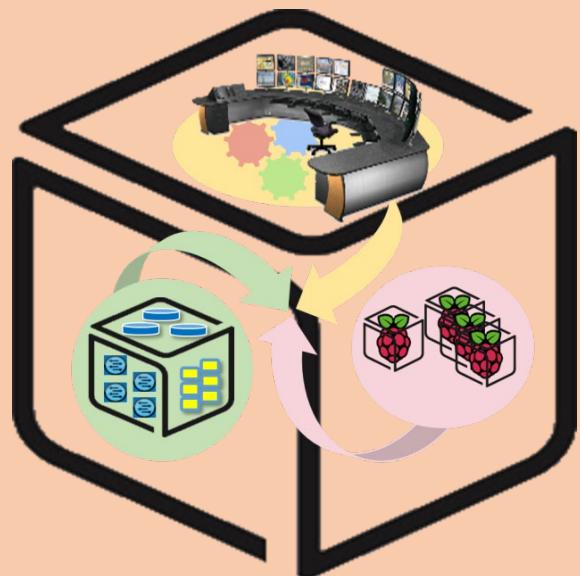


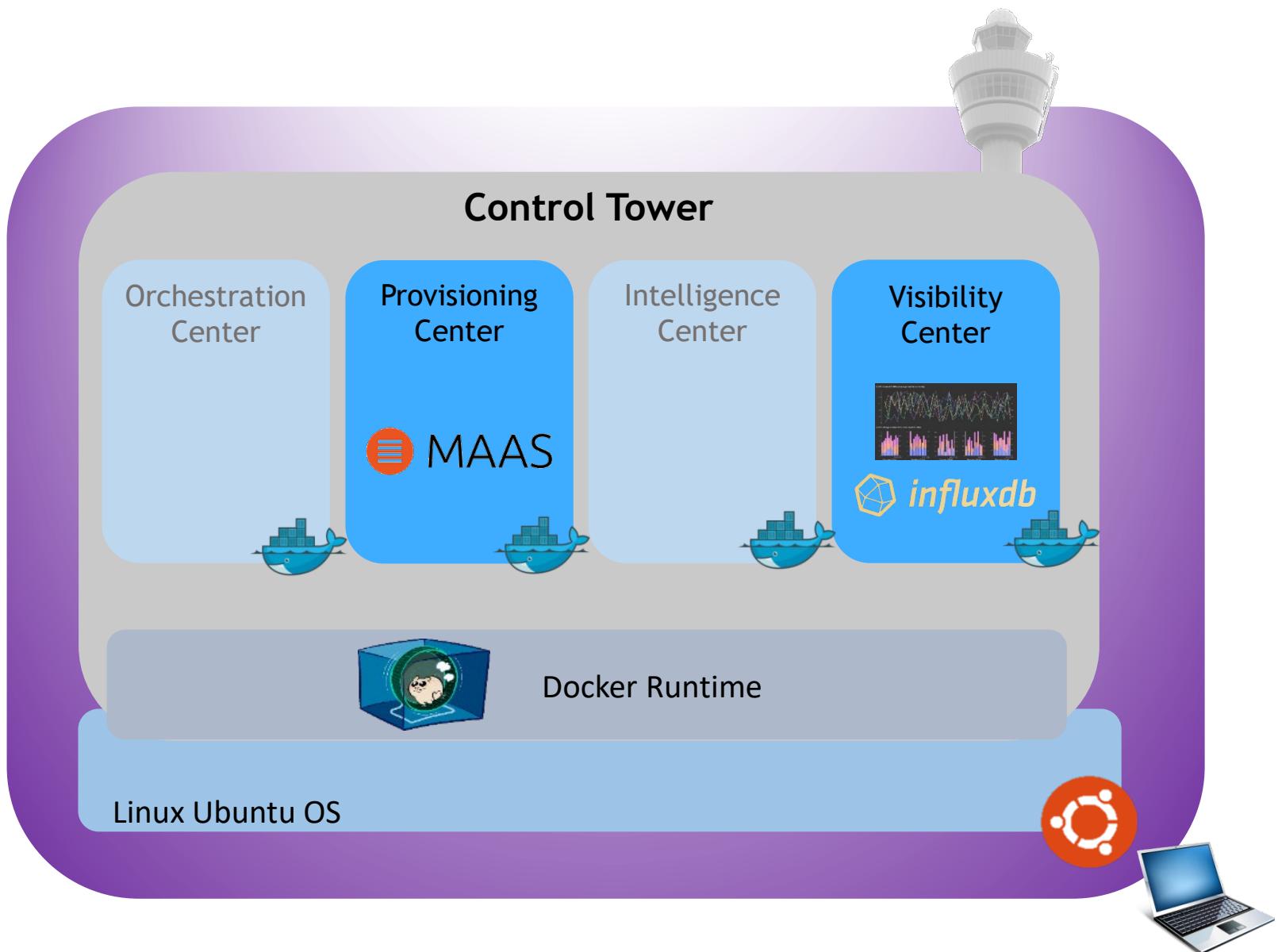
Computer Systems For AI-inspired Cloud Theory & Lab.

