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.

- **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**
  - See how current designs work
  - Get insights on future designs

Motivation

Google Translate

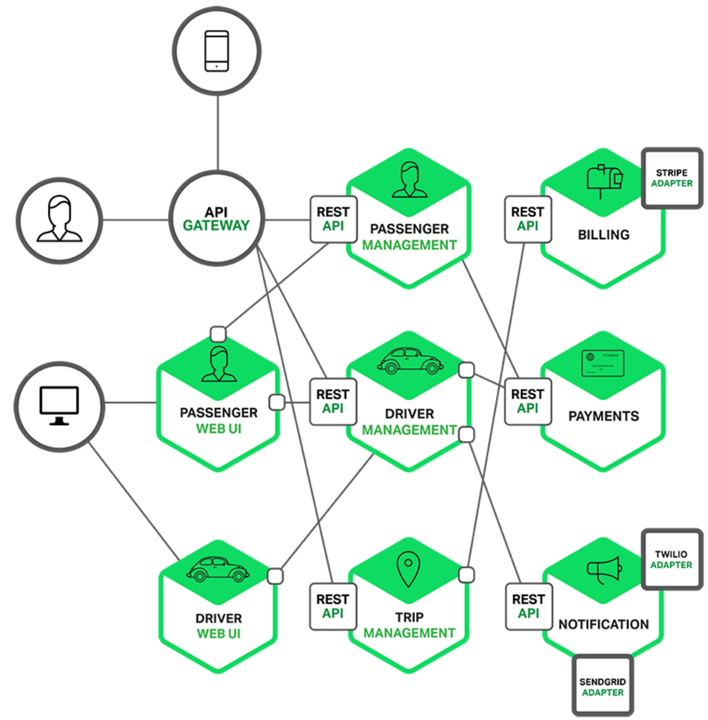
Bing

Google Maps

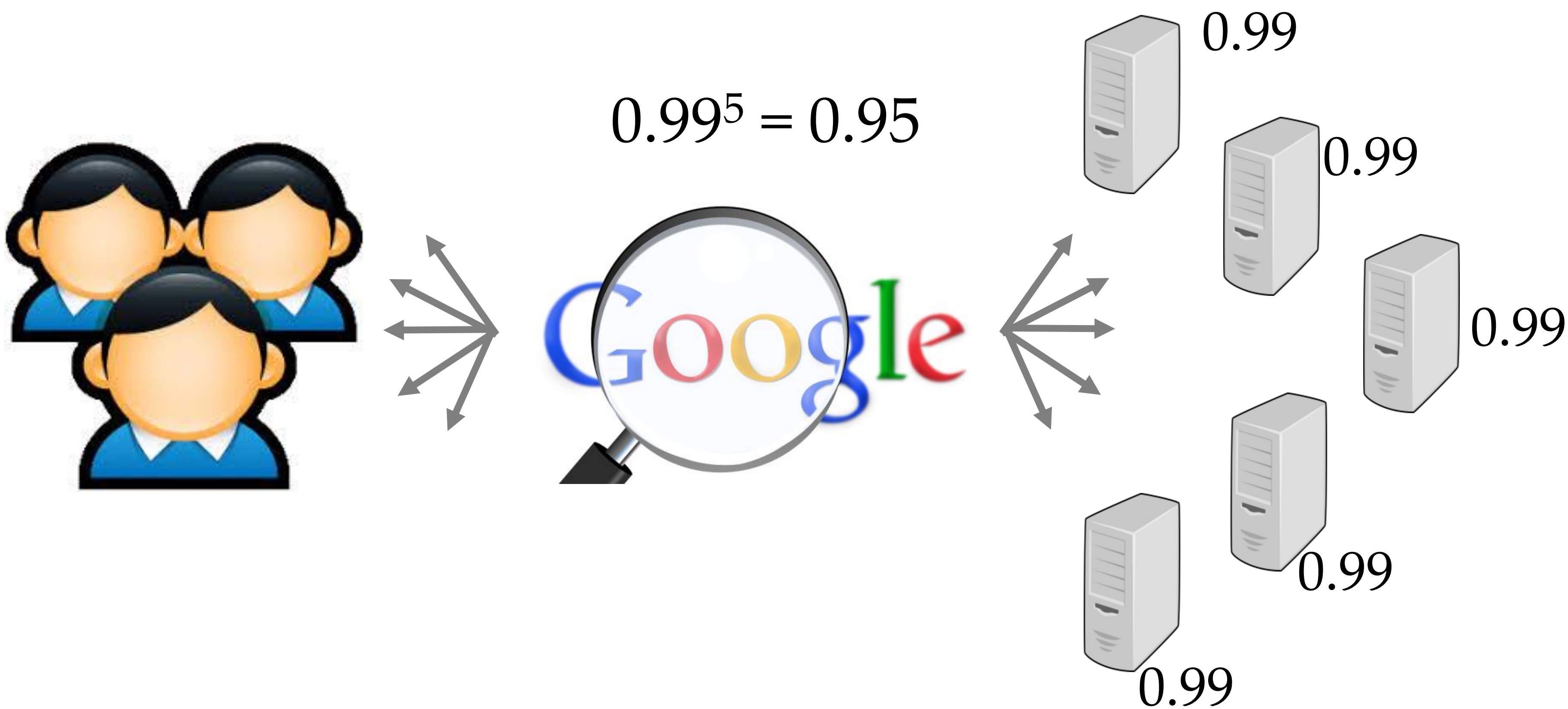
NGINX

MEMCACHED

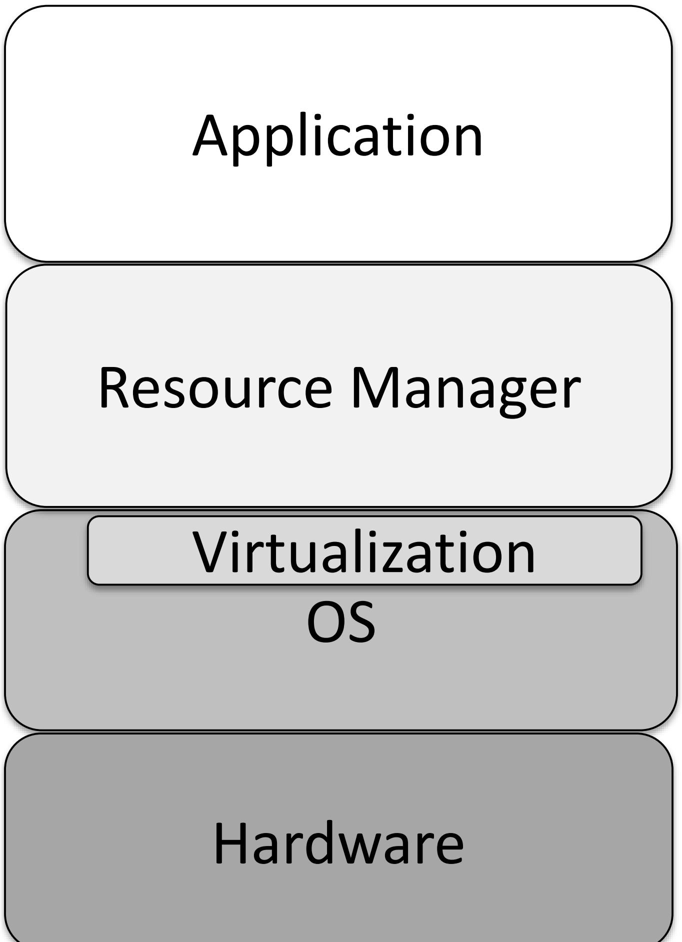
mongoDB



