Cornell University
Computer Systems Laboratory

Workload Characterization of Interactive Cloud Services on Big and Small Server Platforms

Shuang Chen*, Shay Galon**, Christina Delimitrou*,
Srilatha Manne**, and José Martínez*

*Cornell University
**Cavium Inc.

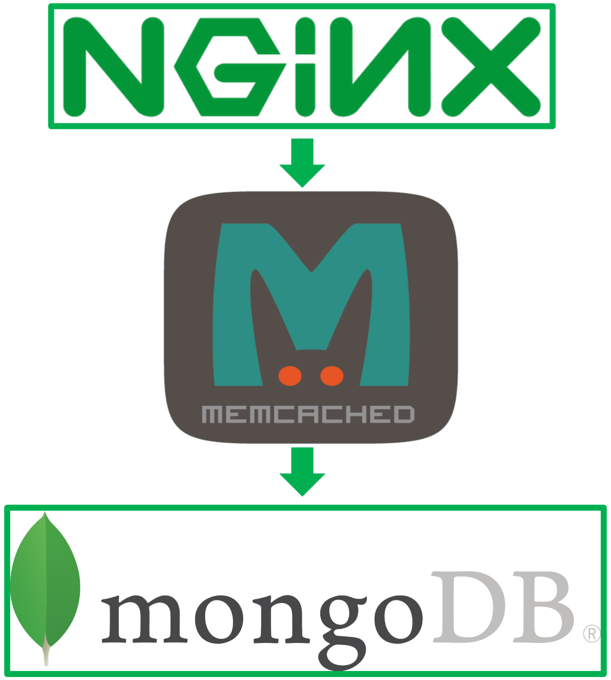
EXECUTIVE SUMMARY

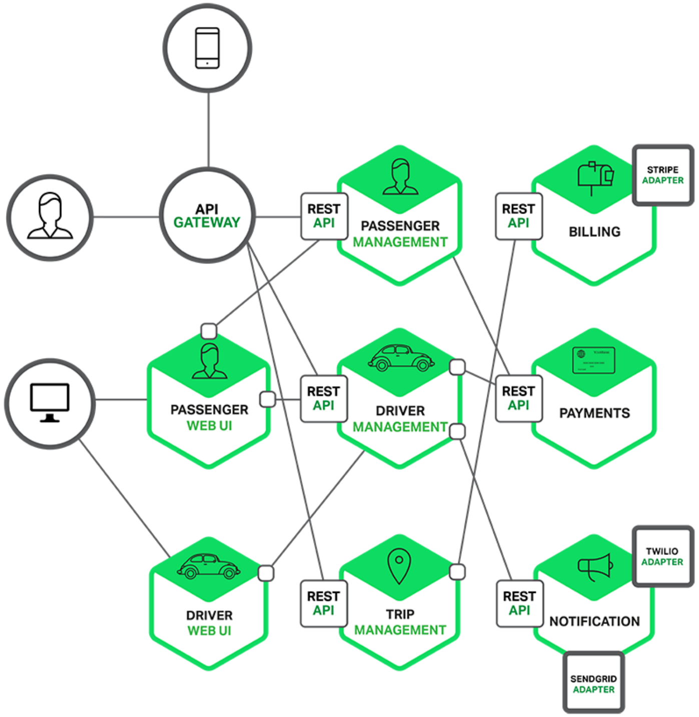
- **How to achieve low tail latency for interactive cloud services?**
  - Tail latency more important and challenging
  - The entire stack from SW to HW is involved

- **Understand how tail latency reacts to application and system changes**
  - Quantify how current designs work
  - Get insights on future designs

Motivation

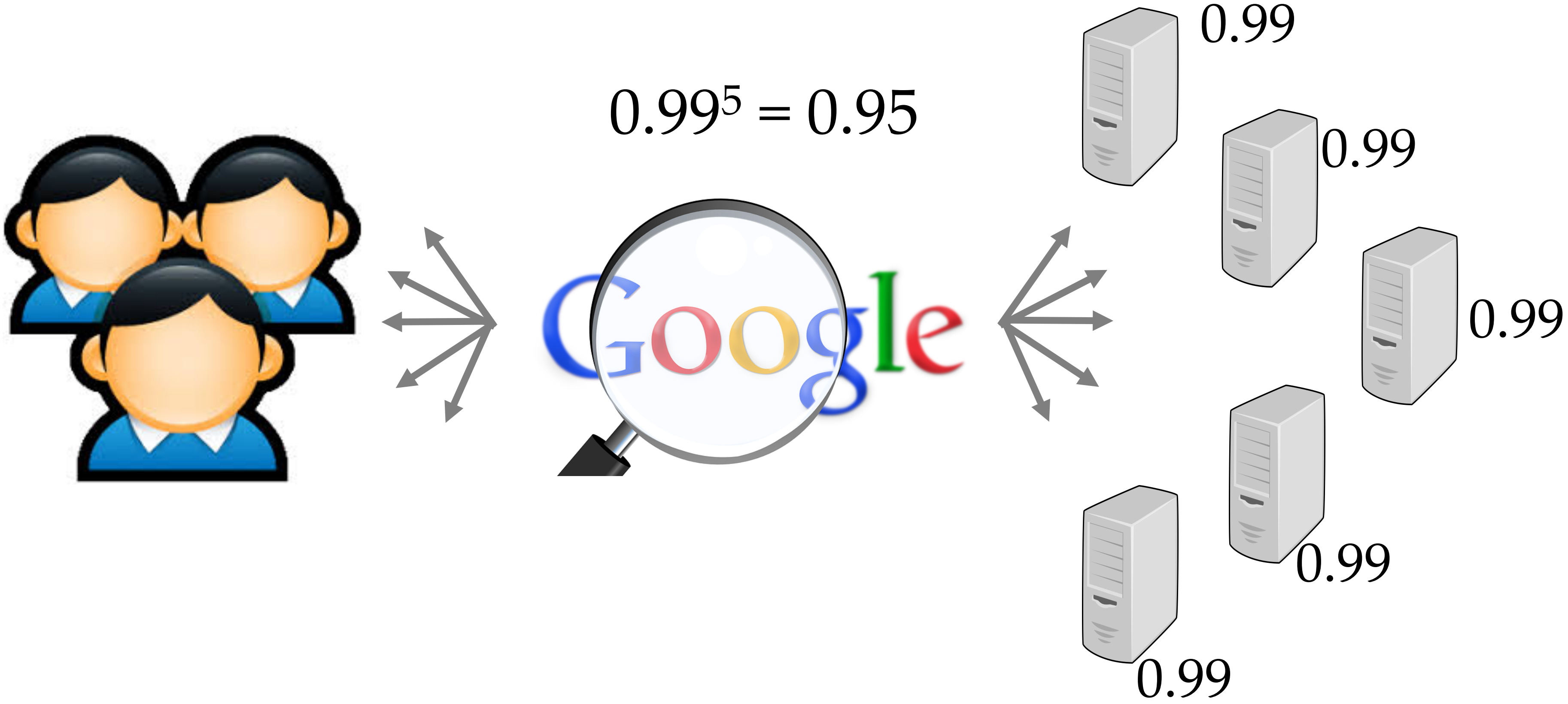
Google Translate
Bing
Search
Google Maps