Lab #3: Tower

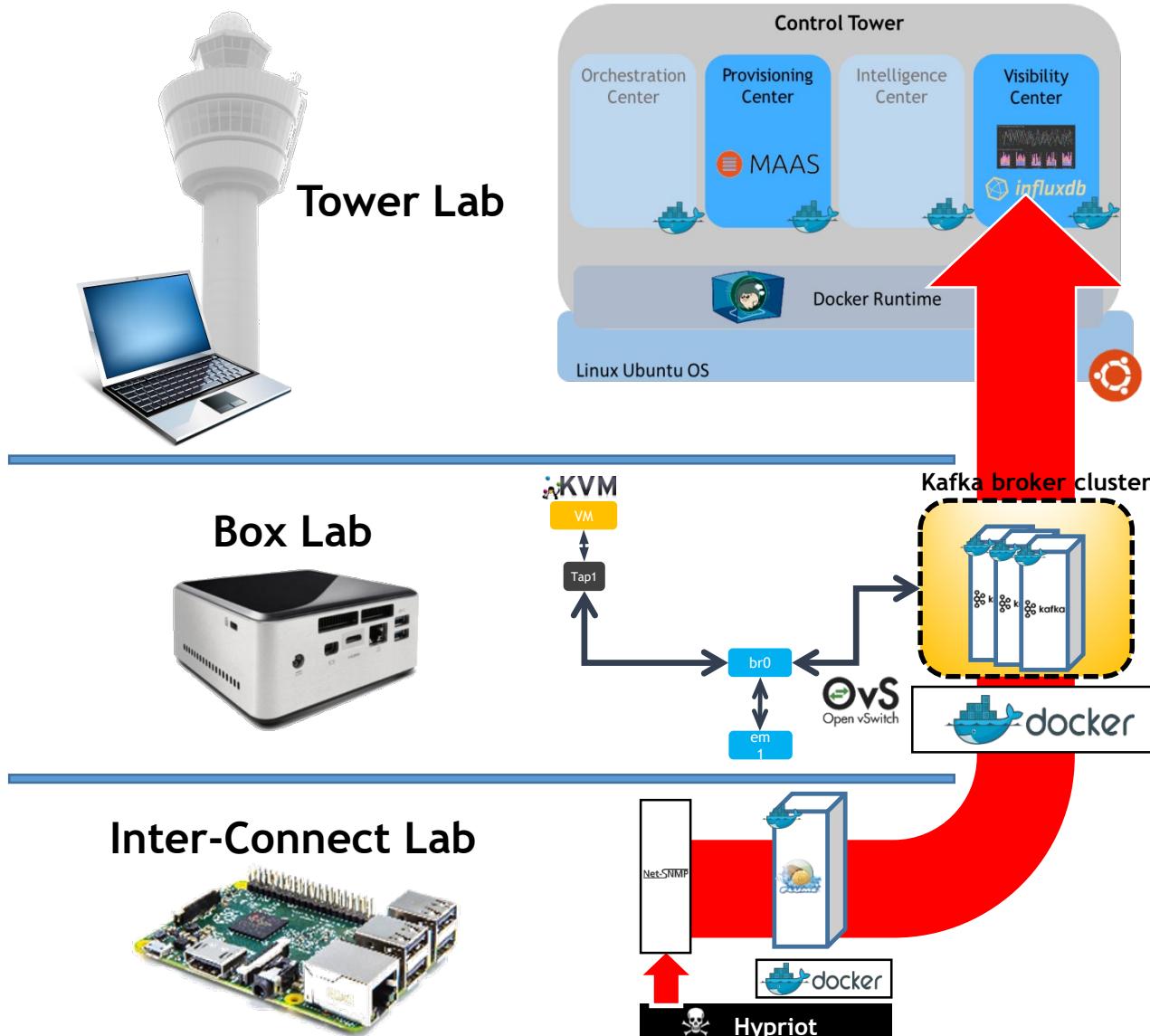


<https://github.com/SmartX-Labs/SmartX-Mini-MOOC>

