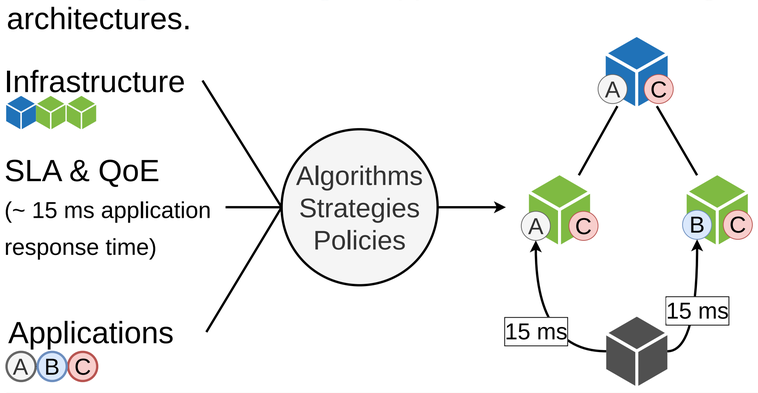
API-driven Cloud-Edge Orchestration with PULCEO*
- A Proof of Concept -
***Platform for Universal and Lightweight Cloud-Edge Orchestration**

Sebastian Böhm · Guido Wirtz
- Distributed Systems Group -
Faculty of Information Systems and Applied Computer Sciences, University of Bamberg

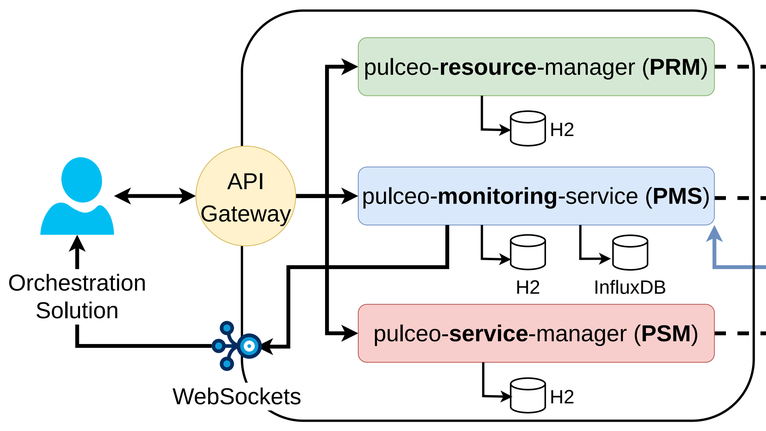
Problem Domain

Edge Computing: Shifting of compute and storage resources in close proximity to end-users
Advantage: *Latency, Bandwidth, Security*
Disadvantage: *Complex Decision Making, Deployment*

Many solution techniques exist for ***service placement***, ***offloading***, and ***scaling*** of applications in cloud-edge architectures.

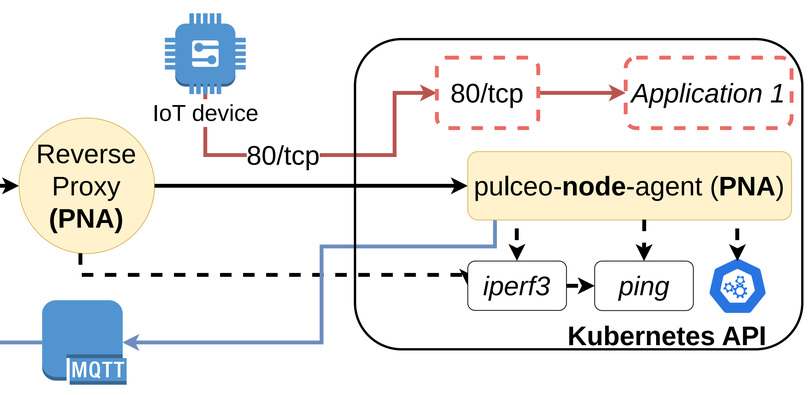


PULCEO System Overview



- RESTful HTTP API exposed via an API Gateway
- Decoupled cloud-edge orchestration
- Real-time data streaming via WebSockets
- Microservice architecture based on [1]