- **Tail latency**
  - e.g., QoS defined as 99^{th} %ile in 500usec



▪ 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

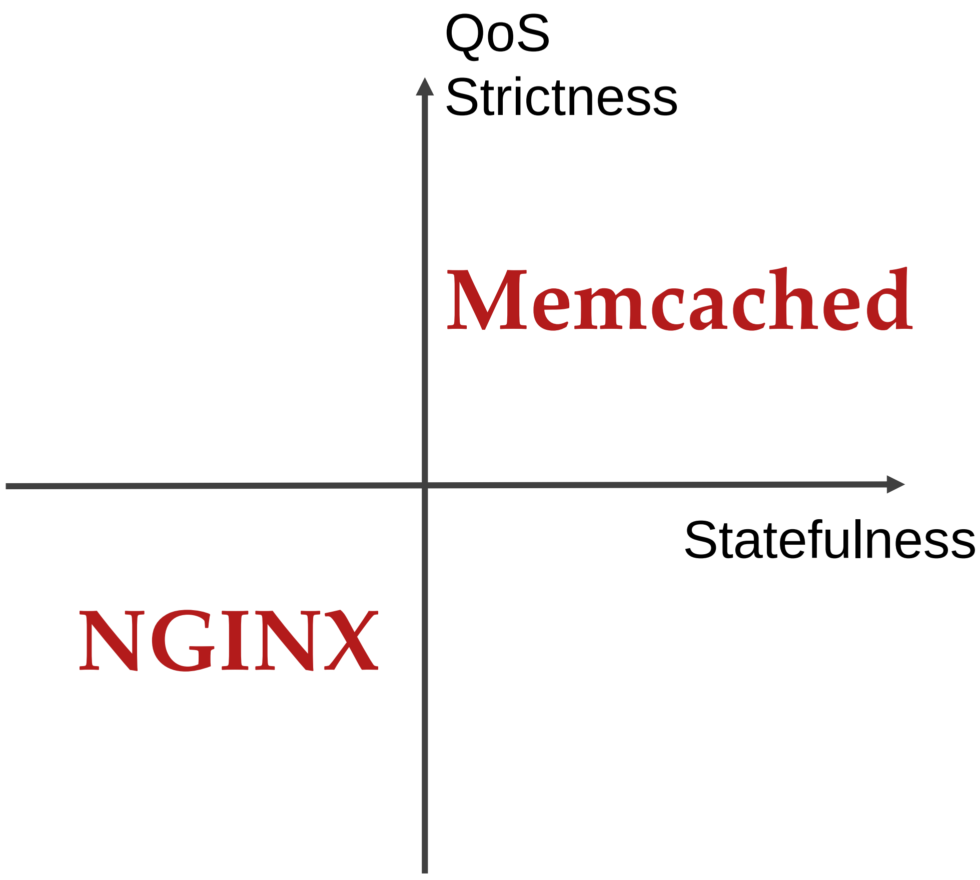
- HW isolation mechanisms
- Hyperthreading

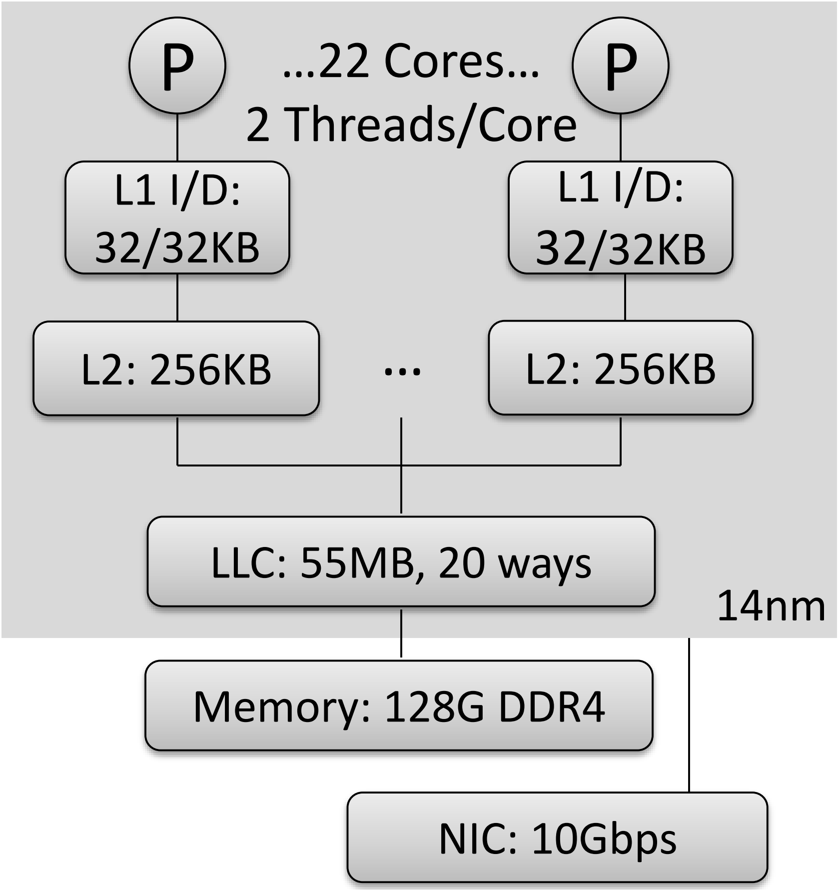
- **By requirement of tail latency**
  - us: memcached
  - ms: web server, in-memory database
  - s: persistent database