NGINX
MEMCACHED
mongoDB


API GATEWAY
REST API
PASSENGER MANAGEMENT
REST API
BILLING
STRIPE ADAPTER
PASSENGER WEB UI
REST API
DRIVER MANAGEMENT
REST API
PAYMENTS
DRIVER WEB UI
REST API
TRIP MANAGEMENT
REST API
NOTIFICATION
TWILIO ADAPTER
SENDGRID ADAPTER

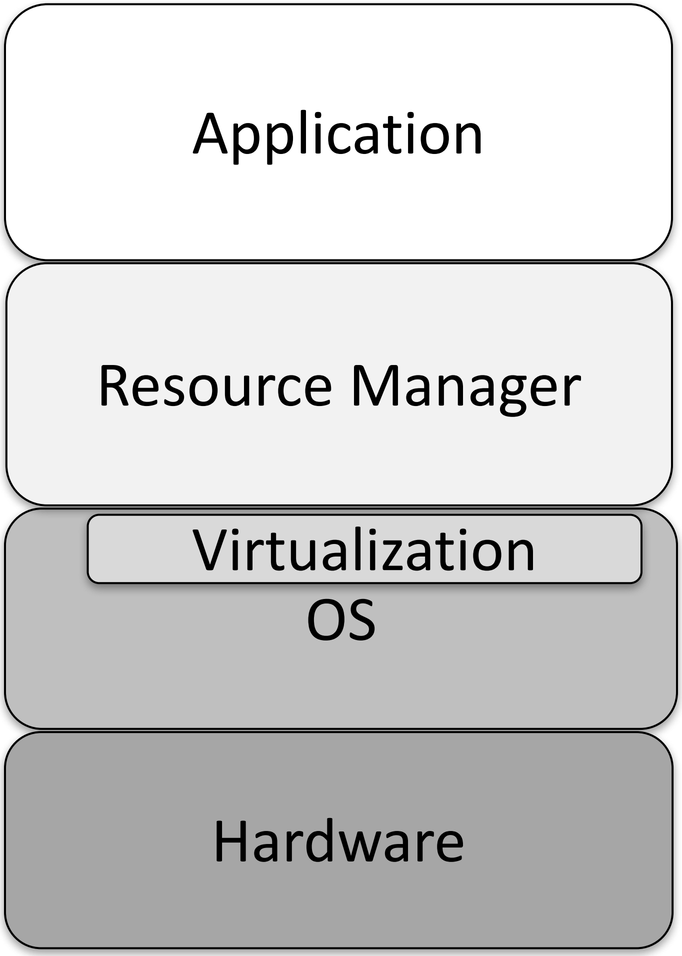
Low Latency

- **Tail latency**
  - e.g., QoS defined as 99th %ile in 1ms



Low Tail Latency Requirements

- The entire stack from SW to HW is involved



- Application bottleneck
- Different user cases
- Scalability

- Overhead of virtualization
- SW isolation mechanisms
- Overhead of context switching