PULCEO Node Agent



- RESTful HTTP API
- Monitoring data transmitted with MQTT
- Latency and bandwidth measurement
- Kubernetes as container manager

Solution: API-driven Holistic Cloud-Edge Orchestration

① Creation supported by PRM

POST /providers
```
{ name: azure-provider,
type: AZURE, credentials: ... }
```

Credentials

Service Principal · Entra ID · Azure
User grants access for **PULCEO**

POST /nodes
```
{ name: cloud1, provider: azure-provider
cpu: 4, memory: 16, region: eastus, ... }
```

POST /nodes
```
{ name: fog1, provider: azure-provider
cpu: 2, memory: 8, region: france, ... }
```

POST /links
```
{ name: fog3-fog1, srcNodeId: fog3,
destNodeId: fog1, ... }
```

POST /nodes
```
{ name: fog3, provider: default ...,
pnaInitToken: ..., hostname: ... }
```

② Monitoring supported by PMS

POST /metric-requests
```
{ type: icmp-rtt, linkId: fog3-fog1,
recurrence: 3600, ... }
```

POST /metric-requests
```
{ type: cpu-util, nodeId: *,
recurrence: 60, ... }
```

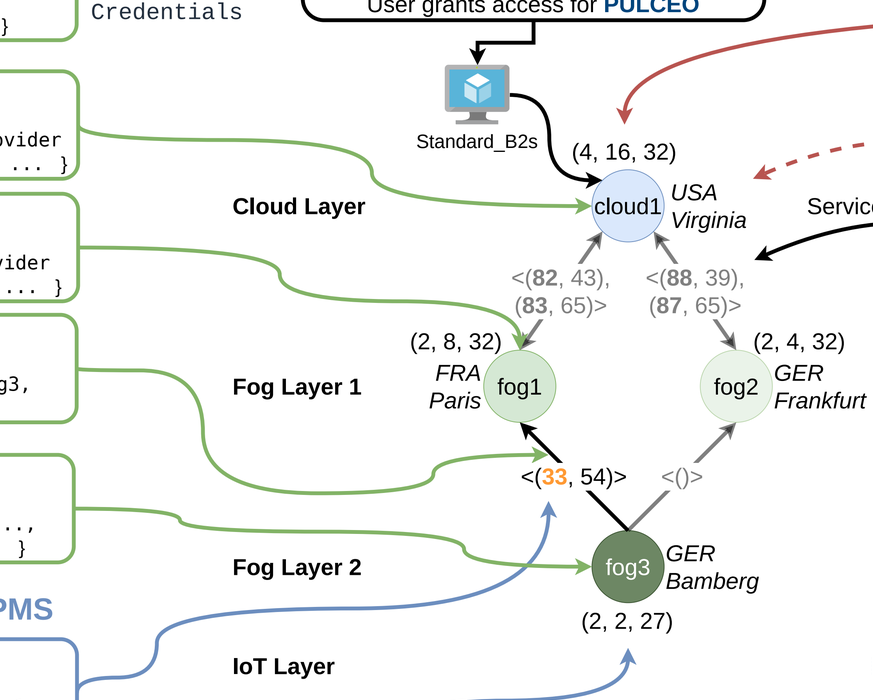


Fig. 3. Combined Sample Cloud-Edge Topology. Directed edges imply the direction of latency and bandwidth measurement, shown by a list of pairs with $<(latency, bandwidth), \ldots>$. Weights W show the available latency and bandwidth. For example, $w(cloud1, fog1)$ has a RTT of 83 ms and an available bandwidth (upload) of 65 Mbps. Vice versa, $w(fog1, cloud1)$ has a RTT of 82 ms and an available bandwidth (upload) of 43 Mbps. Consequently, $upload(w(cloud1, fog)) = download(w(fog1, cloud1))$. The capabilities of *Nodes* are described as 3-tuples with $(CPU, memory, storage)$. The right side of the figure shows a microservice *Application*, modeled as a directed acyclic graph, that must be placed.