- **By statefulness**
  - Stateful: memcached
  - Stateless: web server

SELECTED LC WORKLOADS

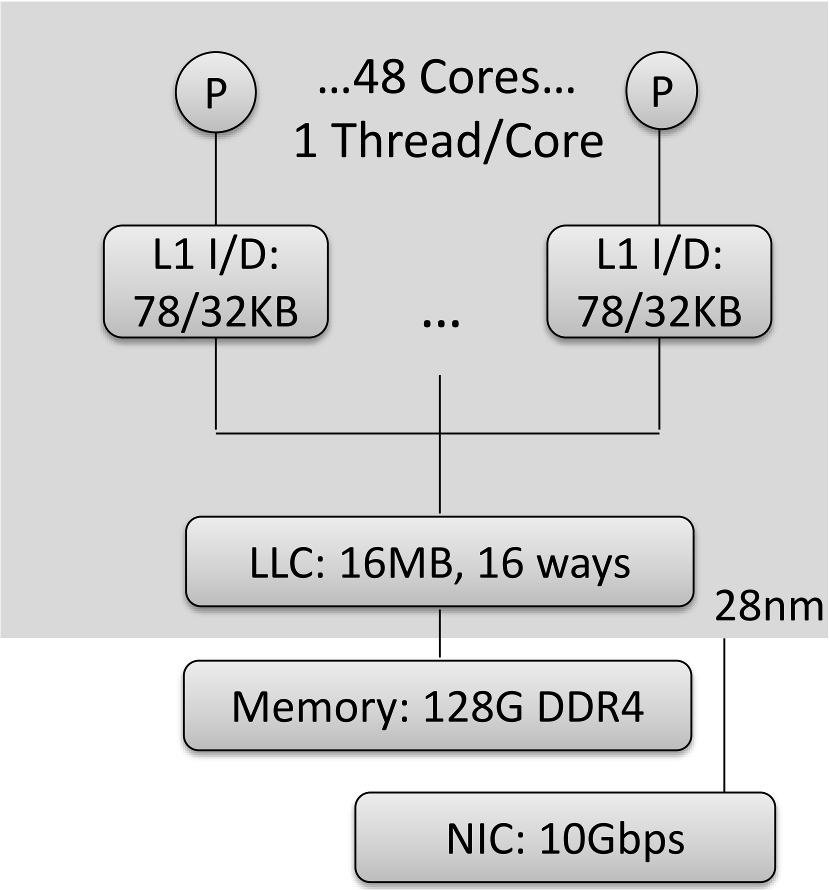
- **NGINX**
  - Web server
  - Stateless
  - 99^{th}% in tens of ms
- **Memcached**
  - Key-value store
  - Stateful
  - 99^{th}% in hundreds of us





