Improving Application Migration to Serverless Computing Platforms:
Latency Mitigation with Keep-Alive Workloads

Minh Vu[#], Baojia Zhang[#], Olaf David, George Leavesley,
Wes Lloyd[1]

December 20, 2018

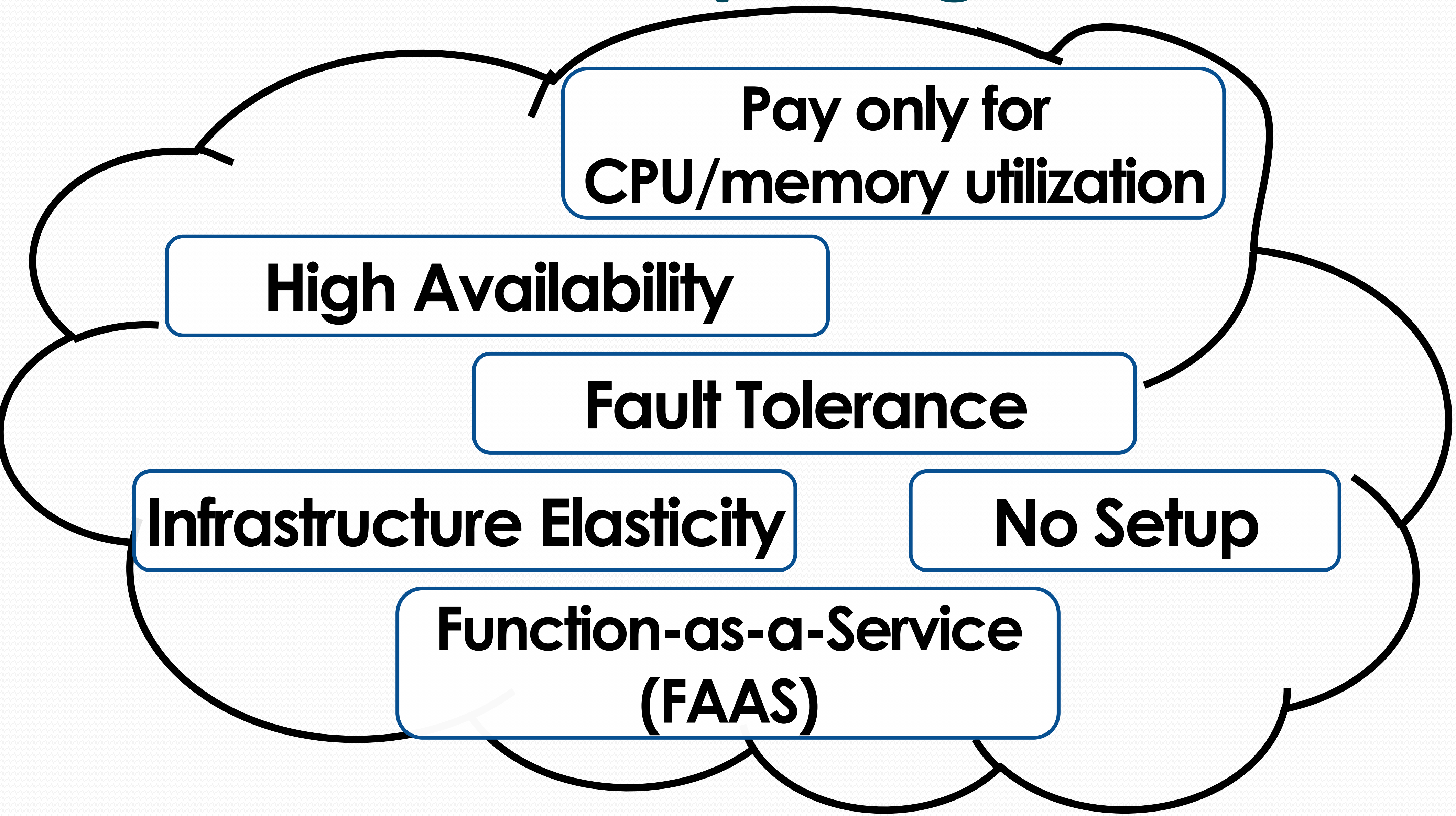
School of Engineering and Technology,
University of Washington, Tacoma, Washington USA

WOSC 2018: 4th IEEE Workshop on Serverless Computing (UCC 2018)

Outline

- Background
- Research Questions
- Experimental Workloads
- Experiments/Evaluation
- Conclusions

Serverless Computing



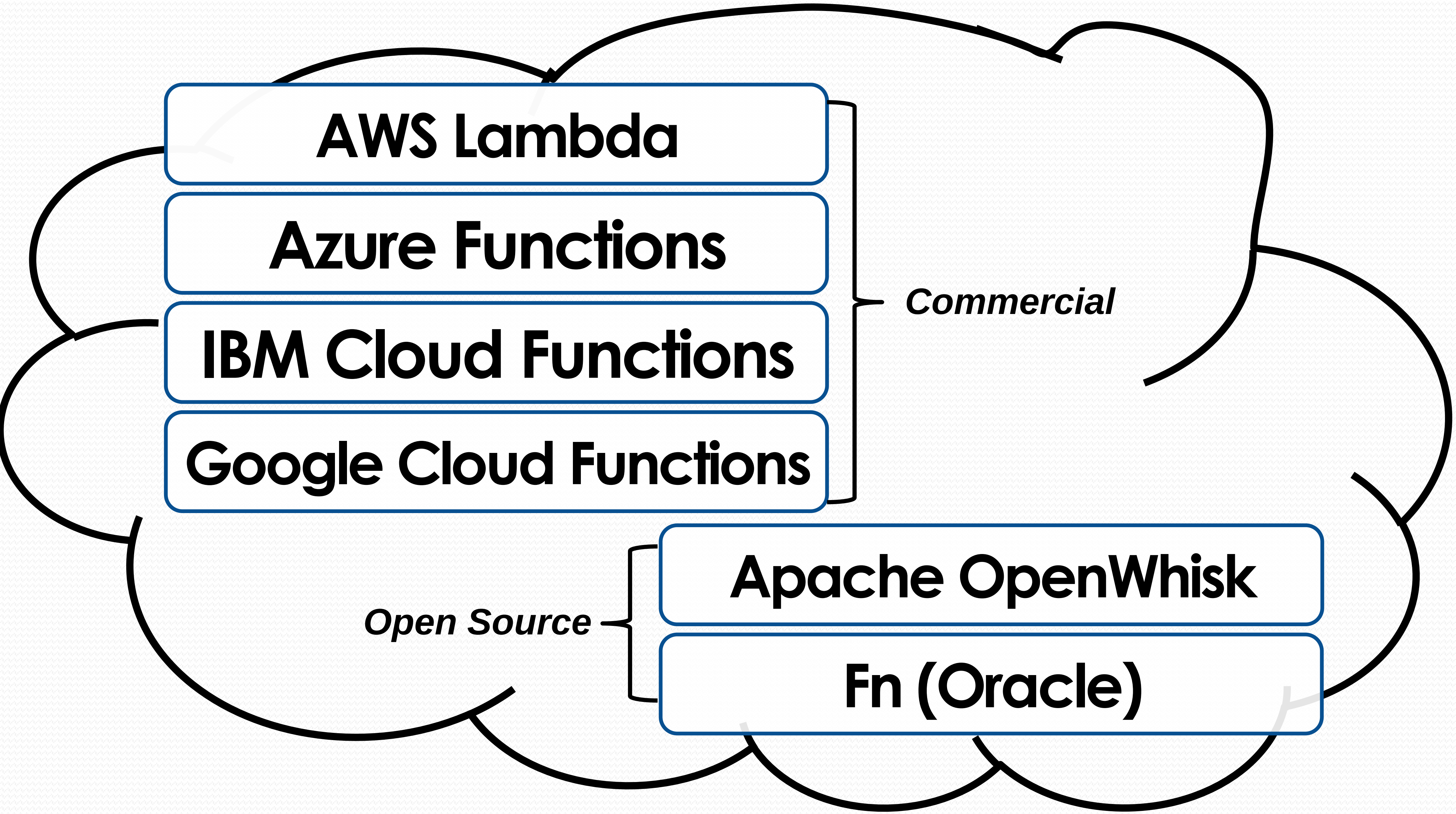
Serverless Computing

Why Serverless Computing?

