FaaSNet: Scalable and Fast Provisioning of Custom Serverless Container Runtimes at Alibaba Cloud Function Compute

Ao Wang, Shuai Chang, Huangshi Tian,
Hongqi Wang, Haoran Yang, Huiba Li, Rui Du, Yue Cheng
Function-as-a-Service

- FaaS enables cloud tenants to launch short-lived tasks (i.e., Lambda functions) with high elasticity and fine-grained resource billing (1ms)
- Function: basic unit of deployment. Application consists of multiple serverless functions
- Popular use cases: Backend APIs, event/async processing…
**FaaS providers** normally limit tenants **code package** in tens of **MB level**

- FaaS provider enables cloud tenants to launch short-lived tasks (i.e., Lambda functions) with *high elasticity* and *fine-grained* resource billing (1ms)
- Function: basic unit of deployment. Application consists of multiple serverless functions
- Popular use cases: Backend APIs, event/async processing…

![Image of tenants, functions, and FaaS provider diagram]
FaaS and custom-container runtimes
FaaS and custom-container runtimes

FaaS + container unlocks new workload possibilities and makes serverless accessible to a broader audience.
FaaS and custom-container runtimes

<table>
<thead>
<tr>
<th></th>
<th>Container</th>
<th>FaaS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Application transplant</td>
<td>Builds once, runs anywhere</td>
<td>Cloud vendor lock-in</td>
</tr>
<tr>
<td>Dev tools</td>
<td>open source ecosystems</td>
<td>Cloud vendor lock-in</td>
</tr>
<tr>
<td>CI/CD</td>
<td>open source ecosystems</td>
<td>Cloud vendor lock-in</td>
</tr>
<tr>
<td>Scalability</td>
<td>Second level</td>
<td>Millisecond level</td>
</tr>
<tr>
<td>Runtimes</td>
<td>Custom</td>
<td>Provided runtimes</td>
</tr>
</tbody>
</table>

FaaS + container unlocks new workload possibilities and makes serverless accessible to a broader audience.
Workload analysis

• Alibaba Cloud Function Compute 15-day-production log during May 2021

• Data centers: Beijing, Shanghai
Workload analysis

FaaS workloads are *bursty* and *dynamic*
Workload analysis

FaaS workloads are *bursty* and *dynamic*
Workload analysis

• Image pull latency distribution

• Proportion of image pull in function cold start
Workload analysis

• Image pull latency distribution

• Proportion of image pull in function cold start

Pull image in tens second level

➢ 57% image pulls larger than 45 seconds
Workload analysis

- Image pull latency distribution
- Proportion of image pull in function cold start

Image pull dominates function startup time
- A large fraction of startup time is spent on pulling images
Workload analysis

- Image pull latency distribution

- **To handle workload dynamicity** - Scalable and resilient provisioning of large numbers of function containers

- **To reduce cold start latency** - Optimize the performance of container provisioning process
State-of-the-art solutions

Bottleneck under *burst* requests

Host VMs – limited resources:
2 CPUs, 4GB Mem, 1Gbps network
State-of-the-art solutions

Problems:

• **Extra, dedicated, centralized** components
• **Limited** VM resources
• VM’s lifecycle is **unpredictable**
• **Multi-Tenancy isolation under FaaS** is not considered
FaaSNet: Scalable and Fast Provisioning of Custom Serverless Container Runtimes at Alibaba Cloud Function Compute
Agenda

• FaaSNet design
• Evaluation
• Conclusion
FaaSNet design

Diagram of FaaSNet design:
- **Gateway**
- **Scheduler**
- **VM manager**
- **Container registry**
- **Metadata store**
- **VM1**, **VM2**, **VM3**, **VM4**, **VM5**
- **FaasMesh FT manager**
- **FaasMesh worker**
- **Function containers**
- Directions:
  - Client req → Gateway → Scheduler → VM manager
  - Container registry → Gateway
  - Metadata store → Gateway
  - VM manager → Scheduler
  - Invoke → VM1
  - VM ctrl → VM1
  - VM1 → VM2, VM3
  - VM2, VM3 → VM4, VM5
  - VM4, VM5 → ...
Function tree (FT)

- FT is perfect self-balanced binary tree in **Function** level
- Exposed 2 APIs
  - insert
  - delete
Container provisioning protocol

1. Query FT

Scheduler

FaaSNet
FT manager

VM2 - Upstream

VM1 - Downstream

MDS

Request

: Data path

: Control path

: Image data
Container provisioning protocol

2. Send Metadata

VM2 - Upstream

VM1 - Downstream

Scheduler

FaaSNet
FT
manager

: Image data
: Data path
: Control path

MDS
Container provisioning protocol

Scheduler

FaaSNet
FT
gmanager

: Image data

: Data path

: Control path

VM2 - Upstream

3. Download image manifest & load image

VM1 - Downstream

MDS
Container provisioning protocol

4. Ready to create Container

MDS

VM2 - Upstream

VM1 - Downstream

Scheduler

FaaSNet
FT
manager

: Image data

: Data path

: Control path
Container provisioning protocol

5. Data fetching

VM2 - Upstream

VM1 - Downstream

Scheduler

FaaSNet
FT manager

: Image data

: Data path

: Control path

MDS
Agenda

• FaaSNet design

• Evaluation

• Conclusion
Experimental setup

• Testbed is up to 1,000 VMs

• VM type: 2 CPUs, 4 GB memory, 1 Gbps network
  • Same as our production environment

• Example container image
  • 758 MB python-based function with ~2 sec duration
FaaSNet’s performance

**Kraken**: Kraken dev cluster

**Baseline**: Original Alibaba Cloud Function Compute
FaaSNet’s performance

**Kraken**: Kraken dev cluster

**Baseline**: Original Alibaba Cloud Function Compute (FC)

**On-demand**: FC + I/O efficient format

**DADI+P2P**: FC + DADI
FaaSNet’s performance

**Kraken**: Kraken dev cluster

**Baseline**: Original Alibaba Cloud Function Compute (FC)

**On-demand**: FC + I/O efficient format

**DADI+P2P**: FC + DADI

**FaaSNet**

FaaSNet has **strong scalability**
Production workload

Scales thousands of containers in seconds
Production workload

Latency (sec)

Timeline (minutes)

TPS

CDF

8 9 10 11 12

FaaSNet

On-demand

RPS

5.6x

7x

IoT trace
Conclusion

• FaaSNet is the first system that provides an end-to-end, integrated solution for FaaS-optimized container runtime provisioning (Alibaba Cloud Function Compute)

• FaaSNet scales 13.4x faster than Alibaba Cloud's current FaaS platform
Thank you!

• Contact: Ao Wang – awang24@gmu.edu

• FT prototype & Alibaba Cloud Function Compute cold start traces
  • https://github.com/mason-leap-lab/FaaSNet
Q&A

- Local disk full -> cache eviction -> performance degradation
- Bandwidth issues