③ Operation supported by PSM

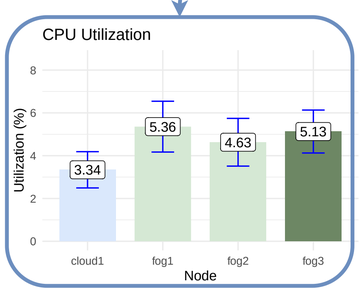
GET /nodes/cloud1/cpu
```
{ cpuCapacity: {...},
cpuAllocatable: { shares: 4000 } }
```

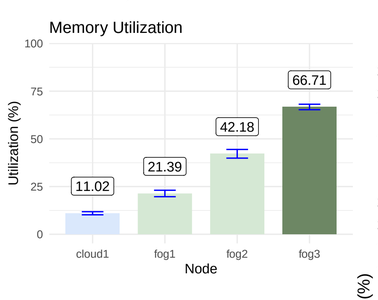
POST /applications
```
{ { nodeId: cloud1, name: edge-iot-
simulator, applicationComponents: [ { name:
component-eis, image: ghcr.io/spboehm/edge-iot-
simulator:v1.1.0, port: 80, protocol:
HTTPS, applicationComponentType:
PUBLIC, environmentVariables: { ... } }]}}
```

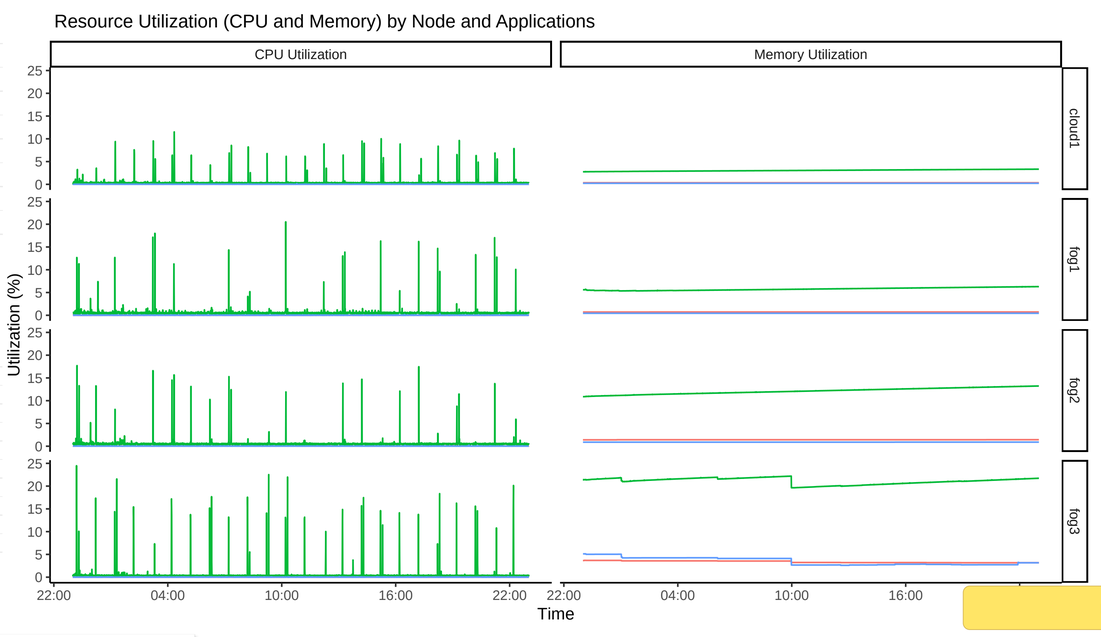
PATCH /nodes/cloud1/cpu/allocatable
```
{ key: shares,
value: 3000 }
```