Many features of distributed systems, that are challenging to deliver, are provided automatically

…they are built into the platform

Serverless Platforms



Serverless Computing

Research Challenges

Image from: https://mobisoftinfotech.com/resources/blog/serverless-computing-deploy-applications-without-fiddling-with-servers/

Serverless Computing Research Challenges

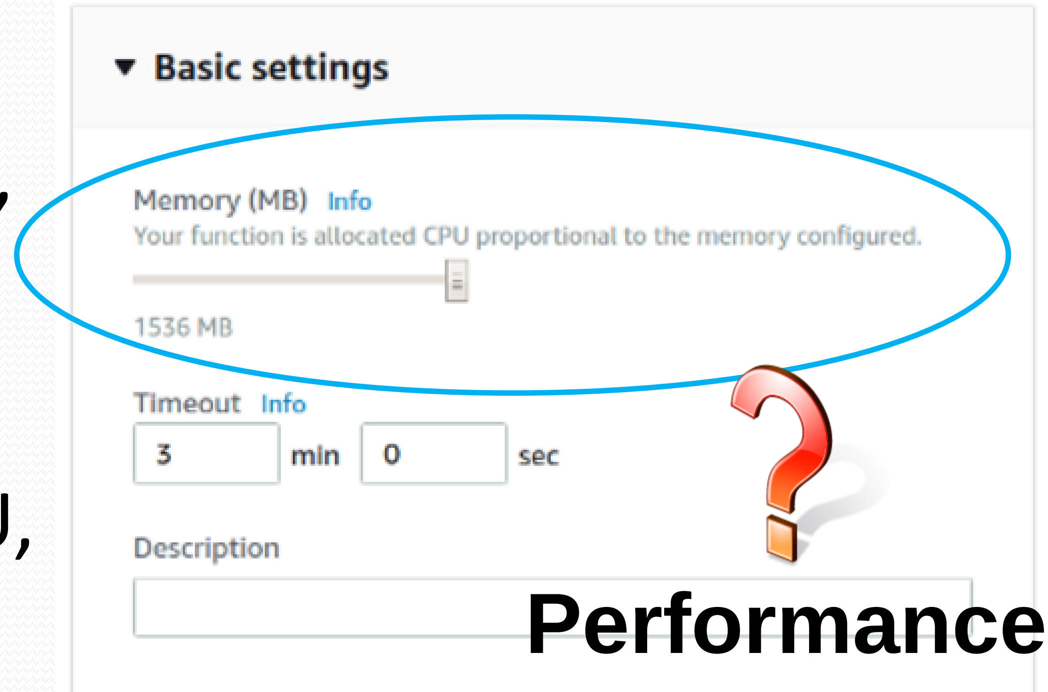
- Memory reservation
- Infrastructure freeze/thaw cycle
- Vendor architectural lock-in
- Pricing obfuscation
- Service composition

Serverless Computing Research Challenges

- Memory reservation
- Infrastructure freeze/thaw cycle
- Vendor architectural lock-in
- Pricing obfuscation
- Service composition

Memory Reservation Question…

- Lambda memory reserved for functions
- UI provides "slider bar" to set function's memory allocation
- Resource capacity (CPU, disk, network) coupled to slider bar:
  *"every **doubling** of memory, **doubles** CPU…"*



- **But how much memory do model services require?**

Infrastructure Freeze/Thaw Cycle

Performance

- Unused infrastructure is deprecated
  - ***But after how long?***
- AWS Lambda: Bare-metal hosts, firecracker micro-VMs
- Infrastructure states: https://firecracker-microvm.github.io/
- **<u>Provider-COLD / Host-COLD</u>**
  - Function package built/transferred to Hosts
- **<u>Container-COLD (firecracker micro-VM)</u>**
  - Image cached on Host
- **<u>Container-WARM (firecracker micro-VM)</u>**
  - “Container” running on Host

Image from: Denver7 – The Denver Channel News

Outline

- Background
- Research Questions
- Experimental Workloads
- Experiments/Evaluation
- Conclusions

Research Questions

RQ1: **PERFORMANCE:** What are the performance implications for application migration? How does memory reservation size impact performance when coupled to CPU power?

RQ2: **SCALABILITY:** For application migration what performance implications result from scaling the number of concurrent clients? How is scaling affected when infrastructure is allowed to go cold?

Research Questions - 2

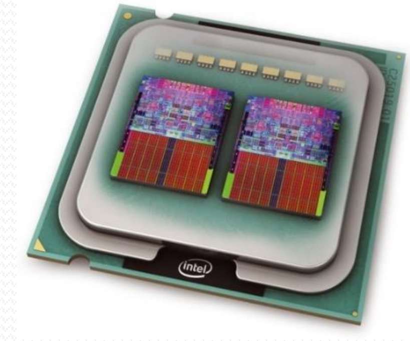
RQ3: **<u>COST:</u>** For hosting large parallel service workloads, how does memory reservation size, impact hosting costs when coupled to CPU power?

RQ4: **<u>PERSISTING INFRSASTRUCTURE:</u>** How effective are automatic triggers at retaining serverless infrastructure to reduce performance latency from the serverless freeze/thaw cycle?

Outline

- Background
- Research Questions
- Experimental Workloads
- Experiments/Evaluation
- Conclusions

AWS Lambda
PRMS Modeling Service



- PRMS: deterministic, distributed-parameter model
- Evaluate impact of combinations of precipitation, climate, and land use on stream flow and general basin hydrology (Leavesley et al., 1983)

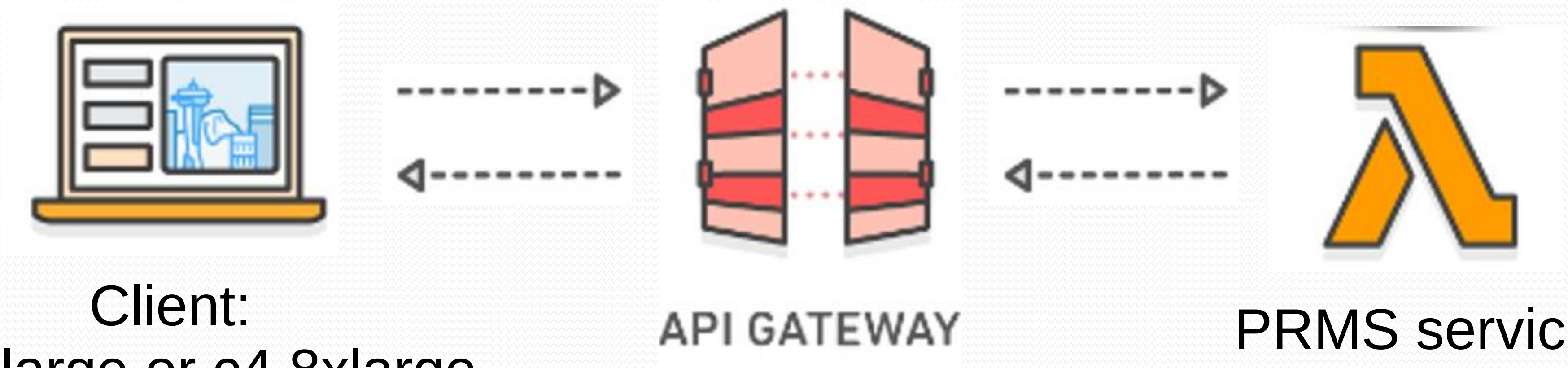
- Java based PRMS, Object Modelling System (OMS) 3.0
- Approximately ~11,000 lines of code
- Model service is 18.35 MB compressed as a Java JAR file
- Data files hosted using Amazon S3 (object storage)

Goal: quantify performance and cost implications of <u>*memory reservation size*</u> and <u>*scaling*</u> for model service deployment to AWS Lambda

PRMS Lambda Testing

REST/JSON

Images credit: aws.amazon.com



Client:
c4.2xlarge or c4.8xlarge
(8 core) (36 core)

BASH: GNU Parallel
Multi-thread client script
"partest"

Up to 100 concurrent synchronous requests

Results of each thread traced individually

<u>Fixed-availability zone:</u>
EC2 client / Lambda server
us-east-1e

PRMS service

Max
service duration:
< 30 seconds

Memory:
256 to 3008MB

PRMS Lambda Testing - 2

REST/JSON

Images credit: aws.amazon.com

Client:
c4.2xlarge or c4.8xlarge
(8 core) (36 core)

PRMS service

Automatic Metrics Collection[1]:

New vs. Recycled Containers/VMs

# of requests per container/VM

Avg. performance per container/VM

Avg. performance workload

Standard deviation of requests per container/VM

Container Identification
UUID → /tmp file

VM Identification
btime → /proc/stat

Linux CPU metrics

[1] Lloyd, W., Ramesh, S., Chinthalapati, S., Ly, L., & Pallickara, S. (April 2018). Serverless computing: An investigation of factors influencing microservice performance. In Cloud Engineering (IC2E), 2018 IEEE International Conference on (pp. 159-169). IEEE.

