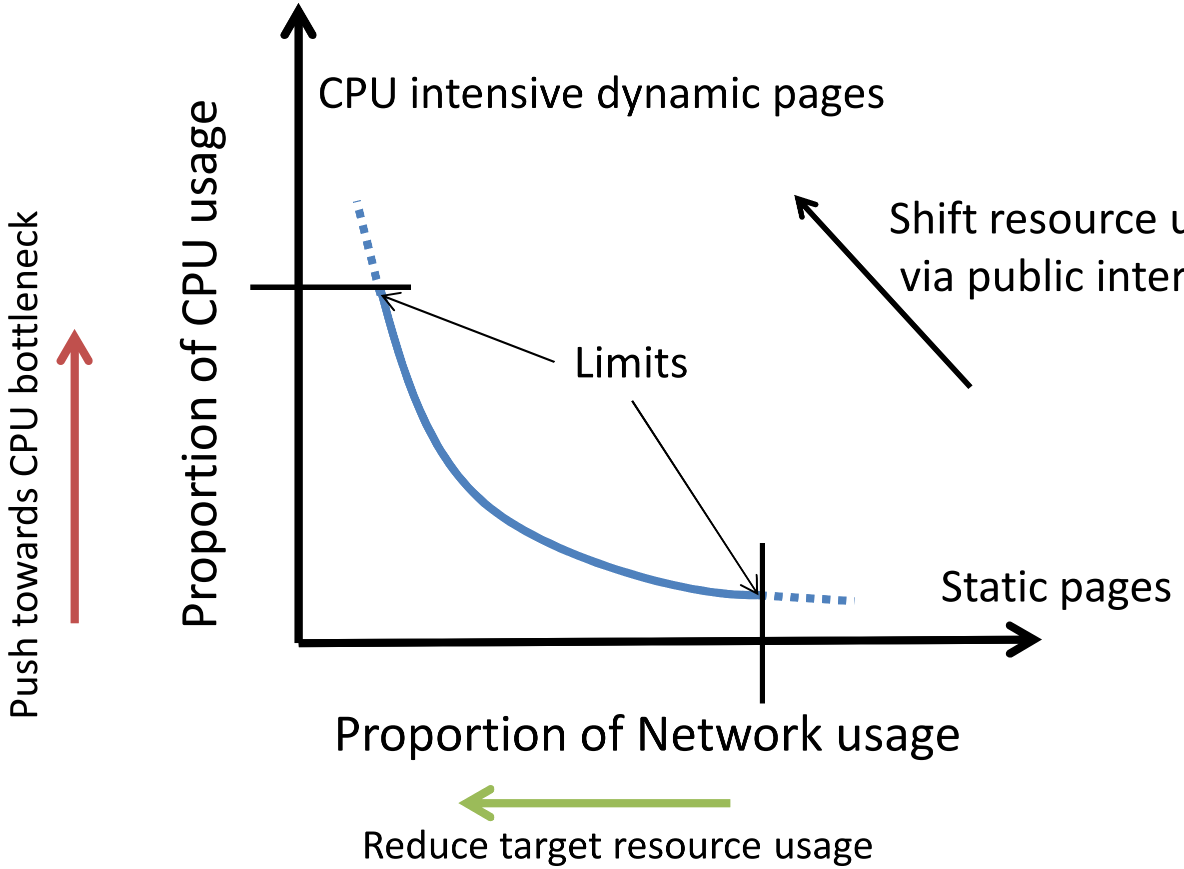
- Reason: Xen gives higher priority to VM consuming less CPU time





Recipe for a Successful RFA

<u>*Shift*</u> resource away from the *target resource* towards the <u>*bottleneck*</u> resource



Building RFA for Cache vs. Network

Victim:

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

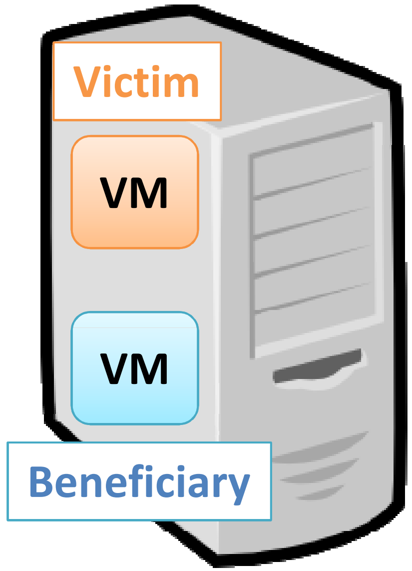
Beneficiary:

- VM whose performance we want to improve (LLCProbe)

Helper:

- Mounts the attack

Beneficiary and victim fighting over a *target resource* (cache)

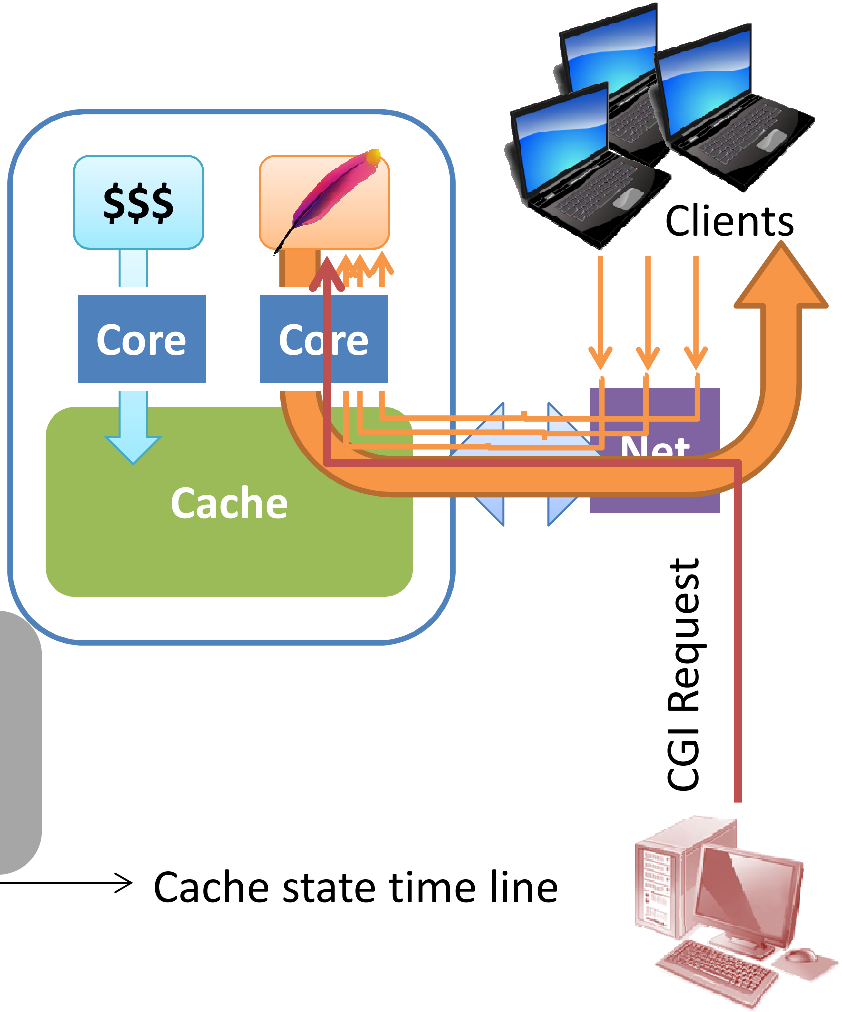


Helper

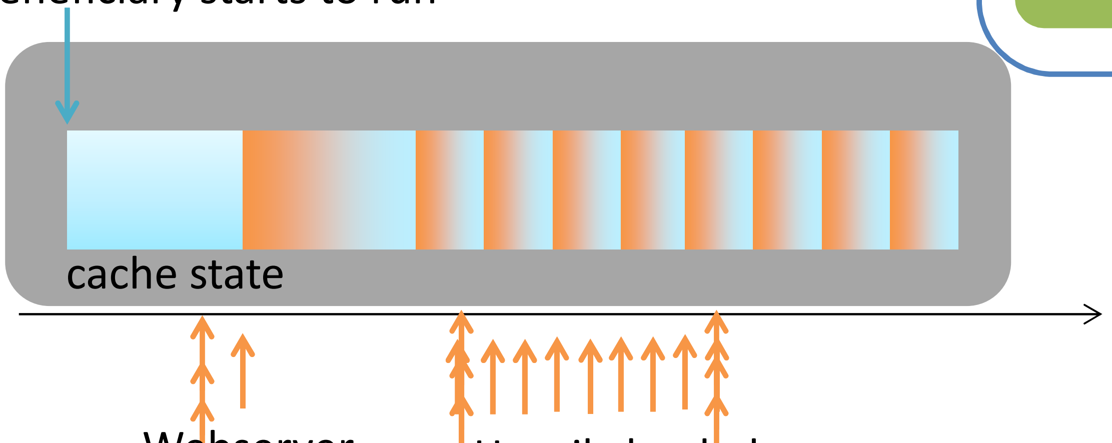
Cache vs. Network w/ RFA

RFA helps in two ways:

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



Beneficiary starts to run



Webserver receives a request

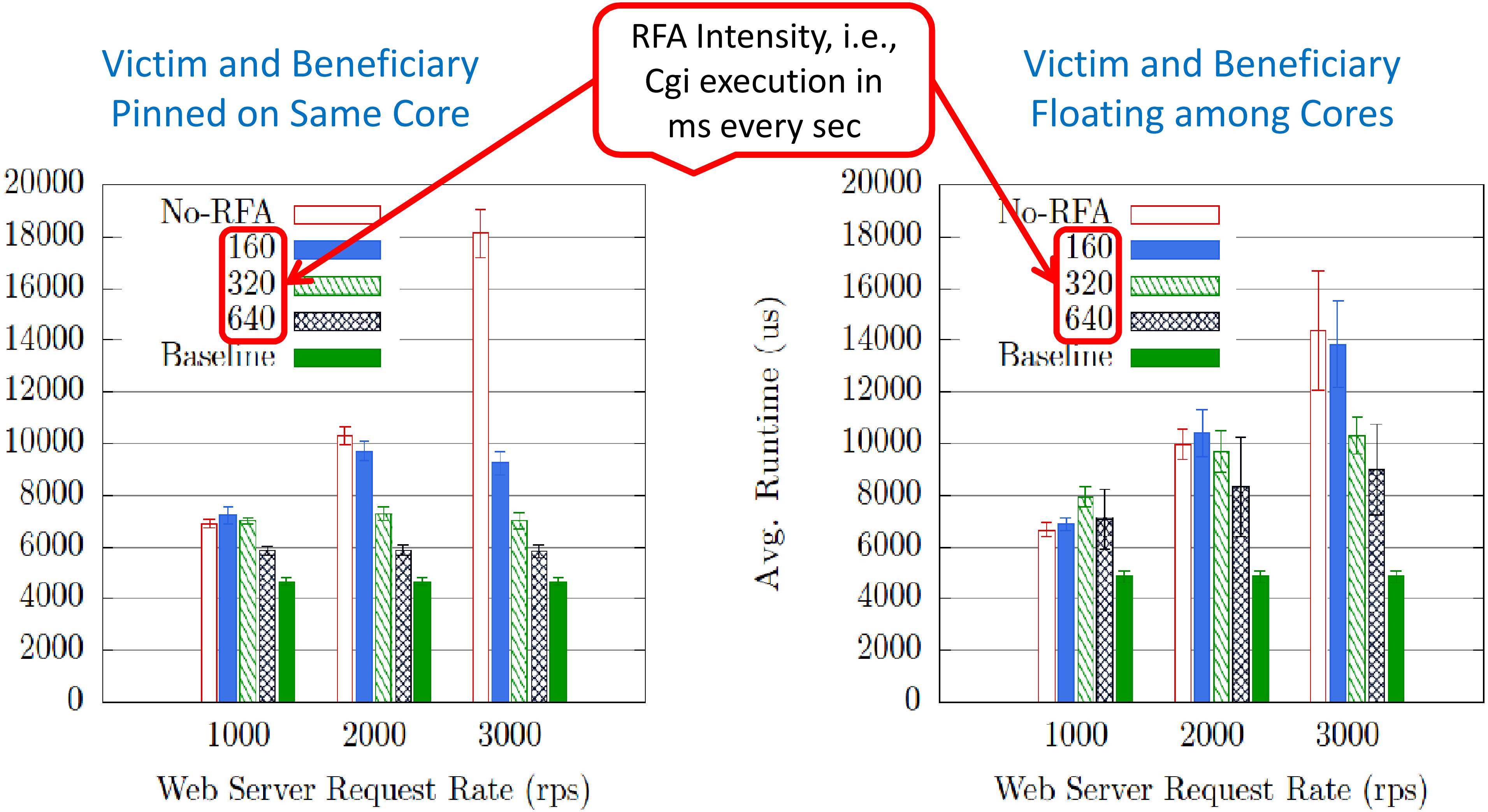
Heavily loaded web server

Heavily loaded webserver requests under RFA

Evaluation on Local Testbed

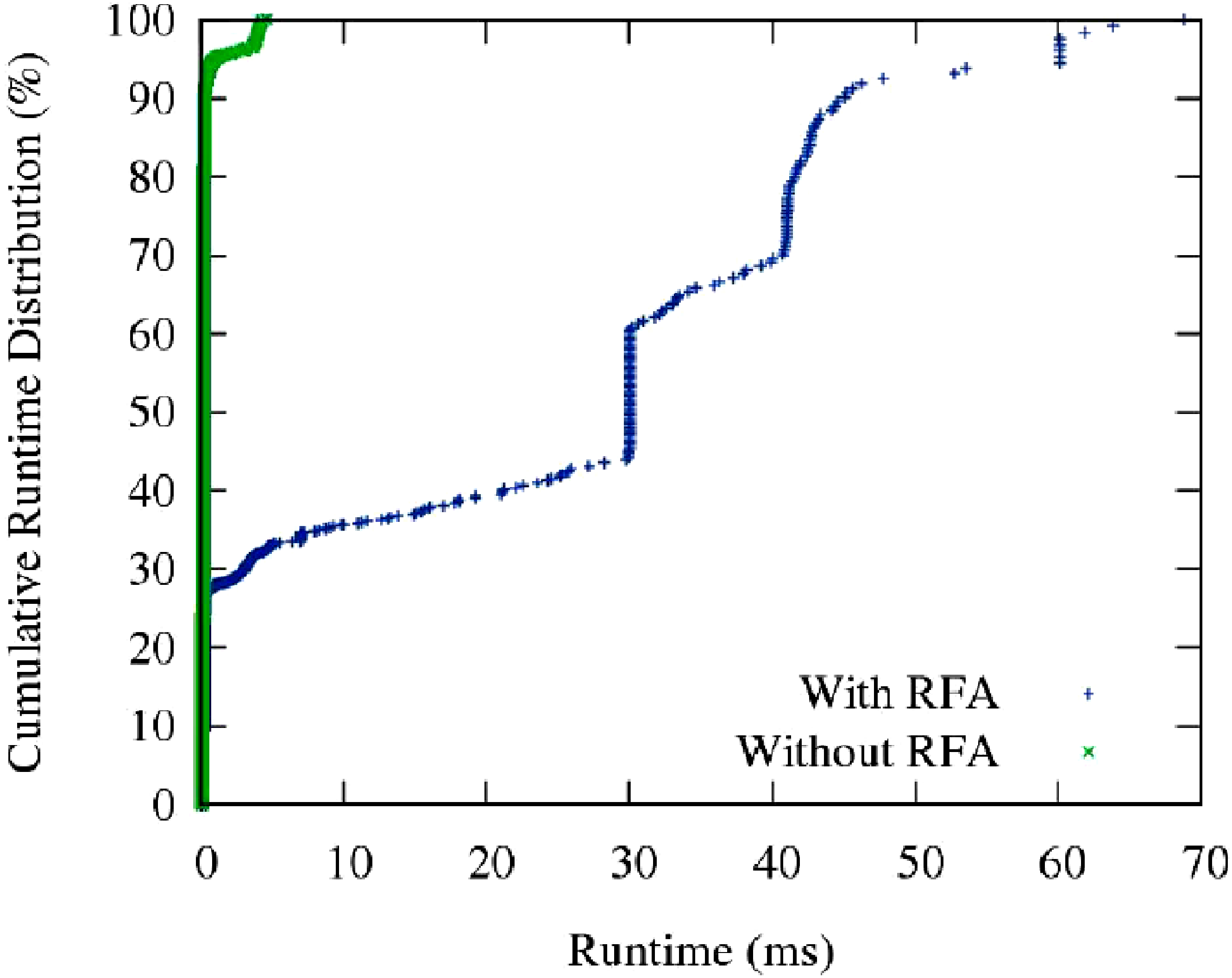
- Victim
  - Web server
    - 4096 32KB static pages
    - Custom CGI script performing busy-wait for a given duration
      - 40ms for the attack
- Beneficiary
  - LLCProbe benchmark co-located with victim
- Helper
  - A third VM on separate machine (can be any computer)