- HW isolation mechanisms
- Hyperthreading

CATEGORIZE LC APPLICATIONS

By QoS Strictness

- us: memcached
- ms: web server, in-memory database
- s: persistent database

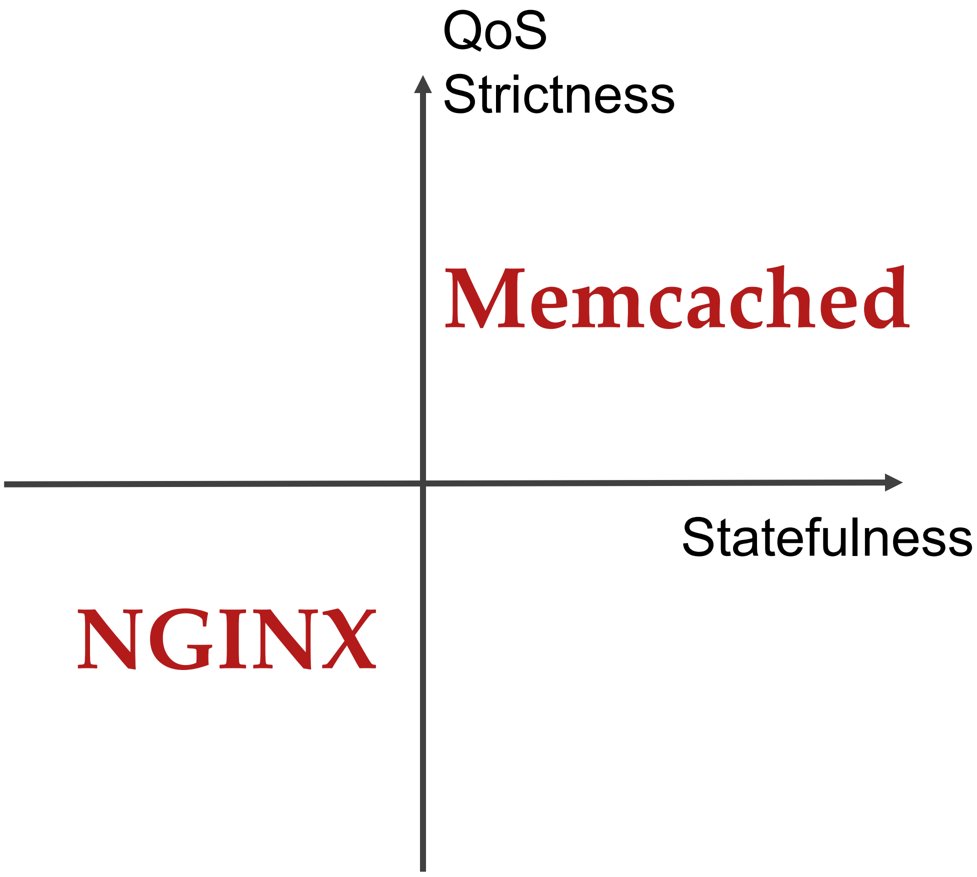
By Statefulness

- Stateful: memcached
- Stateless: web server

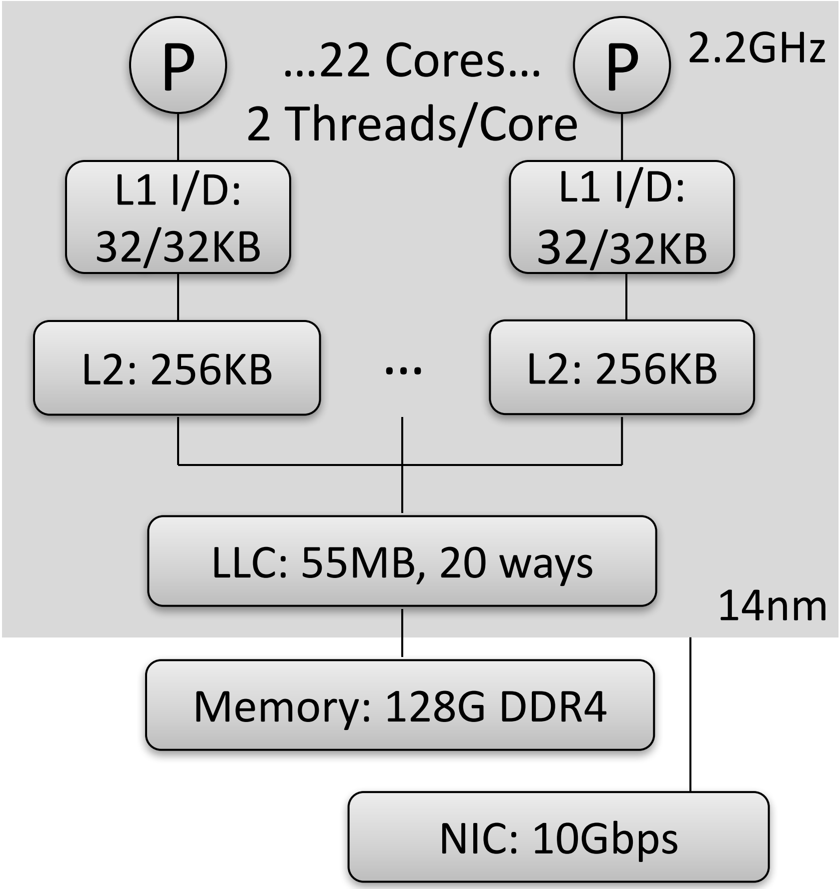
Selected LC Workloads

- **NGINX**
  - Web server
  - Stateless
  - 99th% in tens of ms

- **Memcached**
  - Key-value store
  - Stateful
  - 99th% in hundreds of us



Server Architecture



P ...22 Cores... P

2 Threads/Core

2.2GHz

L1 I/D: 32/32KB

L1 I/D: 32/32KB

L2: 256KB

...

L2: 256KB

LLC: 55MB, 20 ways

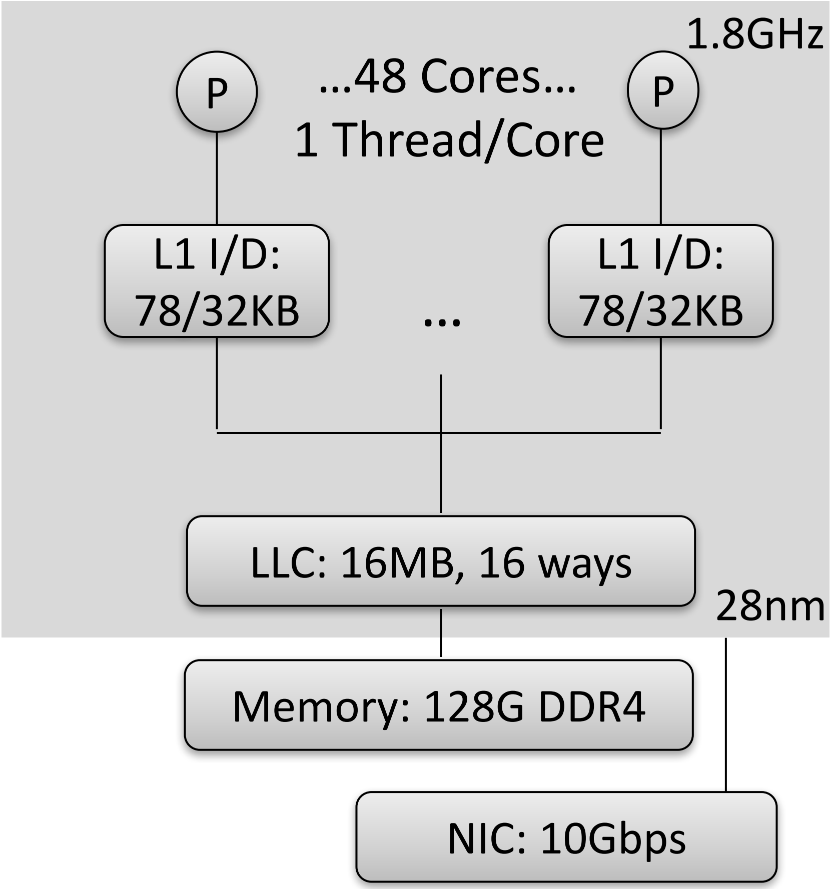
14nm

Memory: 128G DDR4

NIC: 10Gbps

Intel Xeon E5-2699 v4

$4,115



1.8GHz

P ...48 Cores... P

1 Thread/Core

L1 I/D: 78/32KB

...