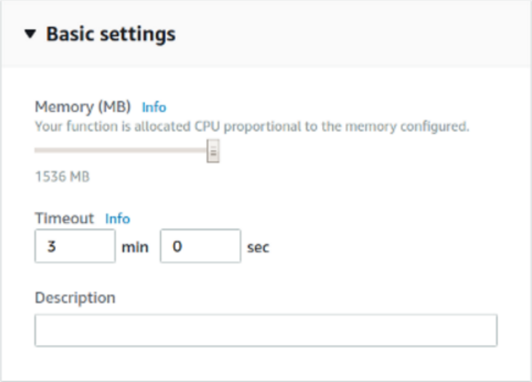
Outline

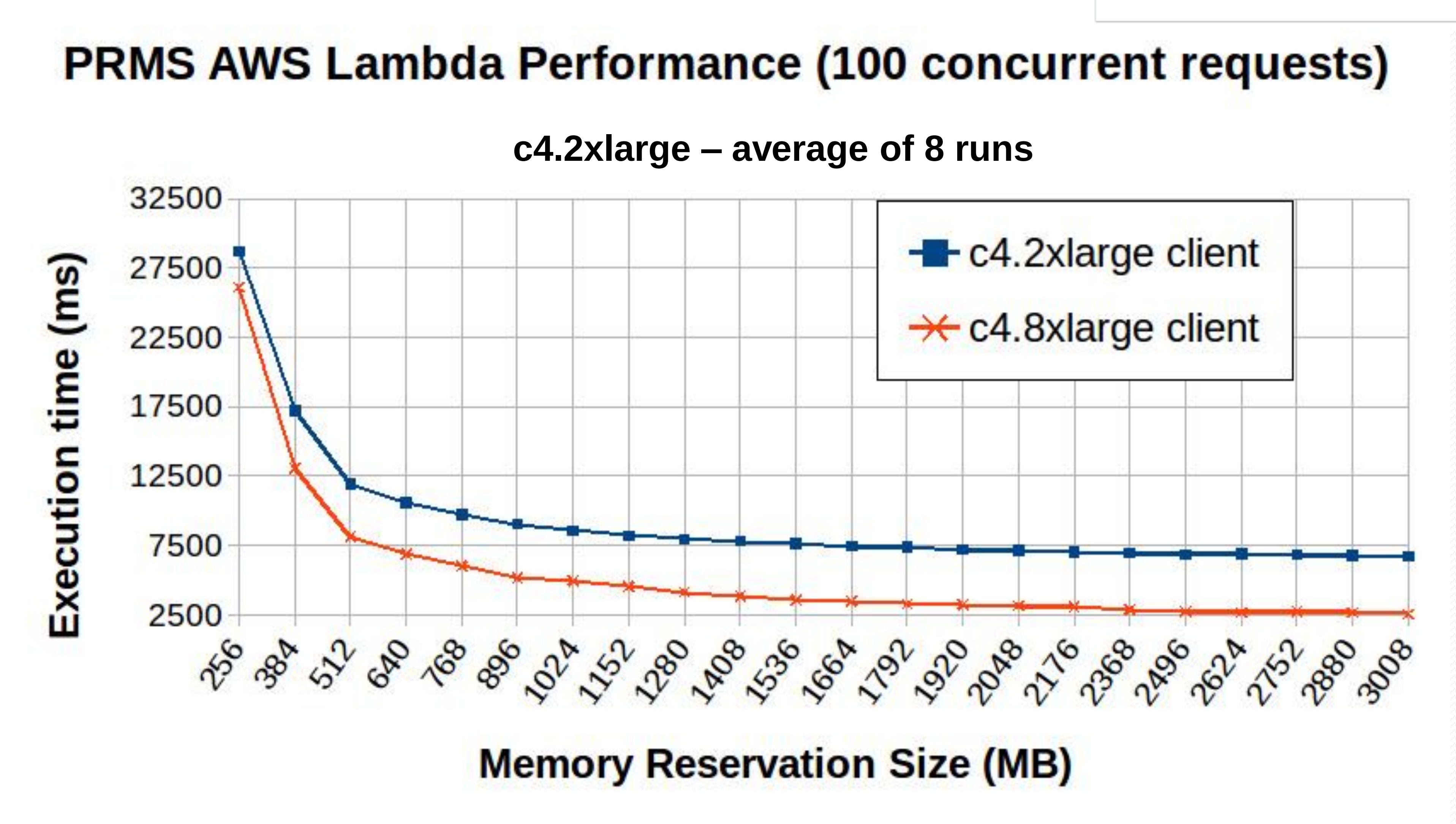
- Background
- Research Questions
- Experimental Workloads
- Experiments/Evaluation
- Conclusions

RQ-1: Performance

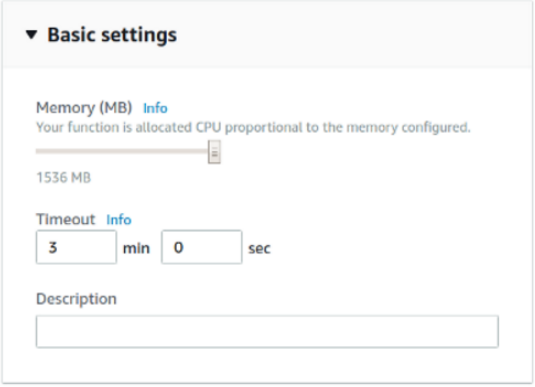
Infrastructure
What are the performance implications of memory reservation size ?

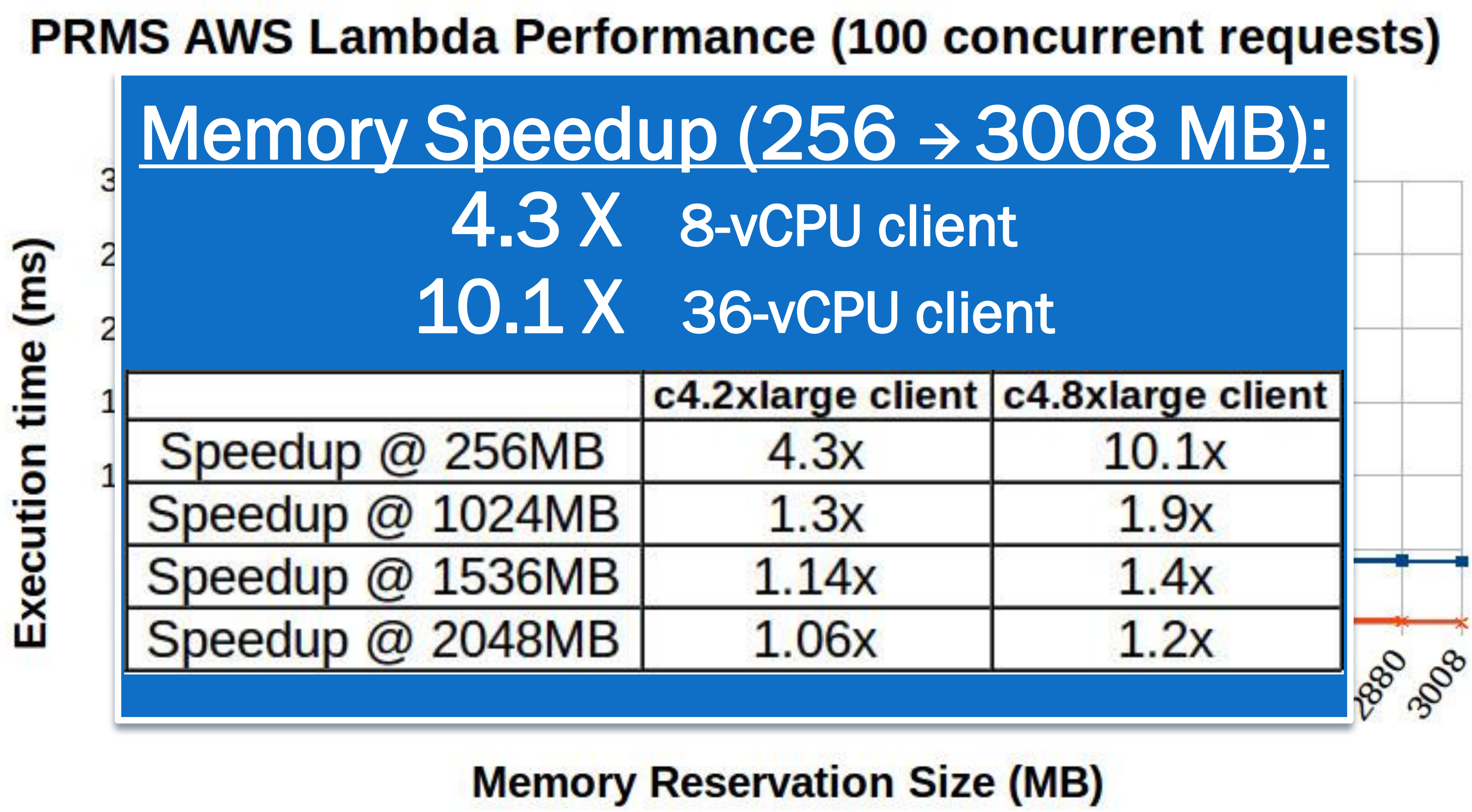
RQ-1: AWS Lambda Memory Reservation Size