Intel Xeon E5-2699 v4

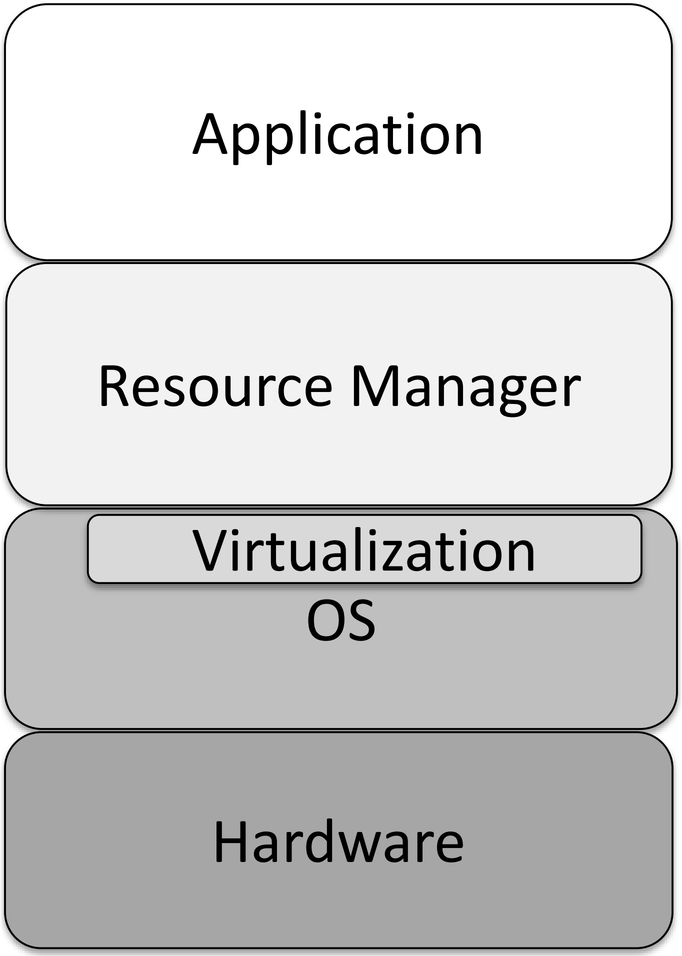
$4,115



Cavium ThunderX

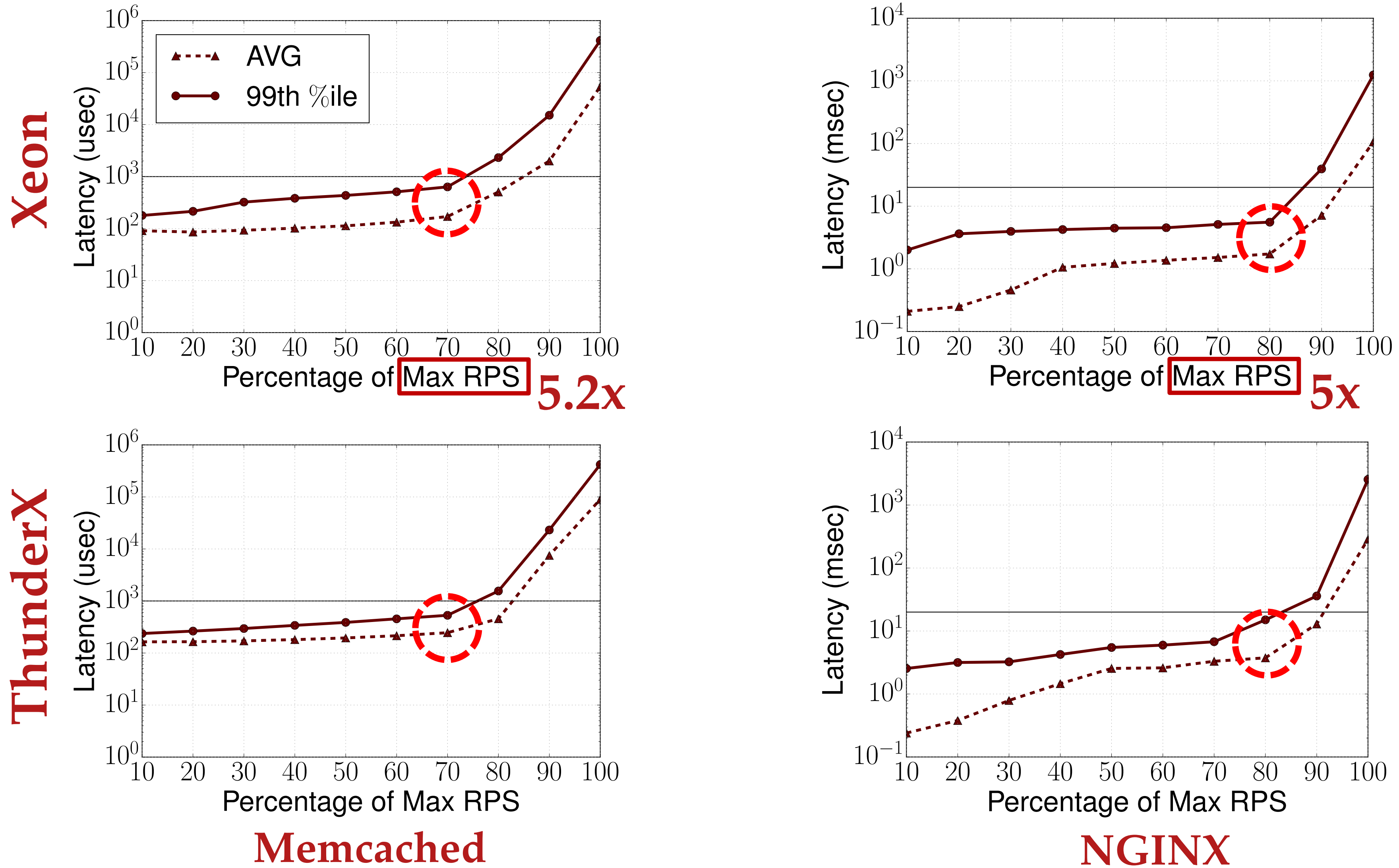
$785

Studied Parameters

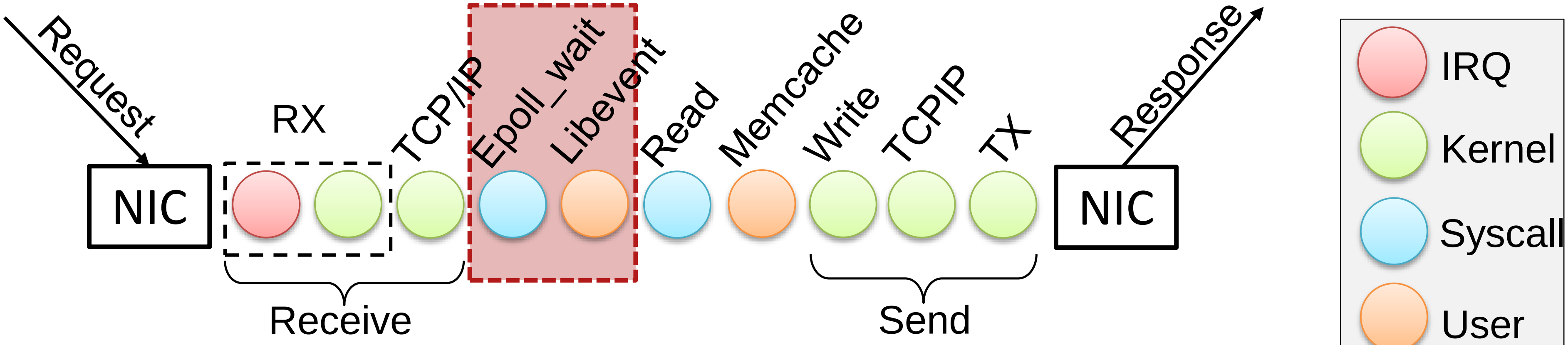


- **Application bottleneck**
- Different user cases
- Scalability

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

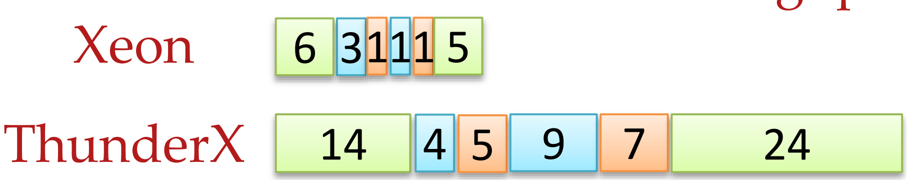

Xeon
ThunderX
AVG
99th %ile
Latency (usec)
Latency (msec)
Percentage of Max RPS
5.2x
5x
Memcached
NGINX