L1 I/D: 78/32KB

LLC: 16MB, 16 ways

28nm

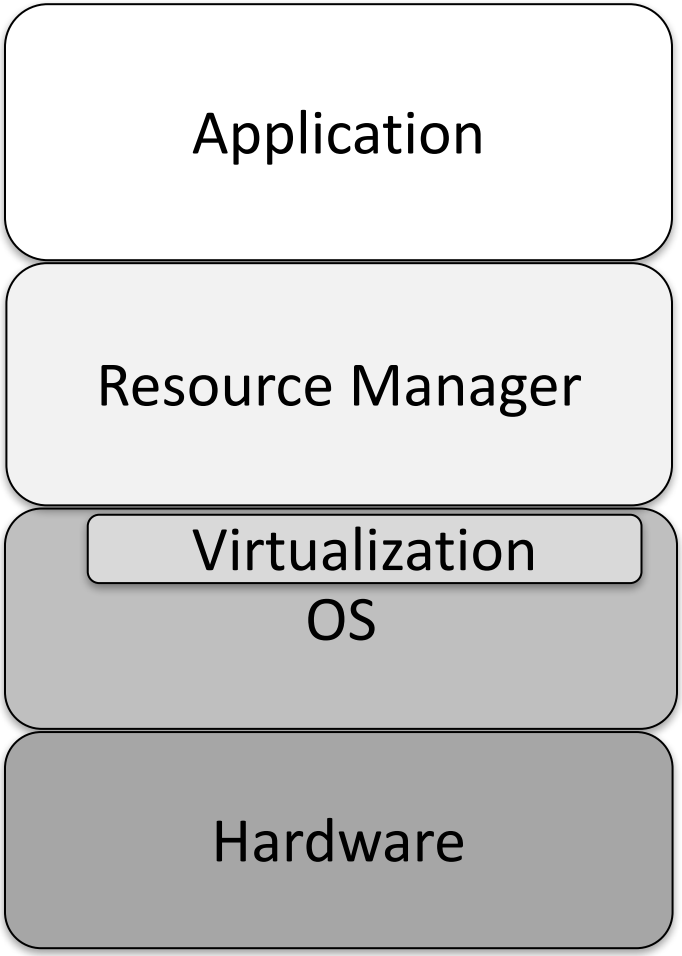
Memory: 128G DDR4

NIC: 10Gbps

Cavium ThunderX

$785

Studied Parameters



Application

Resource Manager

Virtualization

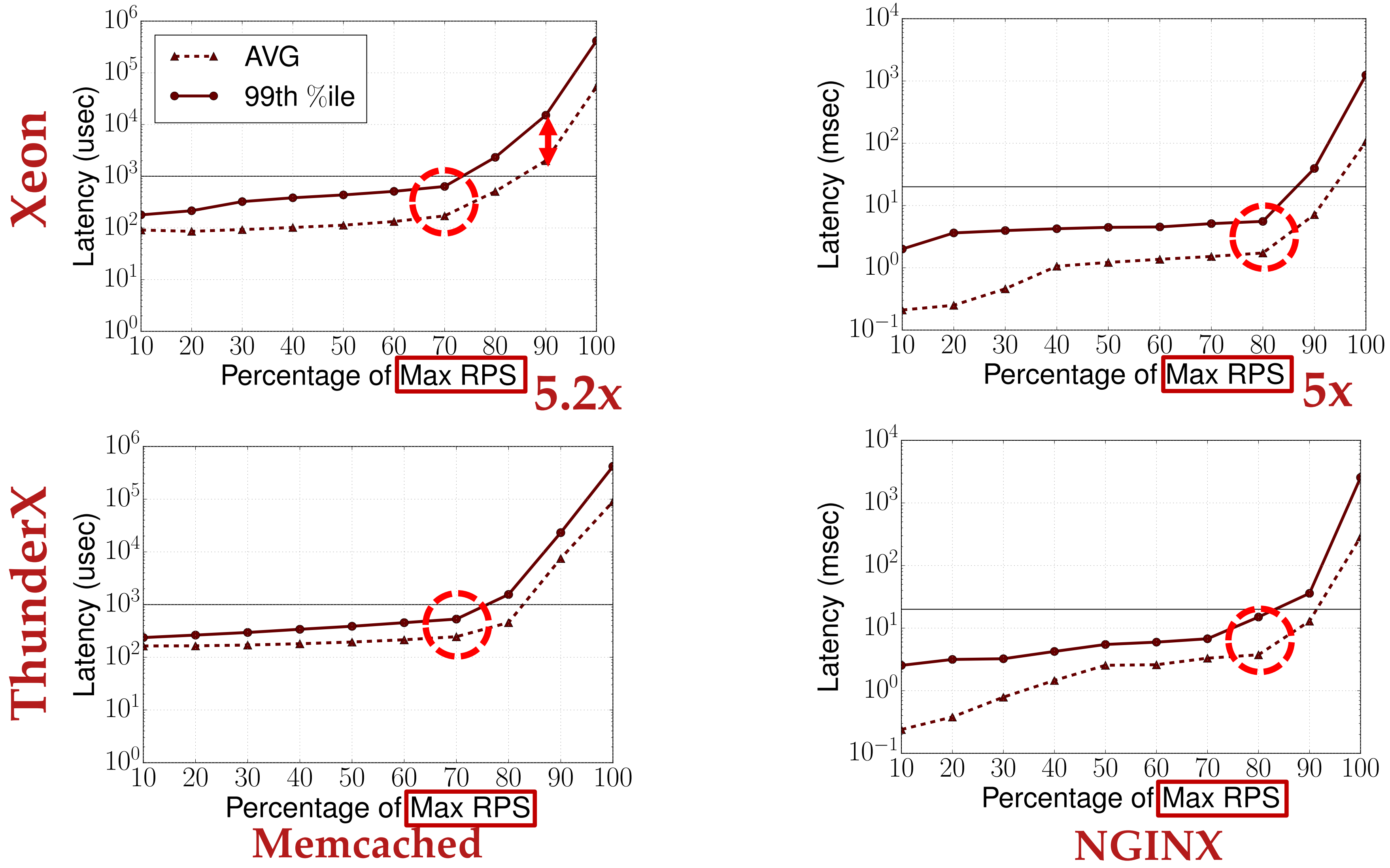
OS

Hardware

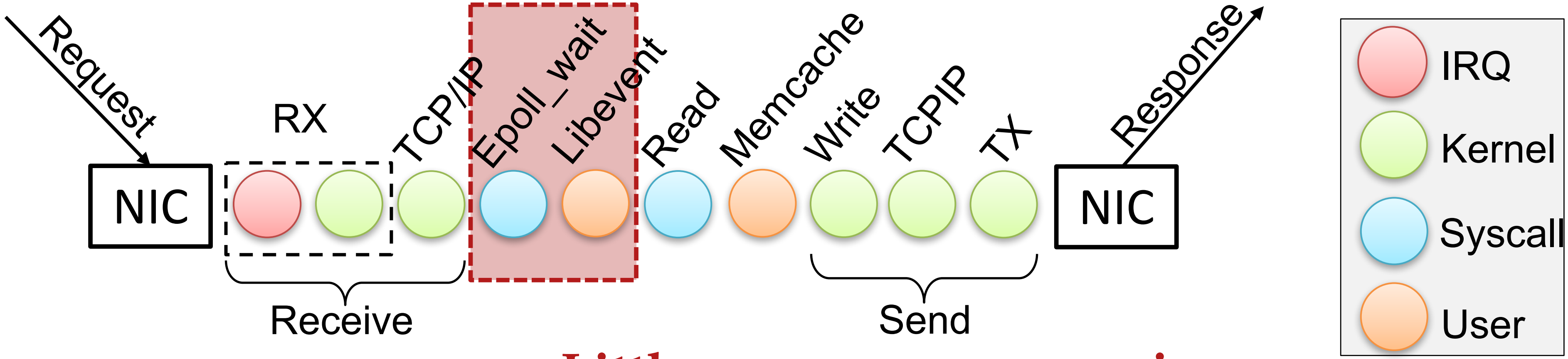
- **Application bottleneck**
- Different user cases
- Scalability

- Overhead of virtualization
- SW isolation mechanisms
- Overhead of context switching
- HW isolation mechanisms
- Hyperthreading

Input Load

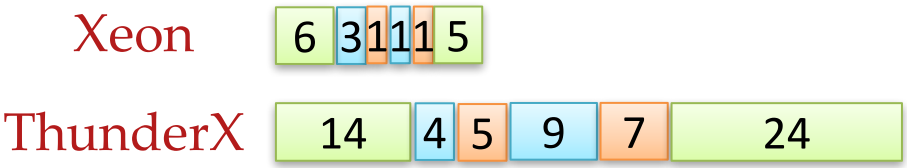

Xeon
ThunderX
AVG
99th %ile
Latency (usec)
Latency (msec)
10^{0}
10^{1}
10^{2}
10^{3}
10^{4}
10^{5}
10^{6}
10^{-1}
10 20 30 40 50 60 70 80 90 100
Percentage of Max RPS
5.2x
5x
Memcached
NGINX