RQ-1: AWS Lambda Memory Reservation Size





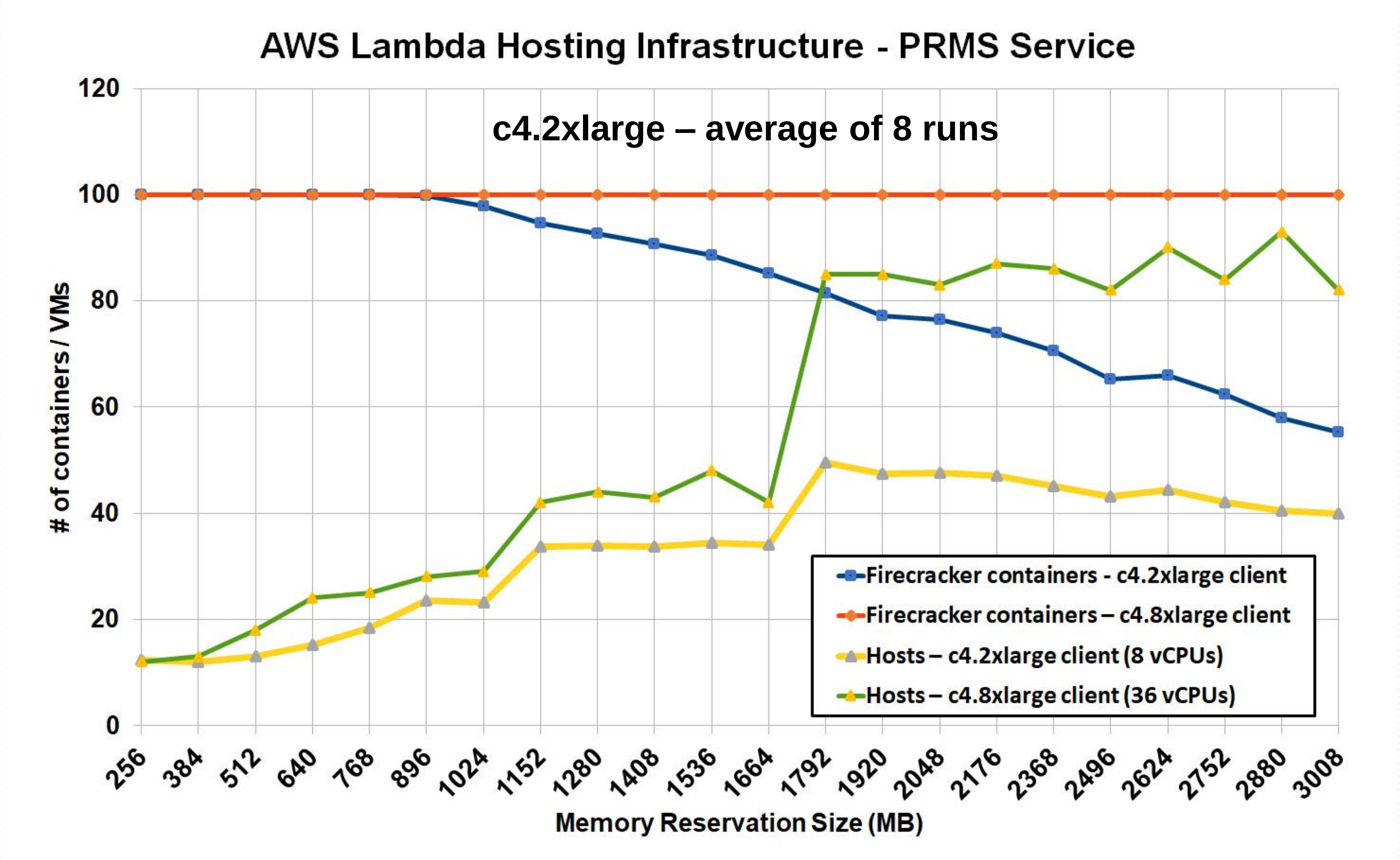
Memory Speedup (256 → 3008 MB):

4.3 X 8-vCPU client

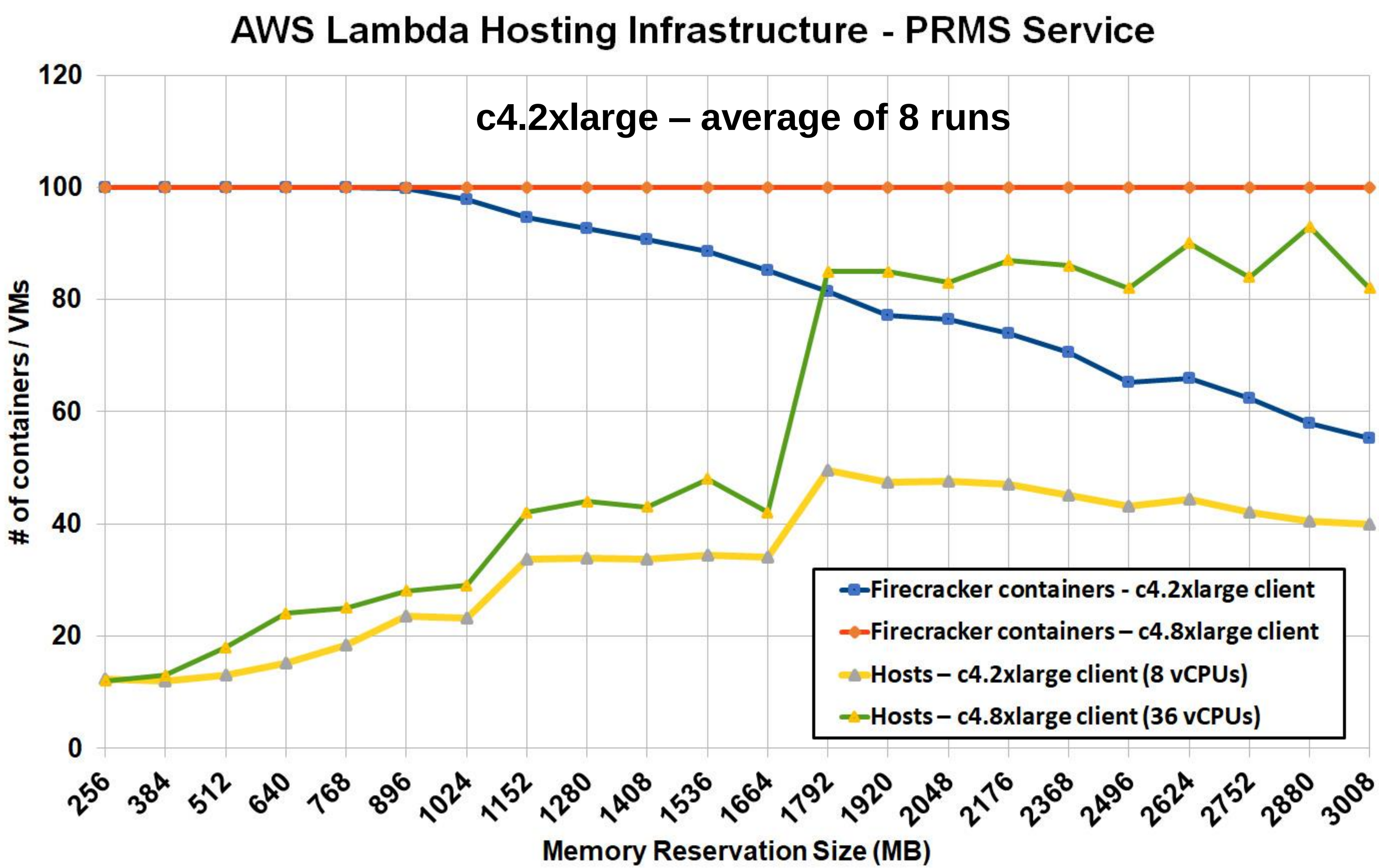
10.1 X 36-vCPU client

	c4.2xlarge client	c4.8xlarge client
Speedup @ 256MB	4.3x	10.1x
Speedup @ 1024MB	1.3x	1.9x
Speedup @ 1536MB	1.14x	1.4x
Speedup @ 2048MB	1.06x	1.2x