MEMCACHED LATENCY DECOMPOSITION



Little user-space processing

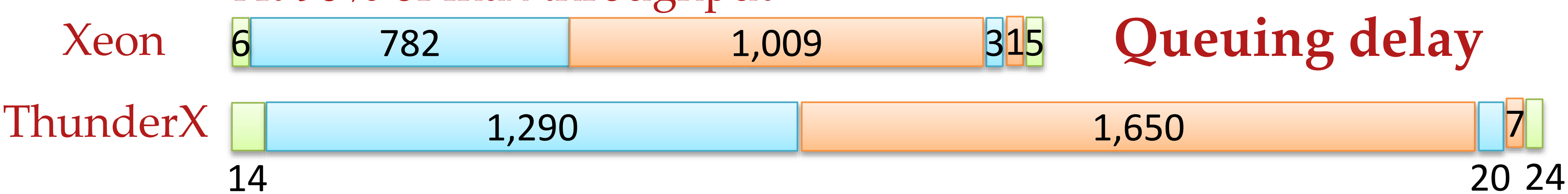
At **10%** of max throughput



Network delay

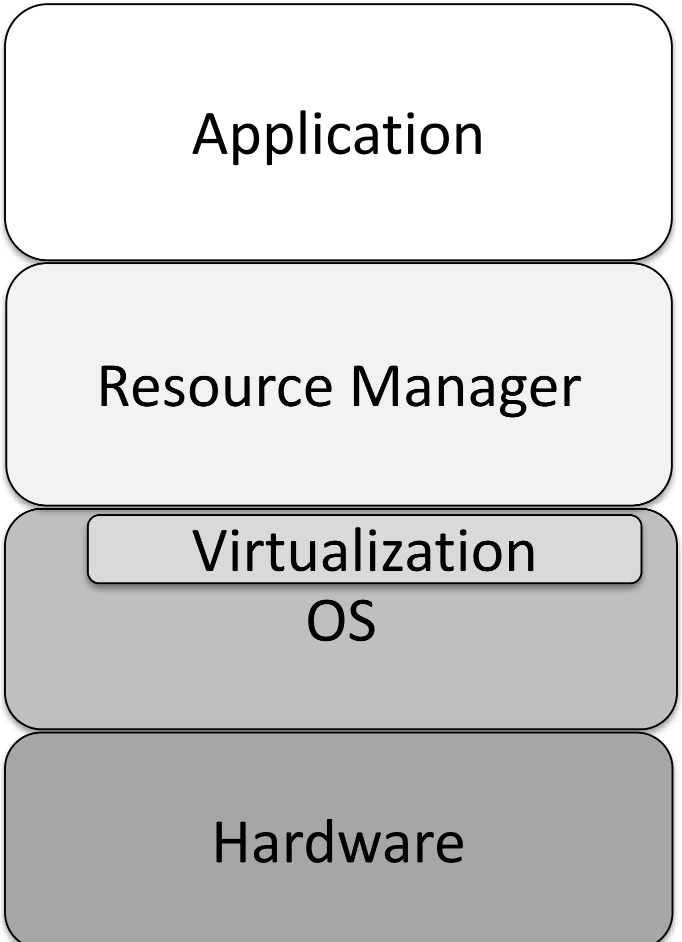
2x slower than Xeon

At **90%** of max throughput



Queuing delay

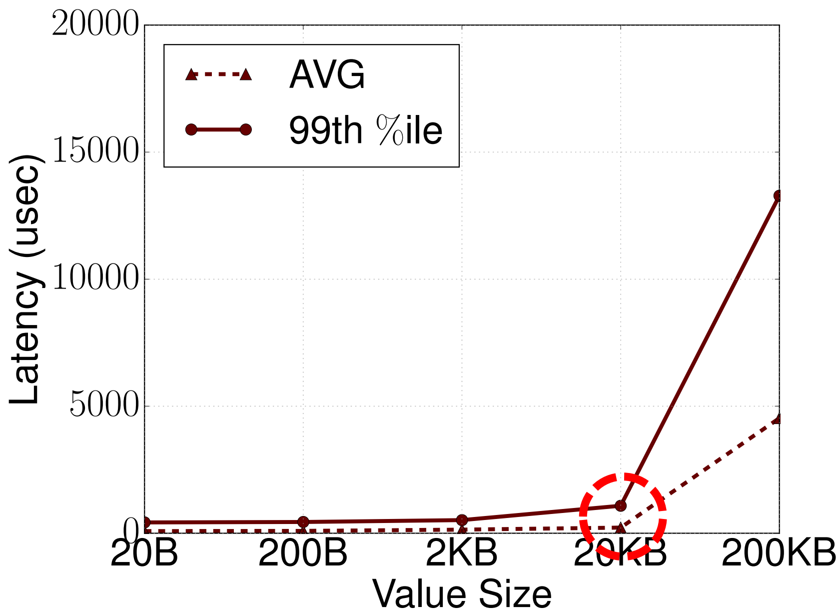
Studied Parameters



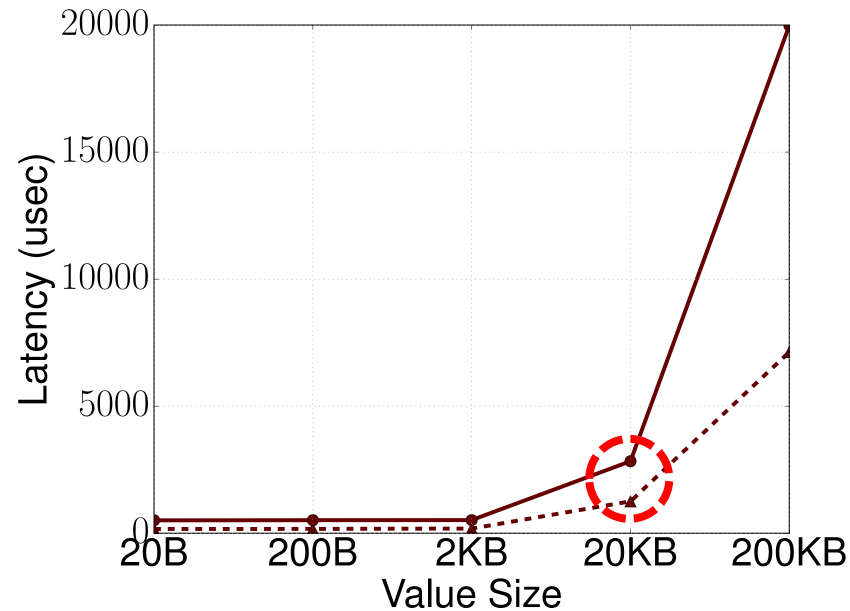
- Application bottleneck
- **Different user cases**
- Scalability