Memcached Latency Decomposition



Little user-space processing

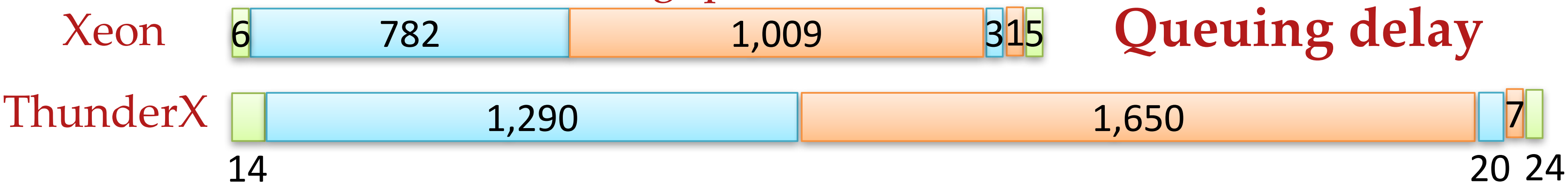
At **10%** of max throughput



Network delay

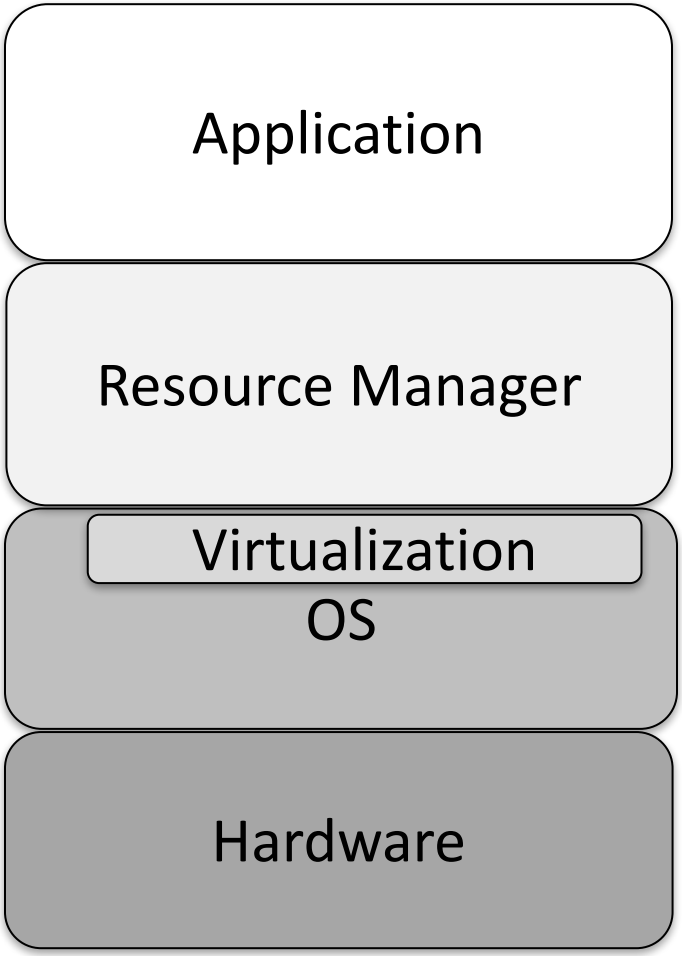
2x slower than Xeon

At **90%** of max throughput



Queuing delay

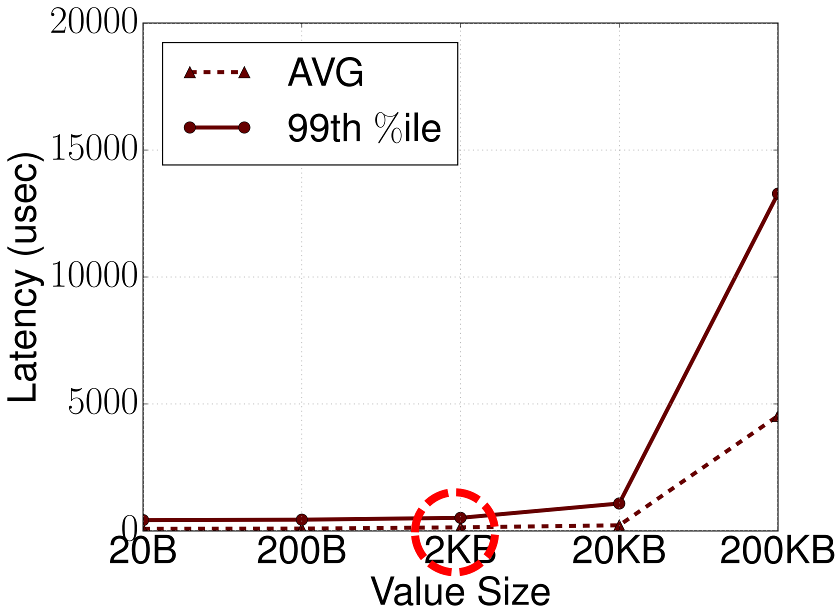
Studied Parameters



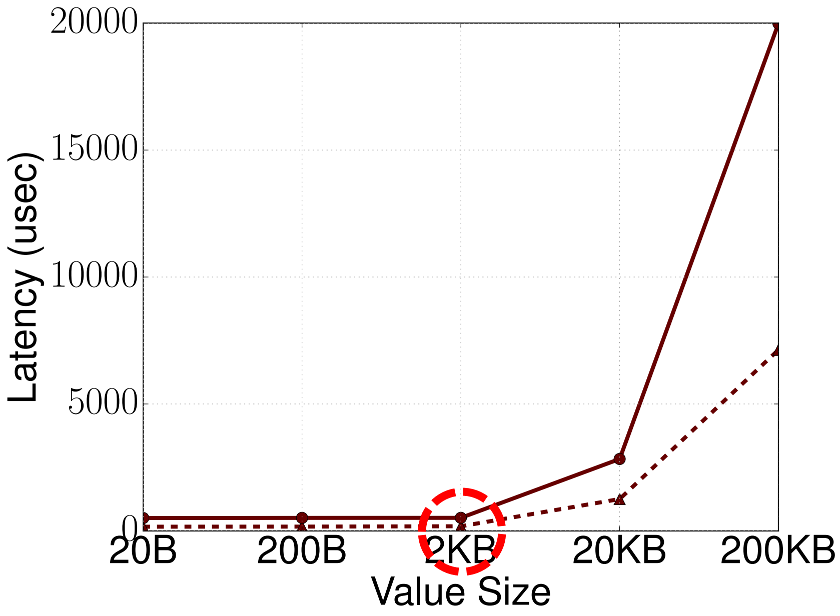
- Application bottleneck
- **Different user cases**
- Scalability

- Overhead of virtualization
- SW isolation mechanisms
- Overhead of context switching
- HW isolation mechanisms
- Hyperthreading