Performance of LLC with RFA

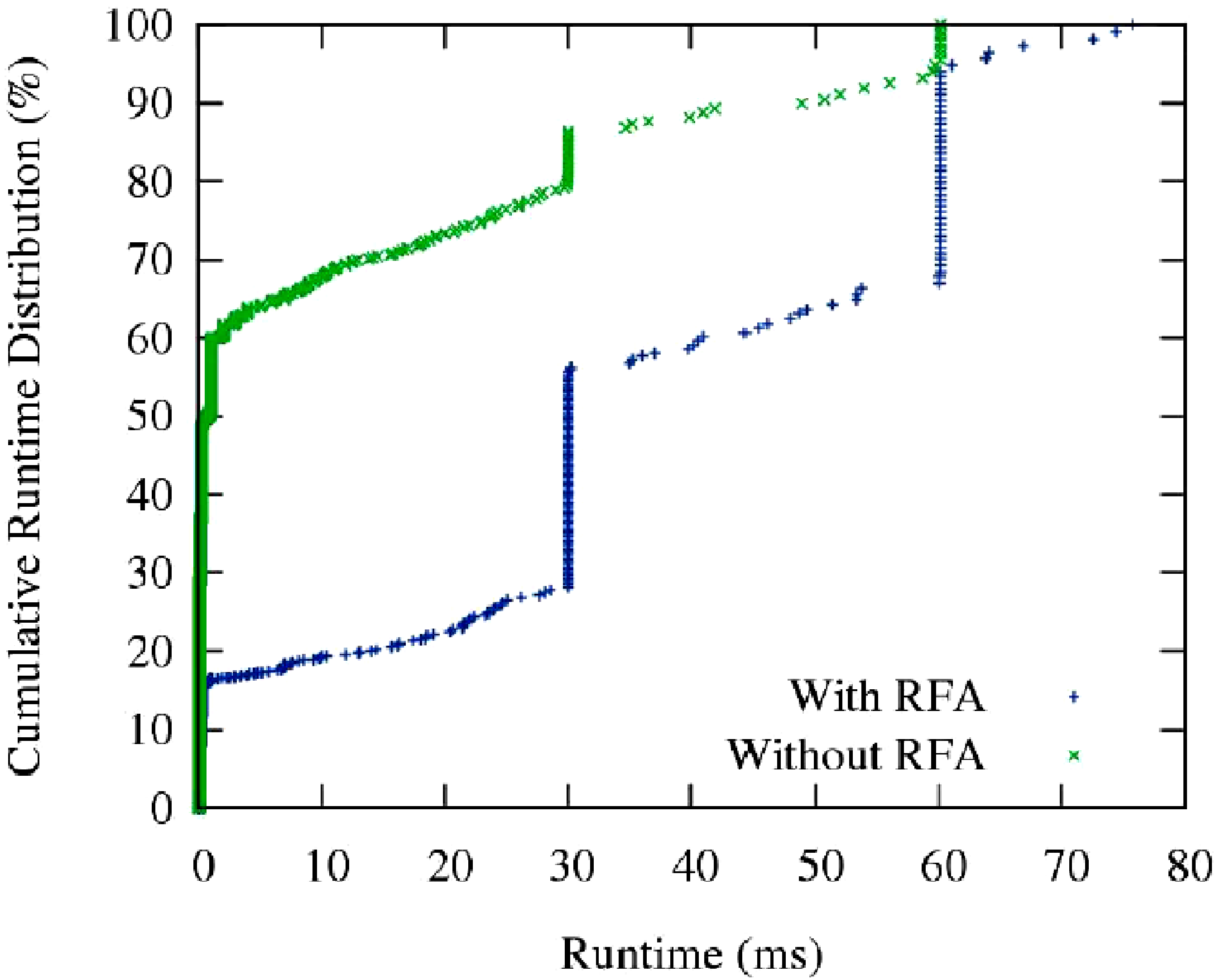


Beneficiary recovers much of the lost performance

Cumulative Runtime Distribution: Web Server



Cumulative Runtime Distribution: LLCProbe



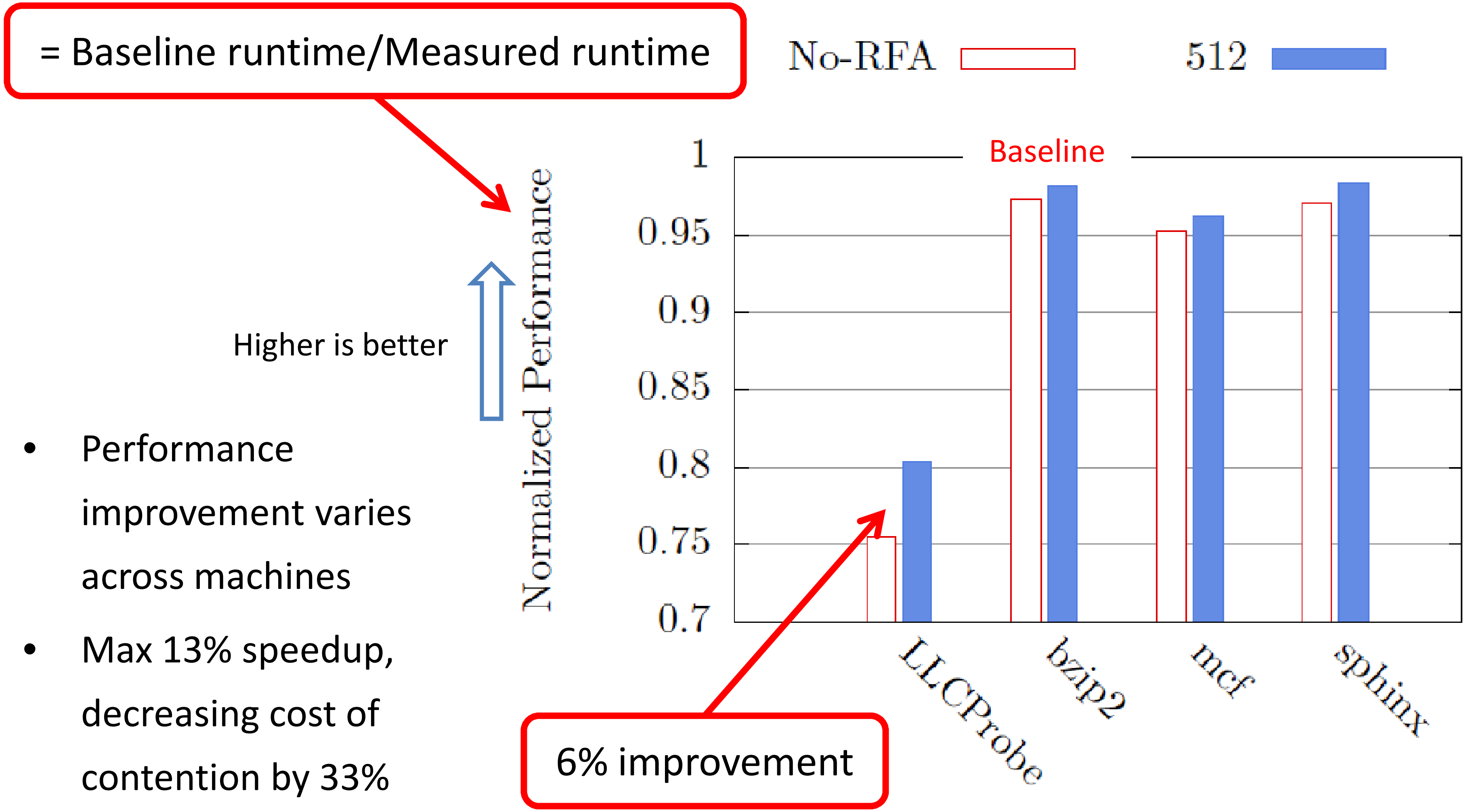
Effect on Victim's Capacity

- Capacity decreases with increased RFA intensity
- Important for different package/floating core situation

Evaluation on EC2

- Objective: See if RFA works in the noisy setting of EC2
- Methodology:
  - Achieve co-location by running large number of m1.small instances
  - Detect co-location by measuring nw latency, cache covert channel
  - Found 12 machines with co-located VMs, 9 are Intel Xeon E5507
  - Run experiments on these 9 machines

RFA Performance in EC2



- Performance improvement varies across machines