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

Memcached Value Size

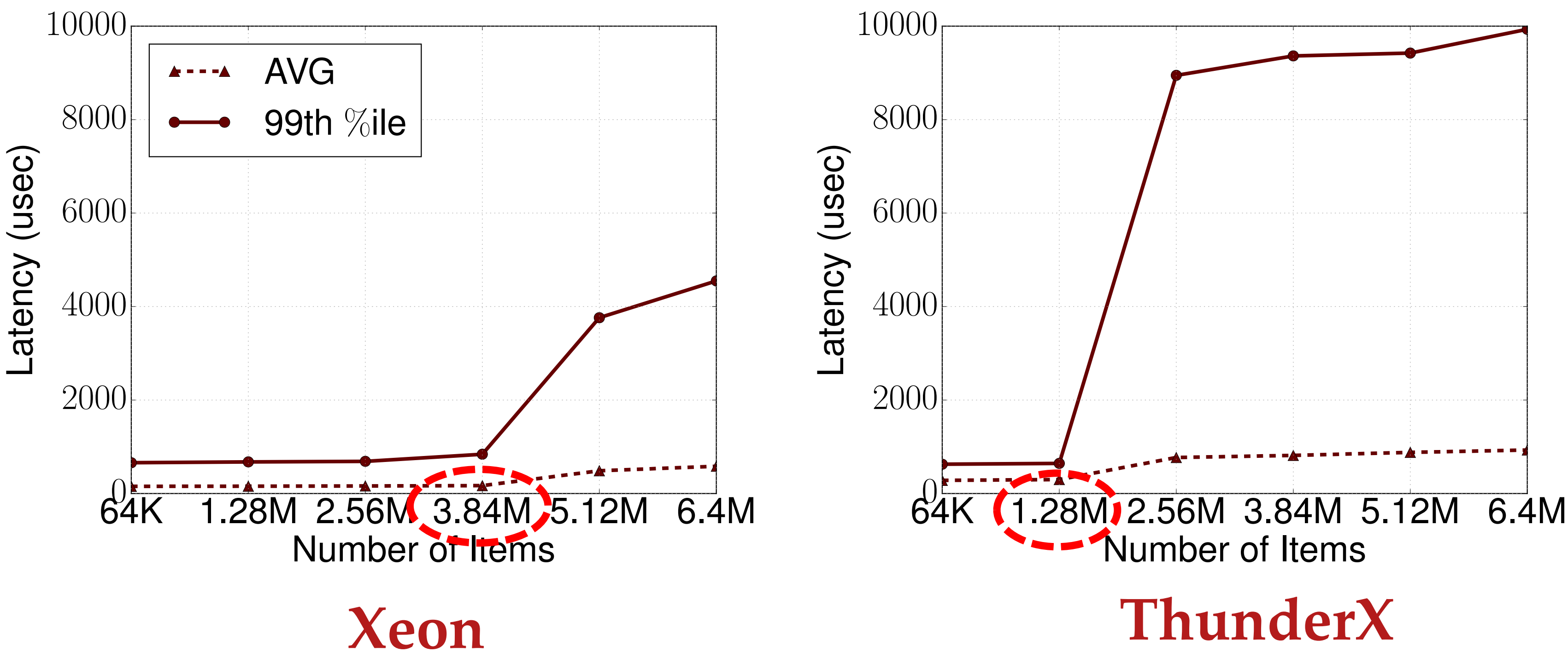


Xeon



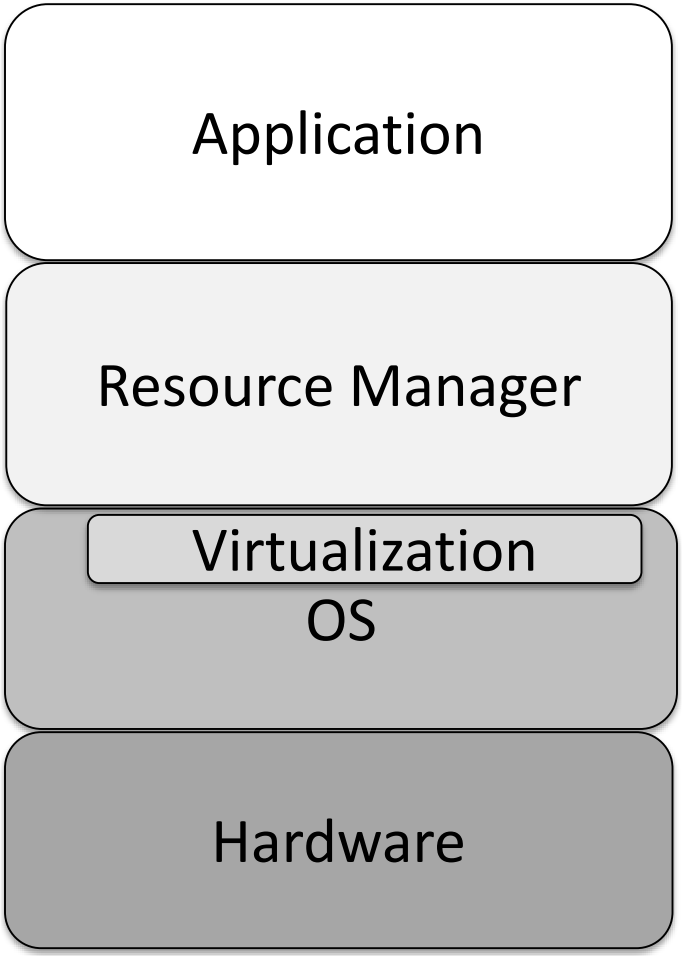
ThunderX

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



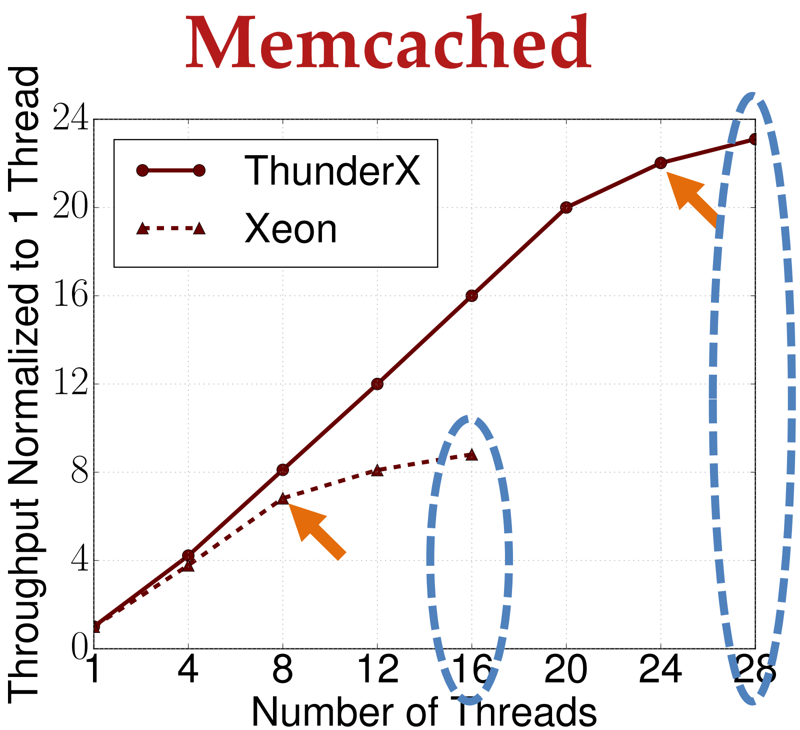
- Cache capacity
- ThunderX is more sensitive