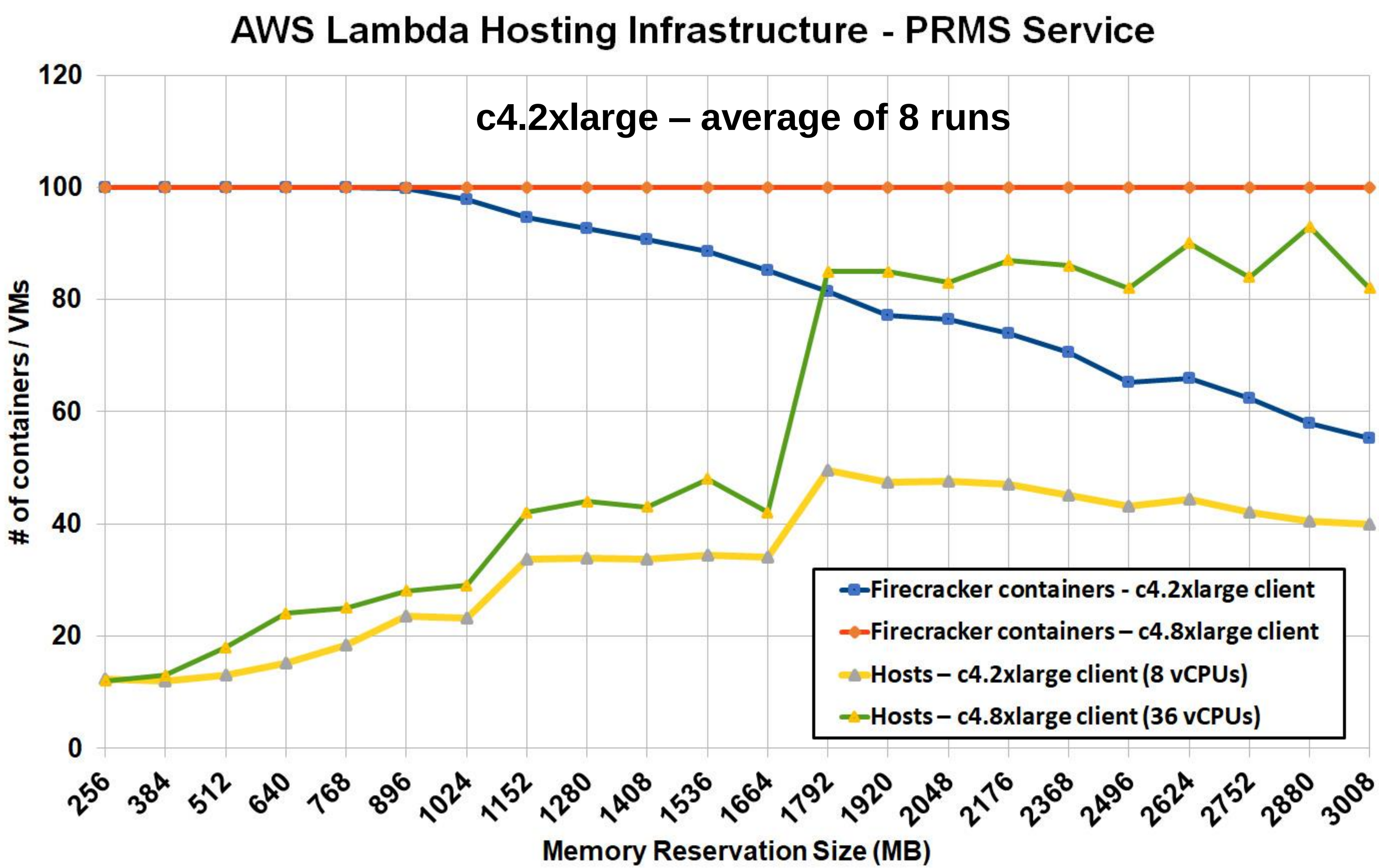
RQ-1: AWS Lambda Memory Reservation Size - Infrastructure



Many more Hosts leveraged when memory > 1536 MB



8 vCPU client struggles to generate 100 concurrent requests >= 1024MB

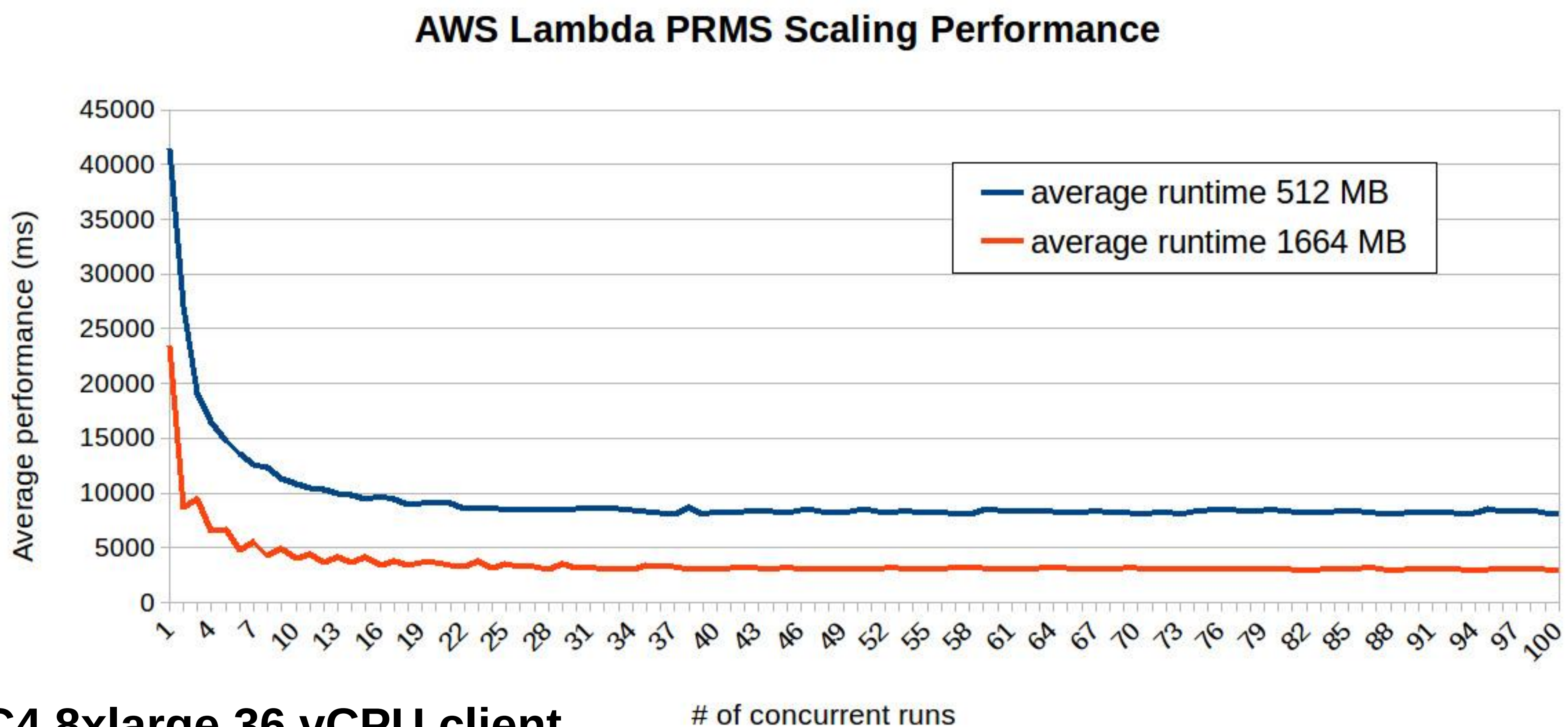


RQ-2: Scalability

How does performance change when increasing the number of concurrent users ?

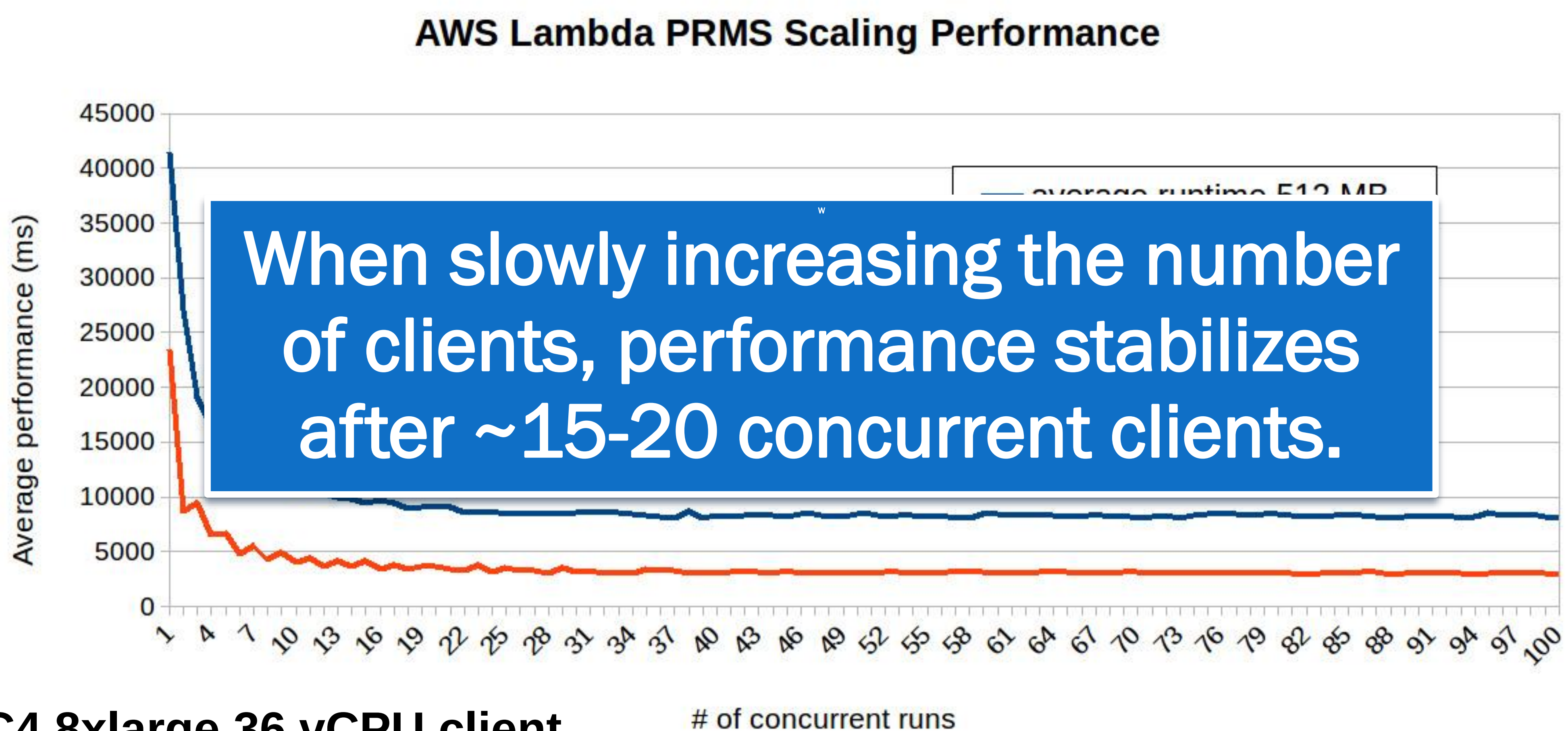
(scaling-up, totally cold, and warm)