- Max 13% speedup, decreasing cost of contention by 33%

Practical Issues

- RFA requires knowledge of
  - Public interface to victim
  - Workload of victim
  - How to shift bottleneck resource
- Web servers may be behind a load balancer
- Cost of running helper
  - Typically small

Preventing RFA

- Better isolation
  - Use non-work conserving scheduling
    - VMs given fixed allocation, idle resources may not be used by active VMs if its quota is fulfilled
  - May conflict with utilization
- Smarter scheduling
  - Group VMs based on resource conflicts
  - Place VMs contending for cache on different packages
  - Schedule VMs with resource conflicts at different times

Conclusion

- Presented a new type of attack exploiting imperfect isolation in cloud
- Attacker improves its own performance at the cost of victim
- Costly ($$$) to both victim and cloud provider, profitable to attacker
- Showed thorough evaluation of how to build such an attack

References

- Venkatanathan Varadarajan, Thawan Kooburat, Benjamin Farley, Thomas Ristenpart, and Michael M. Swift. 2012. Resource-freeing attacks: improve your cloud performance (at your neighbor's expense). In Proceedings of the 2012 ACM conference on Computer and communications security (CCS '12). ACM, New York, NY, USA, 281-292.
- Thomas Ristenpart, Eran Tromer, Hovav Shacham, and Stefan Savage. 2009. Hey, you, get off of my cloud: exploring information leakage in third-party compute clouds. In Proceedings of the 16th ACM conference on Computer and communications security (CCS '09). ACM, New York, NY, USA, 199-212.
- Authors' presentation: http://pages.cs.wisc.edu/~venkatv/ccs12-rfa.pptx