Fine-Grained Isolation for Scalable, Dynamic, Multi-tenant Edge Clouds

Yuxin Ren, Guyue Liu, Vlad Nitu, Wenyuan Shao, Riley Kennedy, Gabriel Parmer, Timothy Wood, Alain Tchana

Presented by: Vlad Nitu
Edge Cloud

• 40 million IoT devices in 2027
• CPSs require real-time reaction
• “code moving to data” -> edge computing
• Edge clouds: tiny datacenters deployed close to the user
Edge cloud

• Application requirements:
  • “Bump-in-the-wire” -> 5G Cellular processing, network middleboxes...
  • Predictable low latency -> Edge caches, IoT/CPS control...

• Edge Cloud requirements:
  • Serve a large number of clients with high churn
  • Efficiently use limited resources
  • Guarantee strong isolation: between untrusted services and clients
Edge Cloud: Isolation
Edge Cloud: Isolation

Service isolation

TLS Termination

Firewall

Inference

Memcached

IDS

LB

DPI

Firewall
Edge Cloud: Isolation

Client isolation
Existing Solutions

• Process

• Container

• Virtual machine
## Existing Solutions

<table>
<thead>
<tr>
<th></th>
<th>Isolation</th>
<th>Scalability</th>
<th>Startup time</th>
<th>High performance networking</th>
</tr>
</thead>
<tbody>
<tr>
<td>process</td>
<td>❌</td>
<td>✔</td>
<td>○</td>
<td>○</td>
</tr>
<tr>
<td>container</td>
<td>○</td>
<td>○</td>
<td>❌</td>
<td>○</td>
</tr>
<tr>
<td>Virtual machine</td>
<td>✔</td>
<td>❌</td>
<td>❌</td>
<td>❌</td>
</tr>
</tbody>
</table>
### Existing Solutions

<table>
<thead>
<tr>
<th></th>
<th>Isolation</th>
<th>Scalability</th>
<th>Startup time</th>
<th>High performance networking</th>
</tr>
</thead>
<tbody>
<tr>
<td>process</td>
<td>✗</td>
<td>✓</td>
<td>○</td>
<td>○</td>
</tr>
<tr>
<td>container</td>
<td>○</td>
<td>○</td>
<td>✗</td>
<td>○</td>
</tr>
<tr>
<td>Virtual machine</td>
<td>✓</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
</tr>
<tr>
<td>EdgeOS</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>
EdgeOS: Isolation, Predictability, and Scale

- Based on Composite microkernel OS, designed for Real-Time guarantees
- High speed data movement (10Gbps+) without sacrificing isolation
- Startup **170X** faster than fork+exec and **84,000X** faster than containers!
- Scales to 1000s of services per host = 1 service per user!
How is this possible?

1. Feather Weight Processes
   - Lightweight process abstraction
   - Minimal memory footprint
   - Recycle FWP for fast startup

2. Memory Movement Accelerator
   - Mediates FWP communication
   - Securely copies data
   - Efficiently manages buffers

3. uKernel and Control Plane
   - Manages lifecycle and scheduling
   - Defines FWP data flow
   - Capability-based access control
EdgeOS architecture

• FWP (Feather-Weight Processes):
  • Minimal abstractions: memory + a small set of kernel resources
  • Input and output message rings
  • Library-based OS services
  • Small enough to instantiate one per incoming client or group of clients
  • Recycled to clean state for fast startup
EdgeOS architecture

- MMA (Memory Movement Accelerator):
  - Enables chains of FWP services
  - Enforce isolation through data copying
  - Executed on dedicated cores
  - Sustain throughput competitive with data sharing
  - Optimized buffer allocation and integration with the FWP scheduler
EdgeOS architecture

- Data plane:
  - FWPs and MMA
  - DPDK-based networking

- Control plane:
  - The EdgeOS controller
  - The FWP Manager
  - The Scheduler
EdgeOS: packet processing steps
Evaluation: start time

- Docker: the execution time of “docker start”
- Firecracker: the start time of the recommended “hello” image
- Linux: fork() + exec()
Evaluation: start time

- EdgeOS creates an FWP 20x faster than a Linux process
Evaluation: start time

- EdgeOS creates an FWP 20x faster than a Linux process
- When the FWP is cached, the activation time is 170x faster than Linux
Evaluation: start time

- EdgeOS creates an FWP 20x faster than a Linux process
- When the FWP is cached, the activation time is 170x faster than Linux
- FWP activation is $\sim 10^5$ faster than “docker start”
Evaluation: memcopy overhead

- EdgeOS provides isolation and adds negligible overheads
Evaluation: scalability

FPWs as middleboxes

Latency (100us) vs #Clients

- ONVM-chain
- ONVM-single
- EOS-chain
- EOS-single
Evaluation: scalability

FPWs as middleboxes

- ONVM-chain
- ONVM-single
- EOS-chain
- EOS-single

Latency (100us) vs. #Clients

25x improvement
Evaluation: scalability

FPWs as middleboxes

Latency (100us) vs. #Clients

- ONVM-chain
- ONVM-single
- EOS-chain
- EOS-single

2x performance advantage
Evaluation: scalability

FWPs as TLS proxys

EOS throughput

Linux throughput

Throughput (10K reqs/sec)

#Instances
Conclusion

• EdgeOS: an OS for Edge clouds
  • Strong copy-based isolation
  • Minimalistic execution instances
  • Optimized for high churn and dense multi-tenancy

• Start-up times up to 170x faster than Linux processes and $10^5$x faster than Docker containers

• Maintain line rate even with chains of 6 FWPs

• Substantially improved scalability
Thank you for your attention!

Vlad Nitu: vlad.nitu@insa-lyon.fr
Yuxin Ren: ryx@gwmail.gwu.edu
Gabriel Parmer: gparmer@gwu.edu
Timothy Wood: timwood@gwu.edu