RQ-2: AWS Lambda PRMS Scaling Performance



C4.8xlarge 36 vCPU client

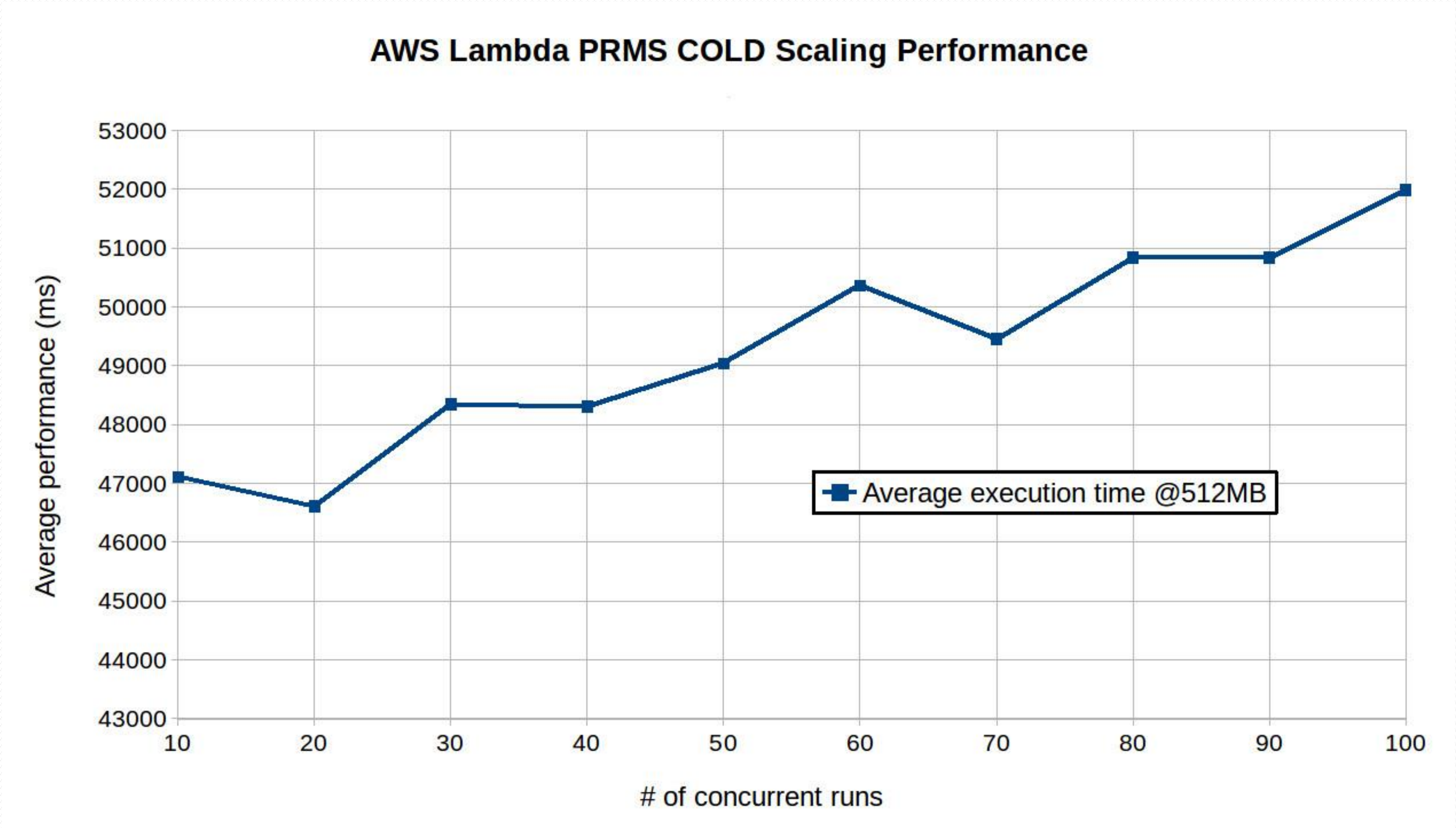
RQ-2: AWS Lambda PRMS Scaling Performance



When slowly increasing the number of clients, performance stabilizes after ~15-20 concurrent clients.

C4.8xlarge 36 vCPU client

RQ-2: AWS Lambda Cold Scaling Performance



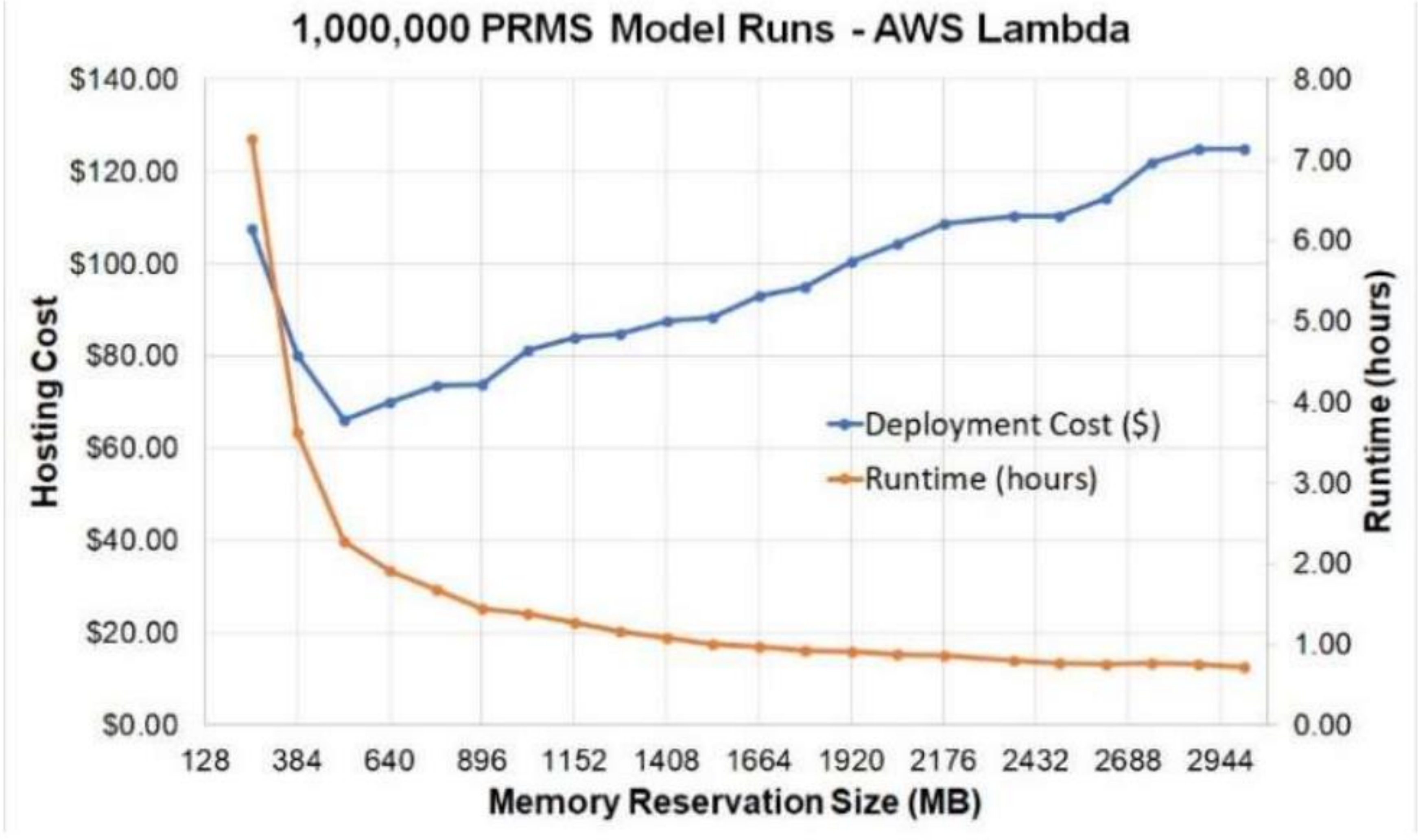
RQ-3: Cost

What are the costs of hosting PRMS using a FaaS platform in comparison to IaaS?