Memcached Value Size



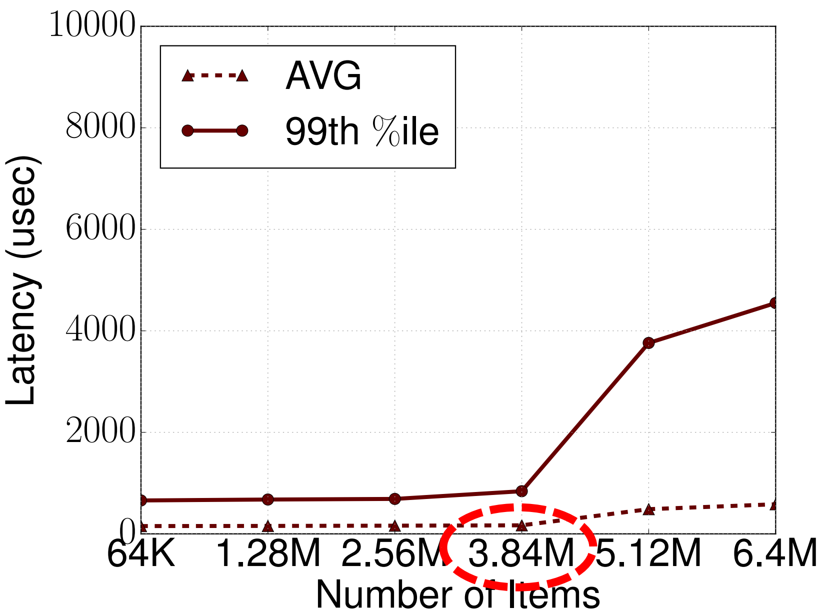
Xeon



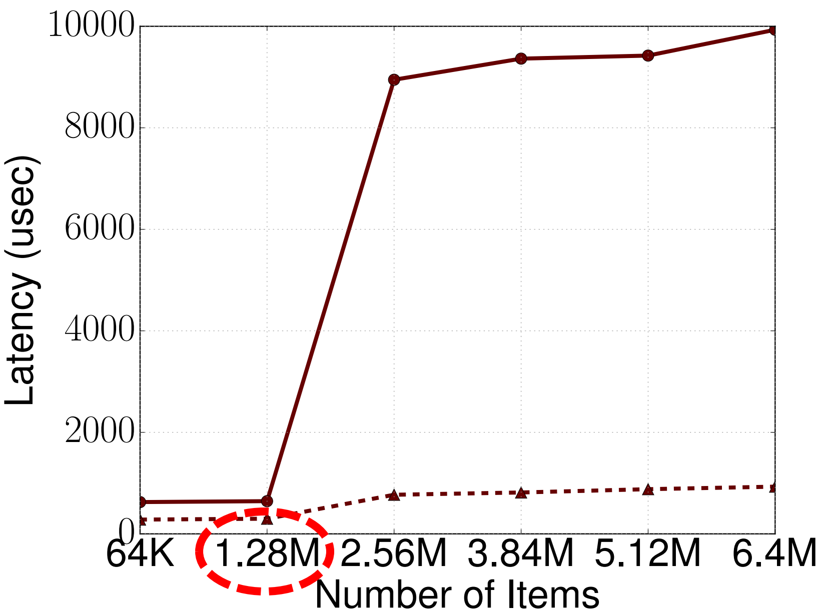
ThunderX

- Memory copy
- Network processing and transmission
- ThunderX is more sensitive

Number of Memcached Items



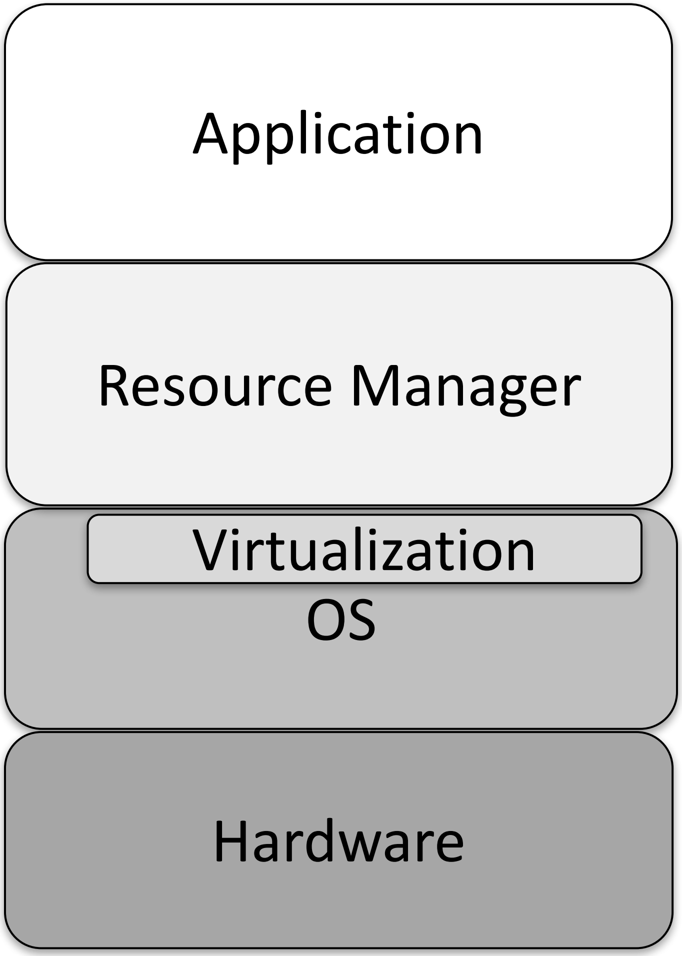
Xeon



ThunderX

- Cache capacity
- ThunderX is more sensitive

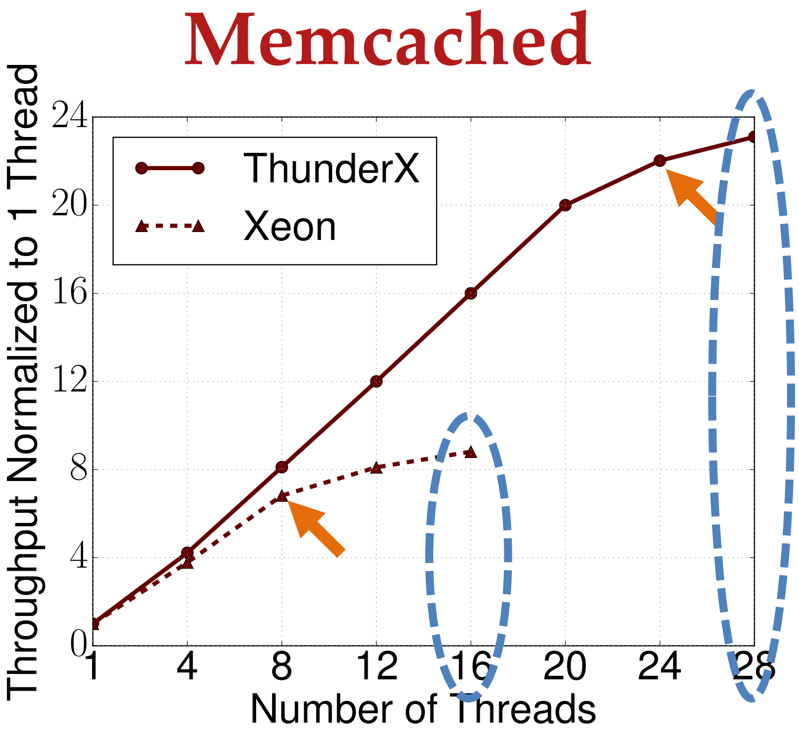
Studied Parameters

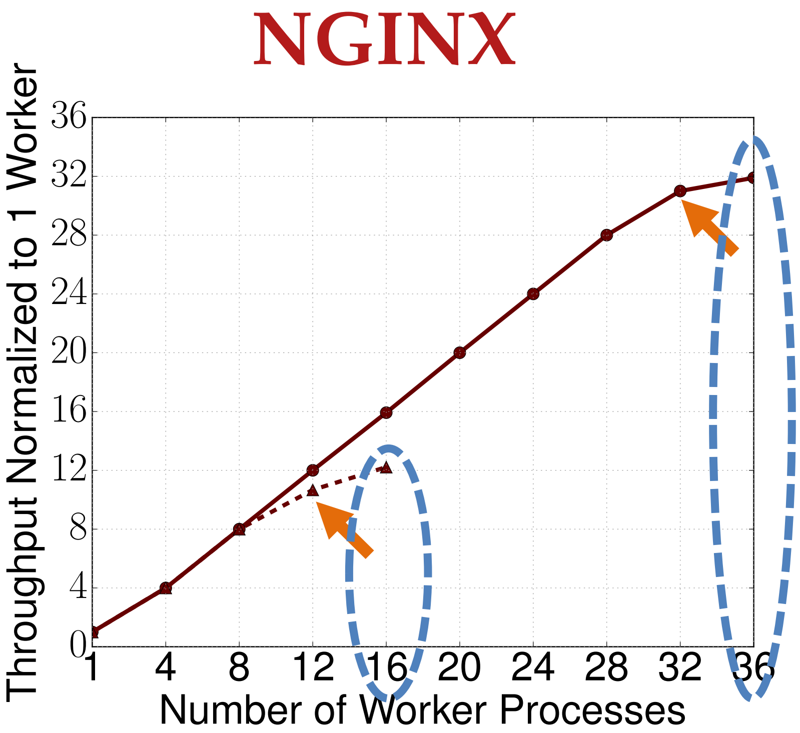


- Application bottleneck
- Different user cases
- **Scalability**

- Overhead of virtualization
- SW isolation mechanisms
- Overhead of context switching
- HW isolation mechanisms
- Hyperthreading

