Towards Multi-Stakeholder Clouds

Bohdan Borysei¹, Stefan Saroiu², Eyal de Lara¹

¹University of Toronto, ²Microsoft
Infrastructure Management Tasks

- Power on/off/reboot
- Firmware upgrade
- Provisioning
- Sensors monitoring
- Hardware monitoring
- Events logging
- Networking configuration
- TPM configuration
- Host configuration
- Troubleshooting
Infrastructure Management Challenge due to Multi-Stakeholders

Reboot

Cloud Provider

Hosting Site Operator

Power Spike
Infrastructure Management Challenge due to Multi-Stakeholders

Performance Degradation

Cloud Provider

Firmware Upgrade

Hosting Site Operator
## MCloud Today

<table>
<thead>
<tr>
<th></th>
<th>Cloud Provider</th>
<th>Hosting Site Operator</th>
</tr>
</thead>
<tbody>
<tr>
<td>AWS Outpost/Wavelength</td>
<td>✔️</td>
<td>✗</td>
</tr>
<tr>
<td>Azure Stack Edge</td>
<td>✔️</td>
<td>✗</td>
</tr>
<tr>
<td>Azure Operator Nexus</td>
<td>✗</td>
<td>✔️</td>
</tr>
<tr>
<td>Google Anthos</td>
<td>✗</td>
<td>✔️</td>
</tr>
</tbody>
</table>
MCloud needs a new management primitive that enables multiple stakeholders to collaboratively manage their infrastructure.
Two-Person Control ($P^2C$)
Two-Person Control (P²C)

- No one party can independently perform infrastructure task
- Disperses control, responsibility, accountability btw parties
- Used when actions have significant consequences in case of mistakes:
  - Financial, military, pharmaceutical sectors
P²C Authorization Framework Categories

1. Either-party
   ○ *e.g.*, creation/termination of a server management session

2. Both-parties
   ○ *e.g.*, system factory reset

3. Both-parties-with-policy
   ○ *e.g.*, filtering CPU load monitoring output for privacy reasons
   ▪ *or* system reboot only when rack power consumption < X
Modern Server

Remote Admin Access via RedFish

*BMC* Baseboard Management Controller
P²C Prototype for BMC

- P²C prototype is built on top of OpenBMC
  - Intercepts API calls and applies P²C authorization framework

- Two deployment prototypes:
  - QEMU running OpenBMC managing single server
  - Standalone daemon making RedFish calls to 8-server rack
Examples of Operations

<table>
<thead>
<tr>
<th>Type</th>
<th>Operation</th>
<th>Description</th>
<th>P²C</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operation</td>
<td>Power on, off, reboot $system</td>
<td>Reset $system</td>
<td>(HSO and cp) or (CP and hso)</td>
</tr>
</tbody>
</table>

**Policy**

`hso.pre: SUM(External.PDU.Outlets.Power) < 1000`
### Examples of Operations

<table>
<thead>
<tr>
<th>Type</th>
<th>Operation</th>
<th>Description</th>
<th>P²C</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operation</td>
<td>Power on, off, reboot $system</td>
<td>Reset $system</td>
<td>(HSO and cp) or</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>(CP and hso)</td>
</tr>
</tbody>
</table>

**Policy**

```solidus
cp.pre: COUNT(External.Servers.State=="Up") >= 6
```
## Examples of Operations

<table>
<thead>
<tr>
<th>Type</th>
<th>Operation</th>
<th>Description</th>
<th>P2C</th>
</tr>
</thead>
<tbody>
<tr>
<td>Logging</td>
<td>Get EventLog entries</td>
<td>Return System Event Log’s collection</td>
<td>(HSO and cp) or (CP and hso)</td>
</tr>
</tbody>
</table>

#### Policy

- **hso.post**: LogEntry.Oem* := null
- **cp.post**: LogEntry.Severity == "Critical"
## Examples of Operations

**Type** | **Operation** | **Description** | **P²C**
---|---|---|---
Monitoring | Get a $chassis Chassis | Returns schema of server’s $chassis | HSO or (CP and hso)

**Policy**

```hso.post: Chassis.Certificates := null```
Server 2 Reboot

Callbacks (Read Server 1-8 Power)

Approve 1-8

Rebooting Server 2

Callback (Server 2 reboot)

Read Server 1-8 Power

Approve (Server 2 reboot)

Return Server 1-8 Power

Check SUM(Server 1-8 Power) less than 1 Kilowatt

Cloud Provider Admin

Our Proxy

Hosting Site Operator
Open Research Directions

- Policy-as-code
  - What is the right trusted execution model for policy-as-code?
- Proxy trust model, provisioning, recovery
- Many-party control
  - Policy conflict resolving
- See paper for more
  - Networking considerations, crowdsourcing, nested P2C authorization framework, etc.
Conclusions

- Identified problem of MCloud management
- Argued that collaboration is required to succeed
- Proposed P²C framework enabling both parties to participate in MCloud management
- Described possible future research directions
Q&A
MCloud Today

AWS Outpost/Wavelength:
- **Amazon** builds/install/operates server/rack
- **Client** provides site facility

Azure Stack Edge:
- **Microsoft** builds/operates server
- **Client** installs server and provides site facility

Azure Operator Nexus:
- **Microsoft** provides BoM specs
- **Client** builds/install/operates whole rack

Google Anthos:
- **Google** provides reference specification
- System Integrator builds whole rack
- **Client** installs/operates whole rack