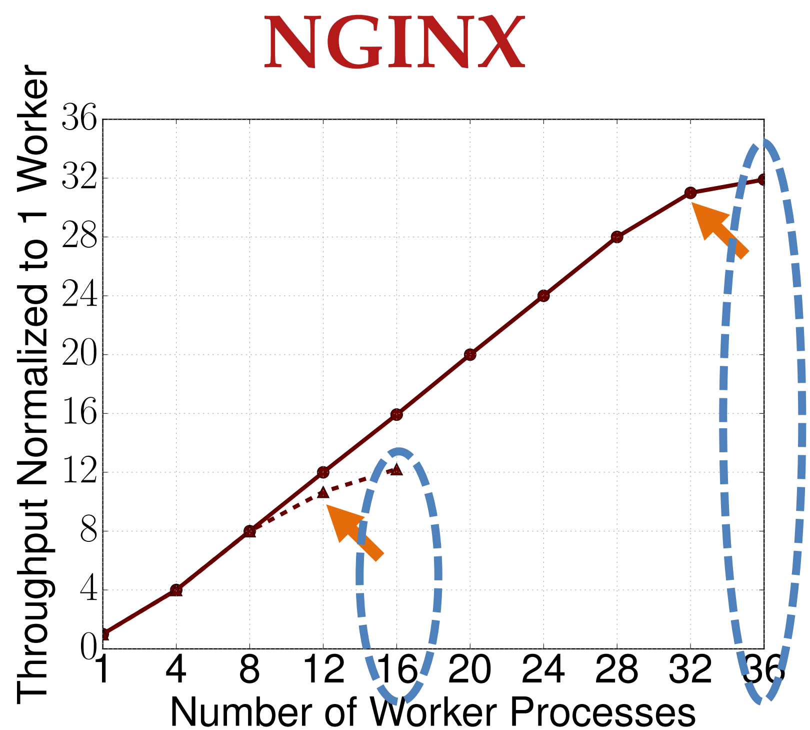
Studied Parameters



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

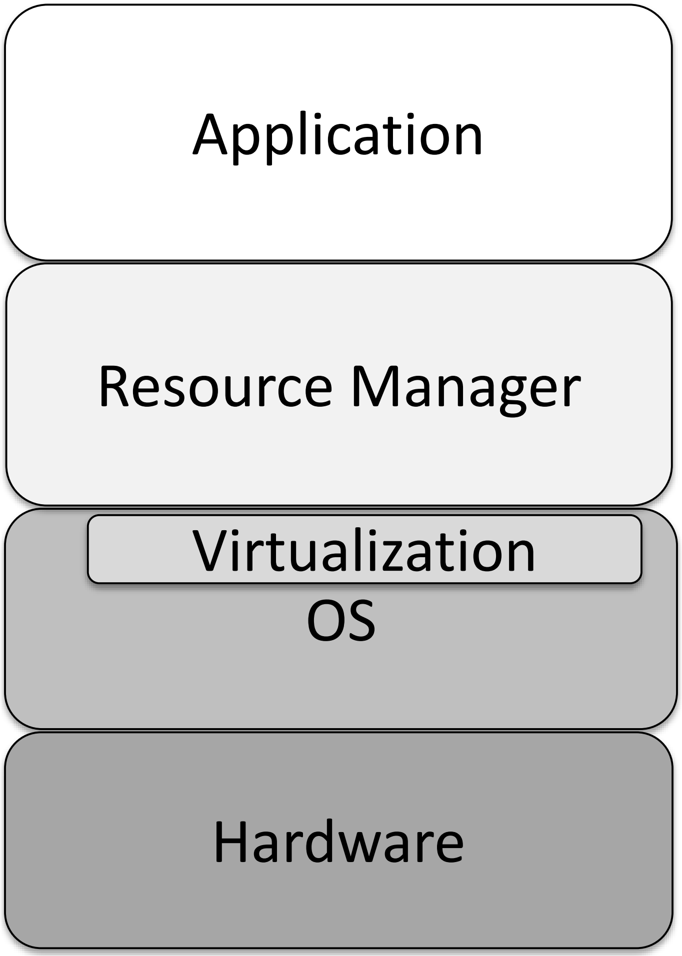
- Overhead of virtualization
- SW isolation mechanisms
- Overhead of context switching
- HW isolation mechanisms
- Hyperthreading





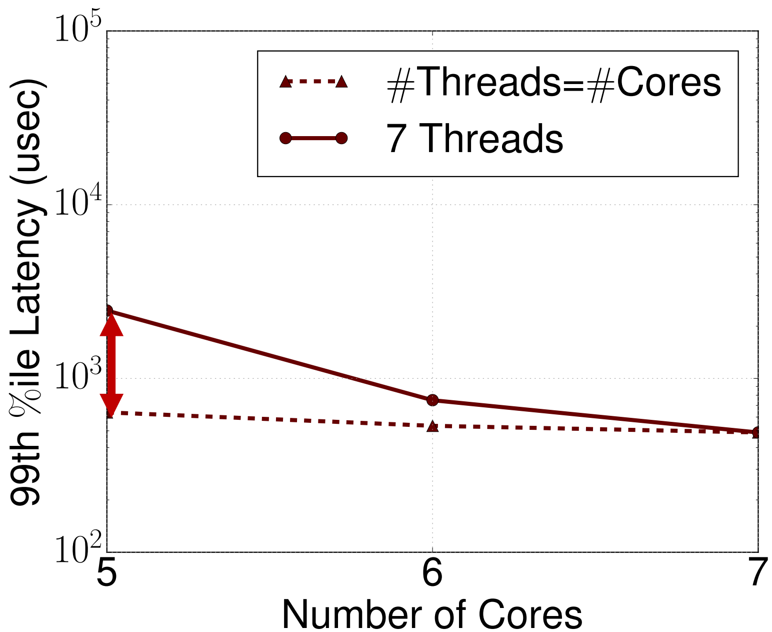
- Interrupt handling
- Load imbalance
- Lock contention