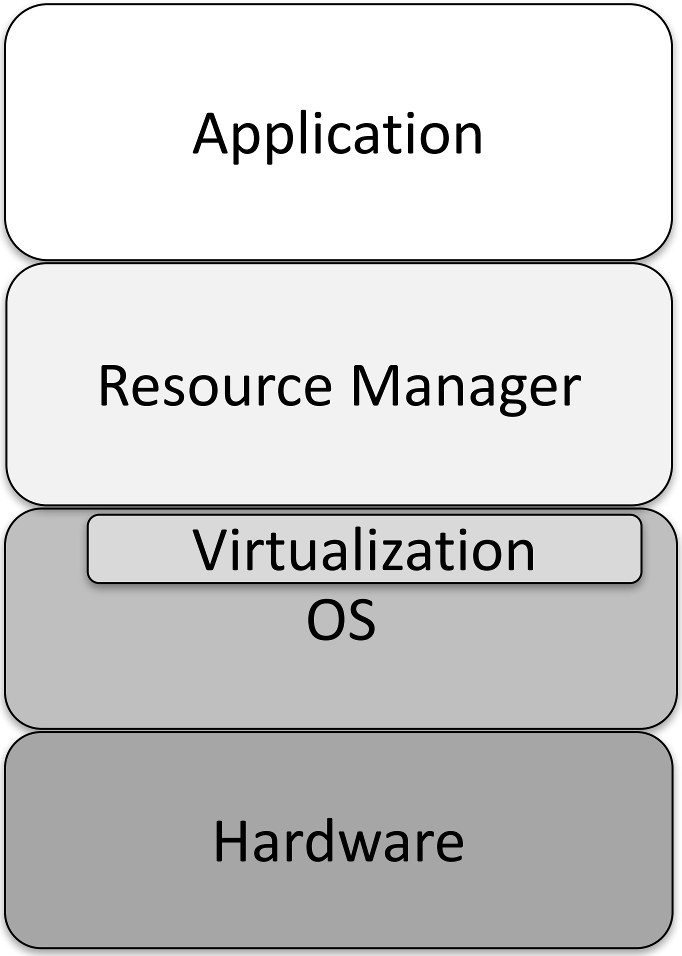
SCALABILITY





- Interrupt handling
- Load imbalance
- Lock contention

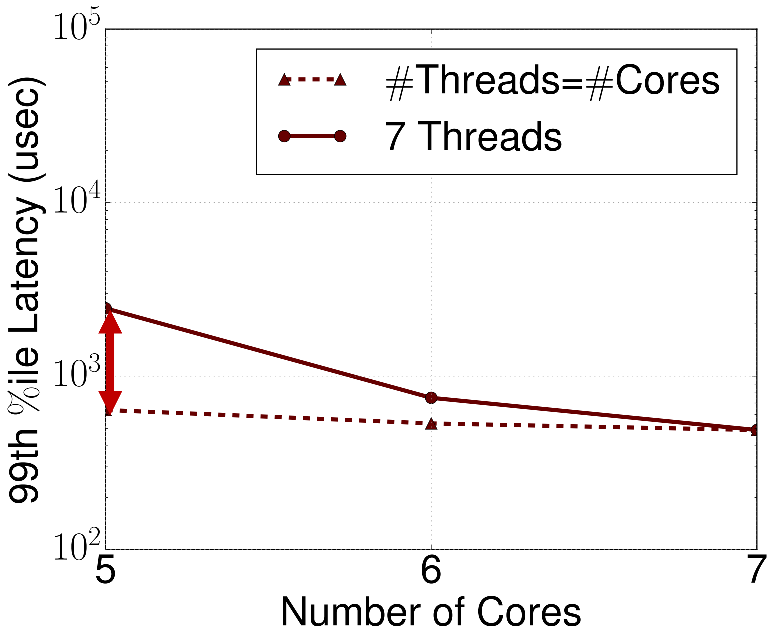
Studied Parameters



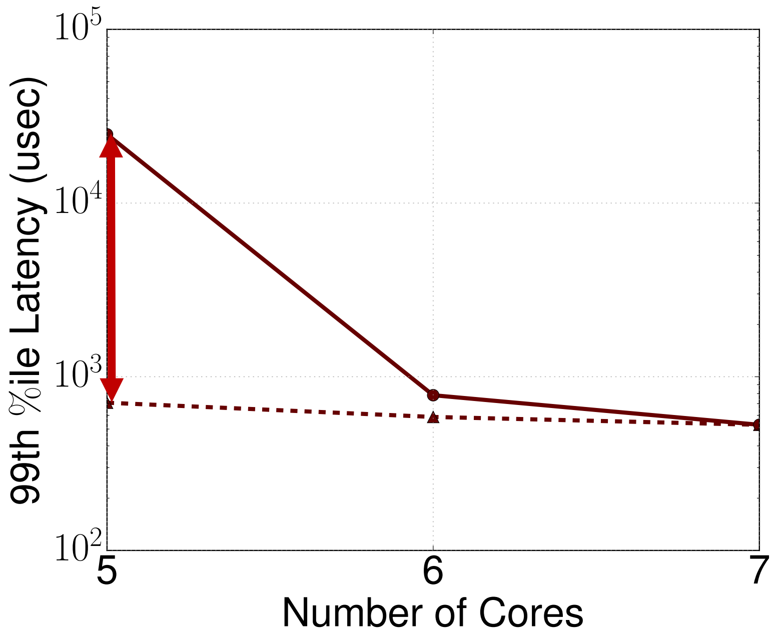
- Application bottleneck
- Different user cases
- Scalability

- Overhead of virtualization
- SW isolation mechanisms
- **Overhead of context switching**
- HW isolation mechanisms
- Hyperthreading

Context Switching



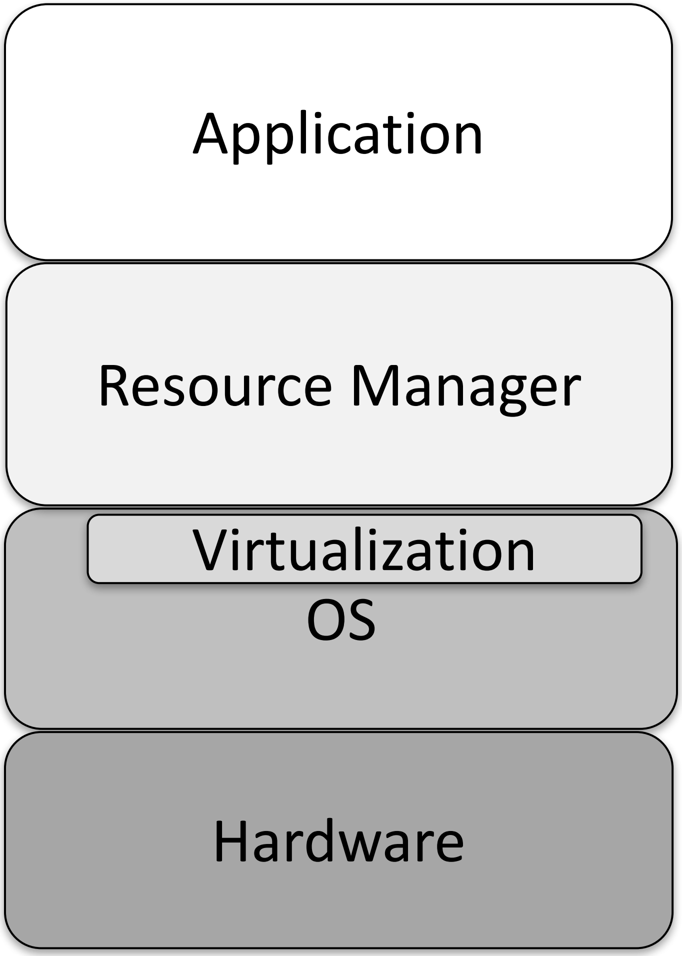
Memcached on Xeon



Memcached on ThunderX

- Statically spawned threads VS dynamically allocated cores
- ThunderX is more sensitive

Studied Parameters



- Application bottleneck
- Different user cases
- Scalability

- Overhead of virtualization
- SW isolation mechanisms
- Overhead of context switching
- HW isolation mechanisms
- **Hyperthreading**

HYPERTHREADING