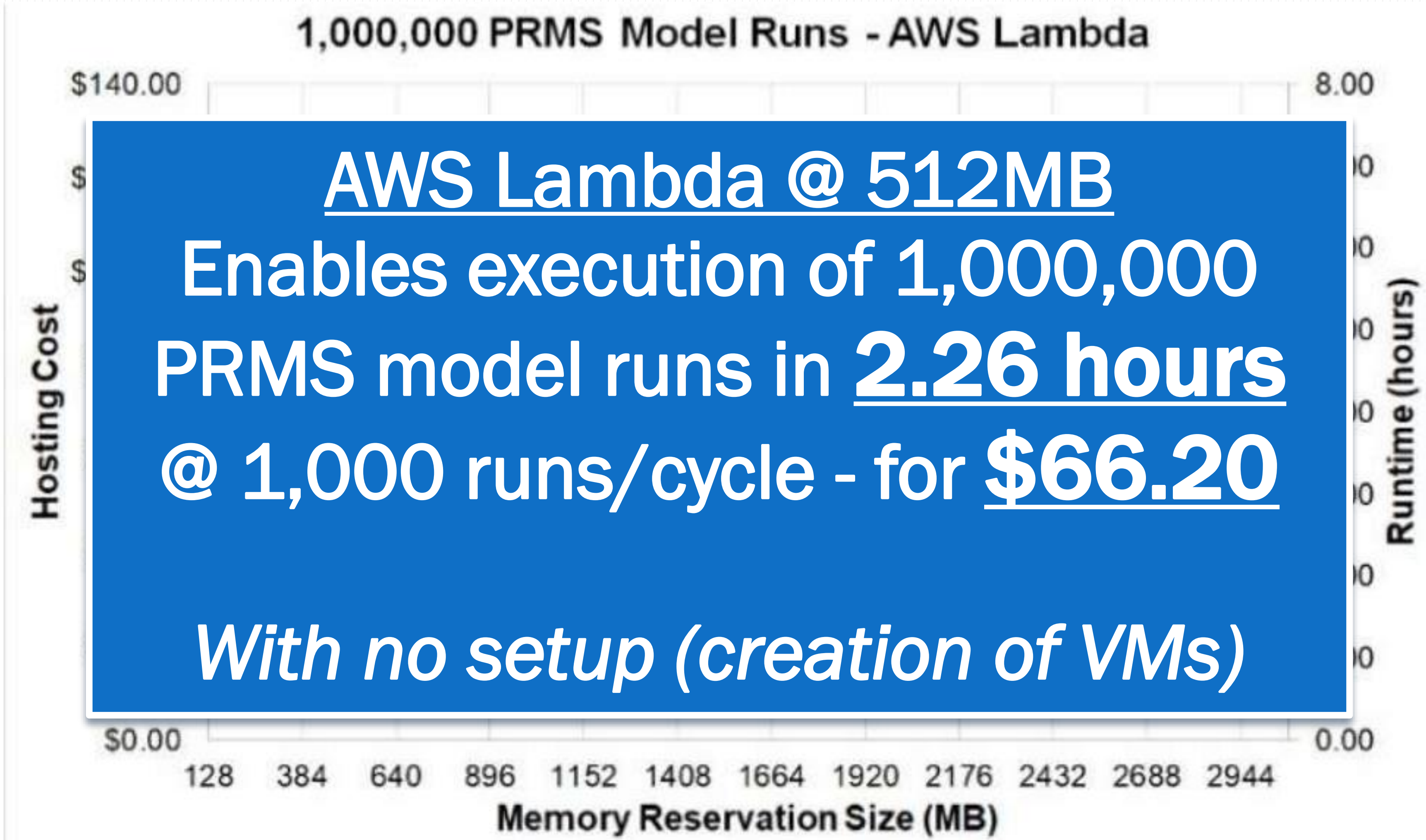
RQ-3: IaaS (EC2) Hosting Cost 1,000,000 PRMS runs

- Using a 2 vCPU c4.large EC2 VM
  - 2 concurrent client calls, no scale-up
- Estimated time: 347.2 hours, **14.46 days**
  - Assume average exe time of 2.5 sec/run
- Hosting cost @ 10¢/hour = **$34.72**

RQ-3: FaaS Hosting Cost 1,000,000 PRMS runs



RQ-3: FaaS Hosting Cost 1,000,000 PRMS runs



AWS Lambda @ 512MB

Enables execution of 1,000,000 PRMS model runs in **2.26 hours** @ 1,000 runs/cycle - for **$66.20**

With no setup (creation of VMs)

RQ-4: Persisting Infrastructure

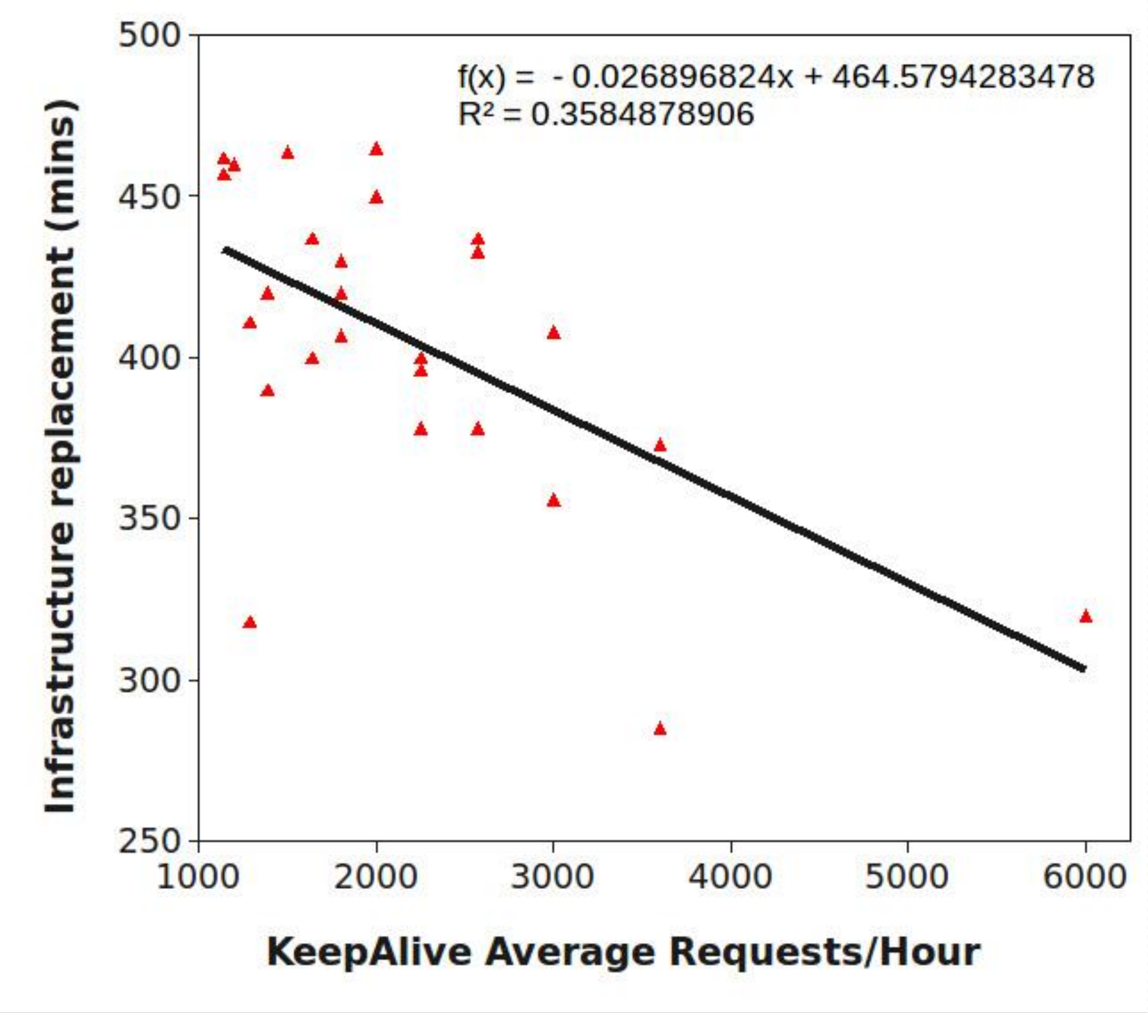
How effective are automatic triggers at retaining serverless infrastructure to reduce performance latency from the serverless freeze/thaw cycle?