Studied Parameters

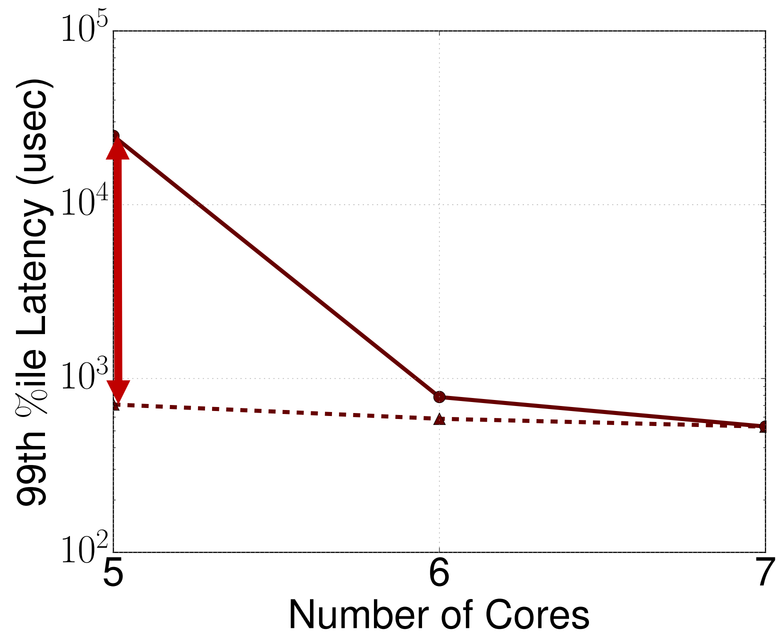


- Application bottleneck
- Different user cases
- Scalability

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



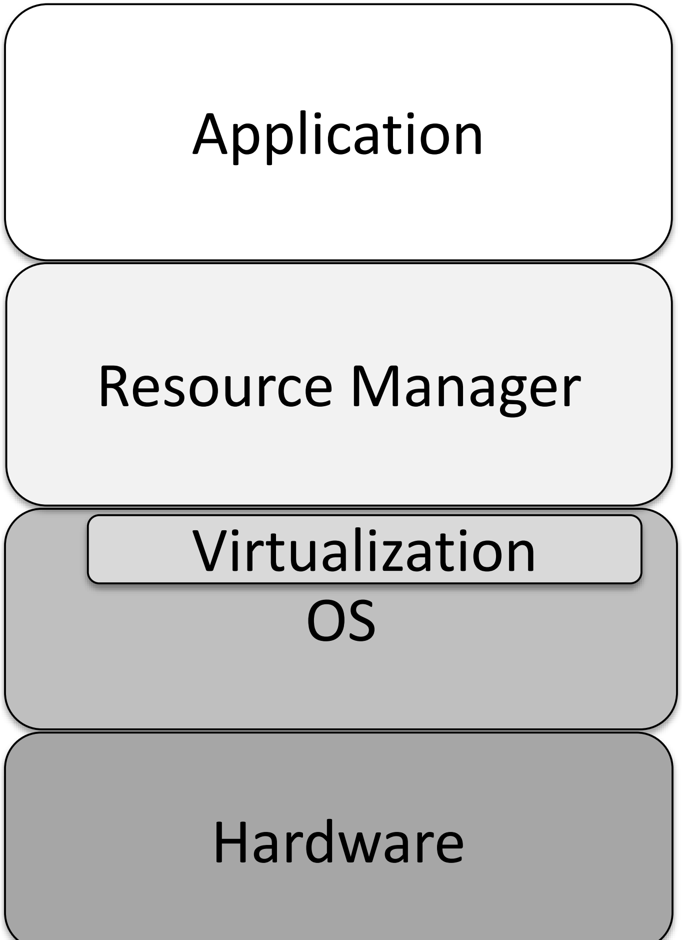
Memcached on Xeon



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**

CSL

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

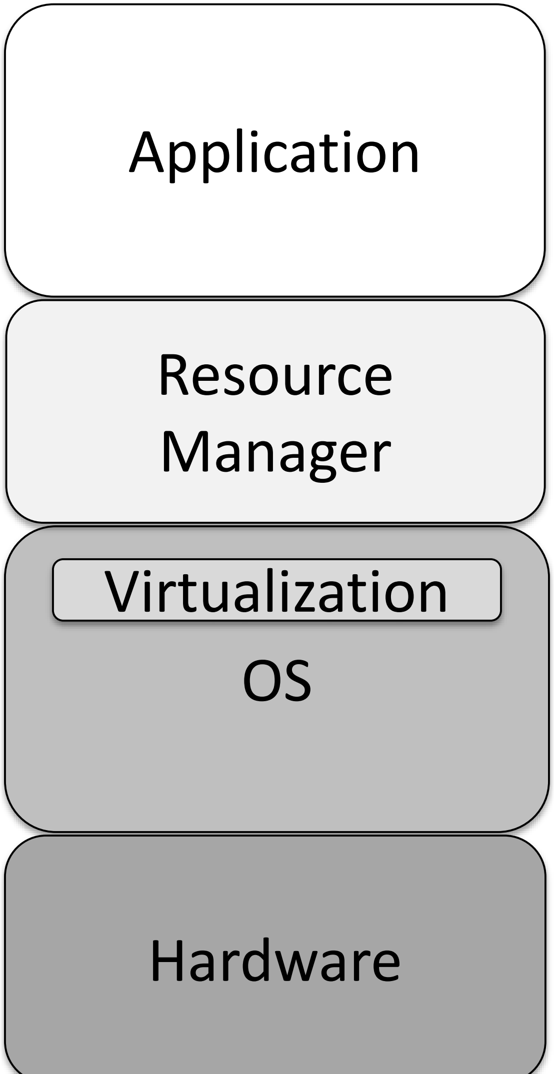
	10%	20%	30%	40%	50%	60%	70%
10%	MN	MN	MN	MN			
20%	MN	MN	MN				
30%	MN	MN					
40%	MN	N					
50%	N	N					
60%	N						
70%							

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 queuing delays
- Improve elasticity
  - Lock alternatives
  - Load balance

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

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