GET /nodes/cloud1/cpu
```
{ "uuid": "...", "nodeUUID": "...",
"nodeName": "cloud1", cpuCapacity: {...},
"cpuAllocatable": { "modelName": "Intel(R)
Xeon(R) CPU E5-2673 v4 @ 2.30GHz",
"cores": 4, "threads": 4, "bogoMIPS": 4589.37,
"minimalFrequency": 2294.685,
"averageFrequency": 2294.685,
"maximalFrequency": 2294.685, "shares": 3000,
"slots": 0.0, "mips": 4589.37, "gflop": 0.0 } }
```

④ Evaluation with cross-cutting support from all microservices







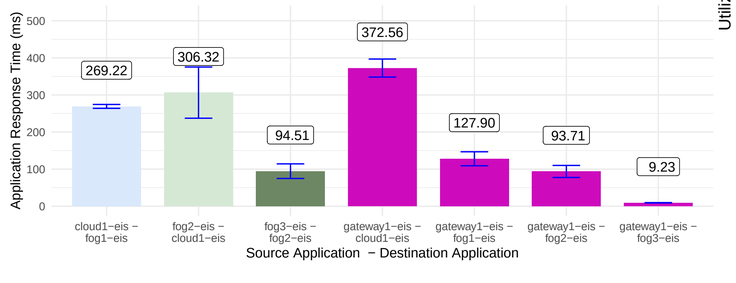


Table 3. ICMP round-trip time (ms) between nodes.

v_1	v_2	Min	Mean	Max	Med
cloud1	fog1	80.795	81.142	82.924	81.098
cloud1	fog2	86.709	88.896	91.149	89.038
fog1	cloud1	80.779	81.107	82.327	81.024
fog2	cloud1	86.460	87.802	88.819	87.960
fog3	fog1	25.558	26.139	33.058	25.802
fog3	fog2	13.077	15.402	24.150	14.709

Table 4. TCP and UDP bandwidth (Mbps) between nodes.

		TCP					UDP				
v_1	v_2	Min	Mean	Max	Med	SD	Min	Mean	Max	Med	SD
cloud1	fog1	65.000	65.000	65.000	65.000	0.000	63.800	63.942	64.000	63.900	0.058
cloud1	fog2	65.000	65.000	65.000	65.000	0.000	63.800	63.875	63.900	63.900	0.044
fog1	cloud1	43.400	64.061	65.000	65.000	4.504	56.200	63.354	64.000	64.000	2.084
fog2	cloud1	39.400	63.887	65.000	65.000	5.338	55.300	63.517	63.900	63.900	1.751
fog3	fog1	53.500	64.392	65.000	65.000	2.385	64.400	64.443	64.600	64.400	0.066
fog3	fog2	64.400	64.950	65.000	65.000	0.169	64.700	64.762	64.800	64.800	0.049

⑤ Documentation

Asynchronous raw data export

POST /metric-exports
```
{ "metricType": "CPU_UTIL" }
```
Response:
```
{ "metricExportUUID": "...",
"metricType": "CPU_UTIL", "numberOfRecords": 100,
"url": "https://pulceo.io/.../684-cpu-util.csv",
"metricExportState": "PENDING" }
```

Orchestration reports

IDLE LOAD

Topology, System and Application Monitoring, Resources, Applications, Response times, Events, Charts

Contributions & Limitations

- RESTful HTTP API for universal orchestration
- Decoupled cloud-edge orchestration
- Holistic cloud-edge orchestration with *creation, monitoring, operation, evaluation, and documentation*
- *Only one representational architecture implemented with stable network conditions*

Distributed Systems Group
Code@GitHub

https://www.uni-bamberg.de/pi
https://github.com/uniba-dsg

[1] B. Costa, J. Bachiega, L. R. de Carvalho, and A. P. F. Araujo, "Orchestration in Fog Computing: A Comprehensive Survey," ACM Comput. Surv., vol. 55, no. 2, pp. 1–34, Feb. 2023, doi: 10.1145/3486221.