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

Minh Vu*, Baojia Zhang*, Olaf David, George Leavesley, Wes Lloyd†

December 20, 2018

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


Outline

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

- Pay only for CPU/memory utilization
- High Availability
- Fault Tolerance
- Infrastructure Elasticity
- No Setup
- Function-as-a-Service (FAAS)

Why Serverless Computing?
Many features of distributed systems, that are challenging to deliver, are provided automatically
...they are built into the platform
Serverless Platforms

- AWS Lambda
- Azure Functions
- IBM Cloud Functions
- Google Cloud Functions
- Apache OpenWhisk
- Fn (Oracle)

Serverless Computing

Research Challenges
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

- Unused infrastructure is deprecated
  - *But after how long?*
- AWS Lambda: Bare-metal hosts, firecracker micro-VMs
- Infrastructure states: [https://firecracker-microvm.github.io/](https://firecracker-microvm.github.io/)
- **Provider-COLD / Host-COLD**
  - Function package built/transferred to Hosts
- **Container-COLD** (firecracker micro-VM)
  - Image cached on Host
- **Container-WARM** (firecracker micro-VM)
  - “Container” running on Host
Improving Application Migration to Serverless Computing Platforms: Latency Mitigation with Keep-Alive Workloads

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

**RQ4:** **PERSISTING INFRASTRUCTURE:** 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 memory reservation size and scaling for model service deployment to AWS Lambda

---

**PRMS Lambda Testing**

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

REST/JSON

API GATEWAY

PRMS service

**Fixed-availability zone:** EC2 client / Lambda server us-east-1e

Max service duration: < 30 seconds

Memory: 256 to 3008MB
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

**PRMS AWS Lambda Performance (100 concurrent requests)**

c4.2xlarge – average of 8 runs

- **c4.2xlarge client**
- **c4.8xlarge client**

![Graph showing execution time vs memory reservation size for c4.2xlarge and c4.8xlarge clients.](image)
### RQ-1: AWS Lambda Memory Reservation Size

#### Memory Speedup (256 → 3008 MB):

<table>
<thead>
<tr>
<th>Speedup @</th>
<th>Memory Speedup (256 → 3008 MB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>256MB</td>
<td>4.3 X 8-vCPU client</td>
</tr>
<tr>
<td>1024MB</td>
<td>10.1 X 36-vCPU client</td>
</tr>
</tbody>
</table>

#### PRMS AWS Lambda Performance (100 concurrent requests)

<table>
<thead>
<tr>
<th>Memory Reservation Size (MB)</th>
<th>c4.2xlarge client</th>
<th>c4.8xlarge client</th>
</tr>
</thead>
<tbody>
<tr>
<td>Speedup @ 256MB</td>
<td>4.3x</td>
<td>10.1x</td>
</tr>
<tr>
<td>Speedup @ 1024MB</td>
<td>1.3x</td>
<td>1.9x</td>
</tr>
<tr>
<td>Speedup @ 1536MB</td>
<td>1.14x</td>
<td>1.4x</td>
</tr>
<tr>
<td>Speedup @ 2048MB</td>
<td>1.06x</td>
<td>1.2x</td>
</tr>
</tbody>
</table>

### RQ-1: AWS Lambda Memory Reservation Size - Infrastructure

#### AWS Lambda Hosting Infrastructure - PRMS Service

- c4.2xlarge – average of 8 runs
- Firecracker containers
- Hosts – c4.2xlarge client (8 vCPUs)
- Hosts – c4.8xlarge client (36 vCPUs)
Many more Hosts leveraged when memory > 1536 MB

AWS Lambda Hosting Infrastructure - PRMS Service

RQ-1: AWS Lambda
Memory Reservation Size - Infrastructure
c4.2xlarge – average of 8 runs

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

AWS Lambda Hosting Infrastructure - PRMS Service

c4.2xlarge – average of 8 runs
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

# of concurrent runs
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

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?

- 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

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

<table>
<thead>
<tr>
<th>Client type</th>
<th>c4.8xlarge VM</th>
<th>c4.8xlarge VM</th>
<th>CloudWatch</th>
<th>CloudWatch</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ping interval</td>
<td>5 min</td>
<td>3 min</td>
<td>5 min</td>
<td>4min</td>
</tr>
<tr>
<td>Keep-Alive calls/batch</td>
<td>109</td>
<td>100</td>
<td>500</td>
<td>500</td>
</tr>
<tr>
<td>Slowdown vs. WARM</td>
<td>13.3%</td>
<td>0.7%</td>
<td>11.6%</td>
<td>35.0%</td>
</tr>
<tr>
<td>Speedup vs. COLD</td>
<td>4.03x</td>
<td>4.53x</td>
<td>4.1x</td>
<td>3.4x</td>
</tr>
<tr>
<td>Test runs</td>
<td>32</td>
<td>32</td>
<td>26</td>
<td>17</td>
</tr>
<tr>
<td>Test duration (hrs)</td>
<td>24</td>
<td>24</td>
<td>18</td>
<td>12</td>
</tr>
<tr>
<td>Average new Lambda firecracker containers/test</td>
<td>2.41</td>
<td>0.38</td>
<td>5.42</td>
<td>14.71</td>
</tr>
<tr>
<td>Keep-Alive runtime avg (ms)</td>
<td>4492</td>
<td>4483</td>
<td>5200</td>
<td>5200</td>
</tr>
<tr>
<td>Memory (GB/sec/hour)</td>
<td>2695</td>
<td>4463</td>
<td>15600</td>
<td>19500</td>
</tr>
<tr>
<td>Keep-Alive cost/yr</td>
<td>$4,484.00</td>
<td>$4,494.76</td>
<td>$2,278.06</td>
<td>$2,847.57</td>
</tr>
</tbody>
</table>

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

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

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

December 20, 2018

RQ-4: Keep-Alive Client Summary

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

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

December 20, 2018
Outline

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

Conclusions

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