RQ-4: Persisting Infrastructure

- Goal: preserve 100 firecracker containers for 24hrs
  - Mitigate cold start latency
- Memory: 192, 256, 384, 512 MB
- All initial host infrastructure replaced between ~4.75 – 7.75 hrs
- Replacement cycle (start→finish): ~2 hrs
- Infrastructure generations performance variance observed from: -14.7% to 19.4% (Δ 34%)
- Average performance variance larger for lower memory sizes: 9% (192MB), 3.6% (512MB)

RQ-4: Persisting Infrastructure

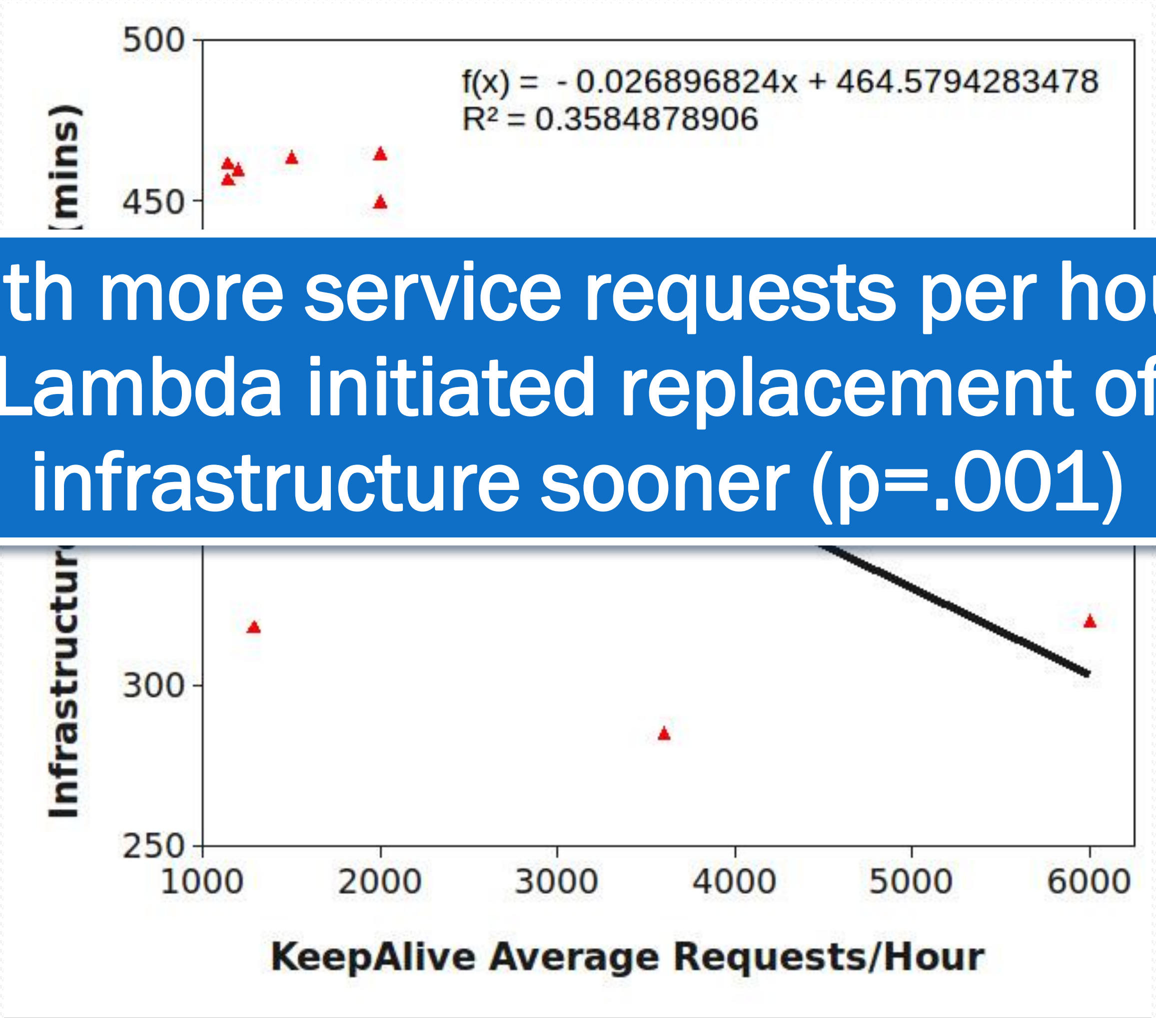
AWS Lambda: time to infrastructure replacement vs. memory reservation size



Memory sizes tested: 192, 256, 384, 512 MB

RQ-4: Persisting Infrastructure

AWS Lambda: time to infrastructure replacement vs. memory reservation size



With more service requests per hour, Lambda initiated replacement of infrastructure sooner (p=.001)

Memory sizes tested: 192, 256, 384, 512 MB

RQ-4: Persisting Infrastructure

Keep-Alive Infrastructure Preservation

- PRMS Service: parameterize for "ping"
  - Perform sleep (idle CPU) – do not run model
  - Provides delay to overlap (n=100) parallel requests to preserve infrastructure
- Ping intervals: tested 3, 4, and 5-minutes
- VM Keep-Alive client:
  c4.8xlarge 36 vCPU instance: ~4.5s sleep
- CloudWatch Keep-Alive client:
  100 rules x 5 targets: 5-s sleep

RQ-4: Keep-Alive Client Summary

Client type	c4.8xlarge VM	c4.8xlarge VM	CloudWatch	CloudWatch
Ping interval	5 min	3 min	5 min	4min
Keep-Alive calls/batch	100	100	500	500
Slowdown vs. WARM	13.3%	0.7%	11.6%	35.0%
Speedup vs. COLD	4.03x	4.53x	4.1x	3.4x
Test runs	32	32	26	17
Test duration (hrs)	24	24	18	12
Average new Lambda firecracker containers/test	2.41	0.38	5.42	14.71
Keep-Alive runtime avg (ms)	4492	4463	5200	5200
Memory (GB-sec/hour)	2695	4463	15600	19500
Keep-Alive cost/year	$4,484.00	$4,494.76	$2,278.06	$2,847.57

Keep-Alive clients can support trading off cost for performance for preserving FaaS infrastructure to mitigate cold start latency

RQ-4: Keep-Alive Client Summary

Client type	c4.8xlarge VM	c4.8xlarge VM	CloudWatch	CloudWatch
Ping interval	5 min	3 min	5 min	4min
Keep-Alive calls/batch	100	100	500	500
Slowdown vs. WARM	13.3%	0.7%	11.6%	35.0%
Speedup vs. COLD	4.03x	4.53x	4.1x	3.4x
Test runs	32	32	26	17
Test duration (hrs)	24	24	18	12
Average new Lambda firecracker containers/test	2.41	0.38	5.42	14.71
Keep-Alive runtime avg (ms)	4492	4463	5200	5200
Memory (GB-sec/hour)	2695	4463	15600	19500
Keep-Alive cost/year	$4,484.00	$4,494.76	$2,278.06	$2,847.57

Keep-Alive clients can support trading off cost for performance for preserving FaaS infrastructure to mitigate cold start latency

RQ-4: Keep-Alive Client Summary

Client type	c4.8xlarge VM	c4.8xlarge VM	CloudWatch	CloudWatch
Ping interval	5 min	3 min	5 min	4min
Keep-Alive calls/batch	100	100	500	500
Slowdown vs. WARM	13.3%	0.7%	11.6%	35.0%
Speedup vs. COLD	4.03x	4.53x	4.1x	3.4x
Test runs	32	32	26	17
Test duration (hrs)	24	24	18	12
Average new Lambda firecracker containers/test	2.41	0.38	5.42	14.71
Keep-Alive runtime avg (ms)	4492	4463	5200	5200
Memory (GB-sec/hour)	2695	4463	15600	19500
Keep-Alive cost/year	$4,484.00	$4,494.76	$2,278.06	$2,847.57

Keep-Alive clients can support trading off cost for performance for preserving FaaS infrastructure to mitigate cold start latency

Outline

- Background
- Research Questions
- Experimental Workloads
- Experiments/Evaluation
- Conclusions

Conclusions

- **<u>RQ-1 Memory Reservation Size</u>**:
  - MAX memory: 10x speedup, 7x more hosts
- **<u>RQ-2 Scaling Performance</u>**:
  - 1+ scale-up near warm, COLD scale-up is slow
- **<u>RQ-3 Cost</u>**
  - m4.large $35 (14d), Lambda $66 (2.3 hr), $125 (42 min)
- **<u>RQ-4 Persisting Infrastructure (Keep-Alive)</u>**
  - c4.8xlarge VM $4,484/yr (13.3% slowdown vs warm, 4x ↑), CloudWatch $2,278/yr (11.6% slowdown vs warm, 4.1x ↑)

Questions