- **Reduce the overhead of context switching**
  - Allocate two threads on two hyperthreads
- **Make better use of execution units**
  - Co-locate different applications

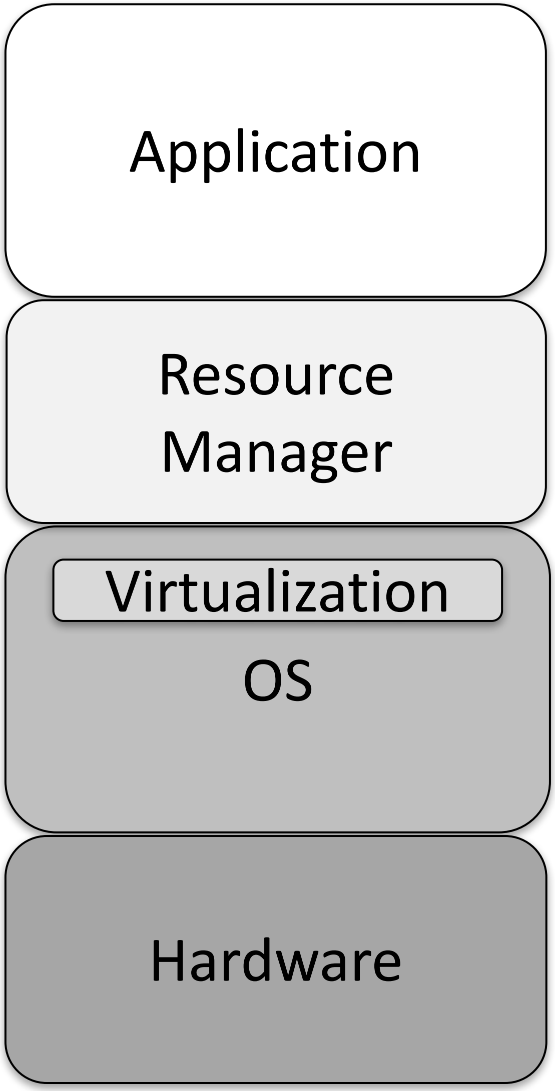
	10%	20%	30%	40%	50%	60%	70%
10%	MN	MN	MN	MN			
20%	MN	MN	MN				
30%	MN	MN					
40%	MN	N					
50%	N	N					
60%	N						
70%							

Memcached & Nginx on the same hyperthreads

	10%	20%	30%	40%	50%	60%	70%
10%	MN	MN	MN	MN	MN	MN	M
20%	MN	MN	MN	MN	MN	M	M
30%	MN	MN	MN	MN	M	M	M
40%	MN	MN	MN	MN	M	M	M
50%	MN	MN	N	N			
60%	N	N	N				
70%	N	N					

Memcached & Nginx on different hyperthreads

QUESTIONS?



- Reduce network/queuing delays
- Optimize common user cases
- Improve elasticity
  - Lock alternatives
  - Load balance

- Reduce the overhead of virtualization
- Reduce context switching
- Make best use of SW isolation mechanisms

- Big VS Small Cores
- Make best use of HW features