Tower Lab: Concept



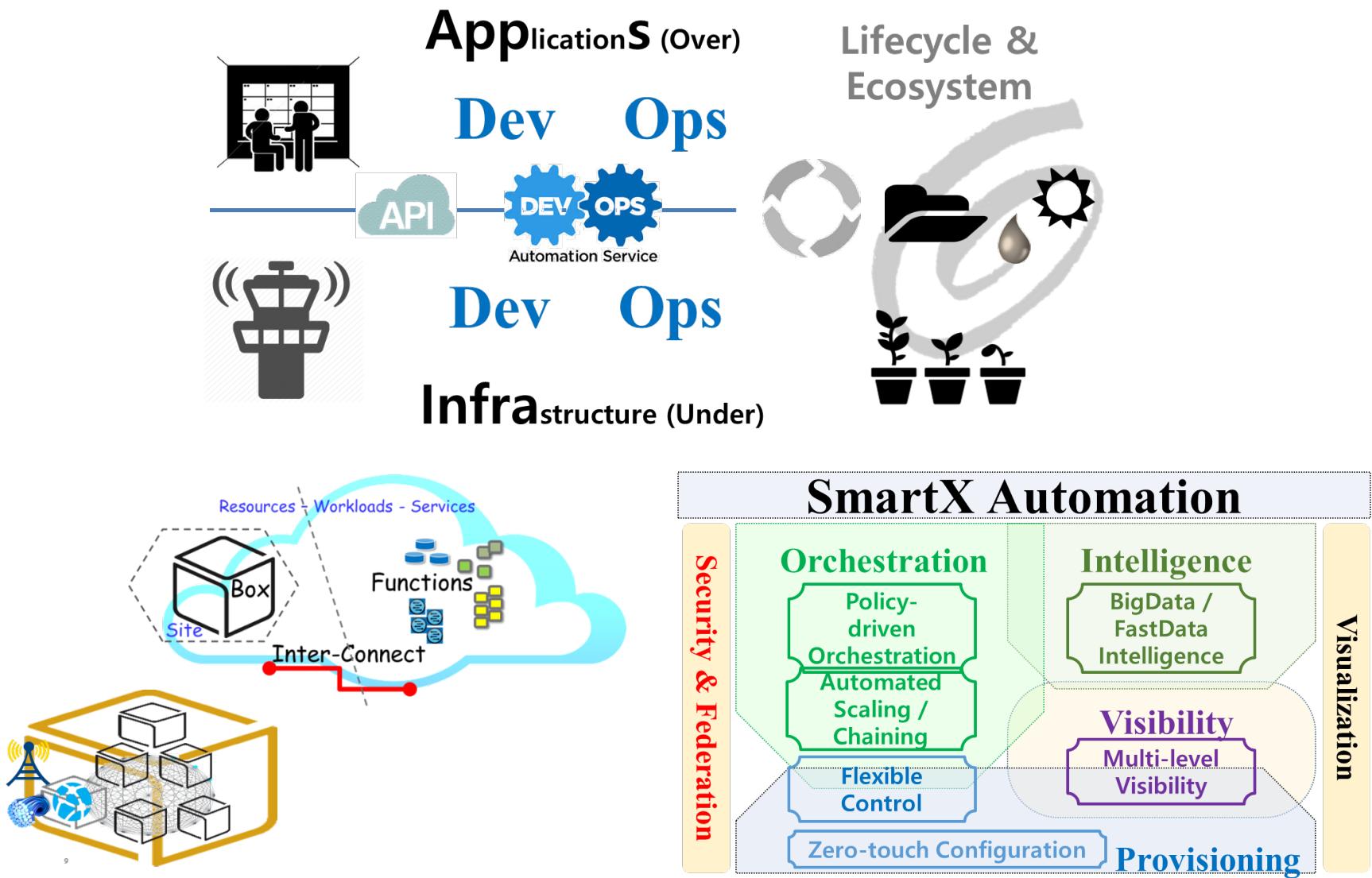
SmartX Labs #1~#3: Relationship



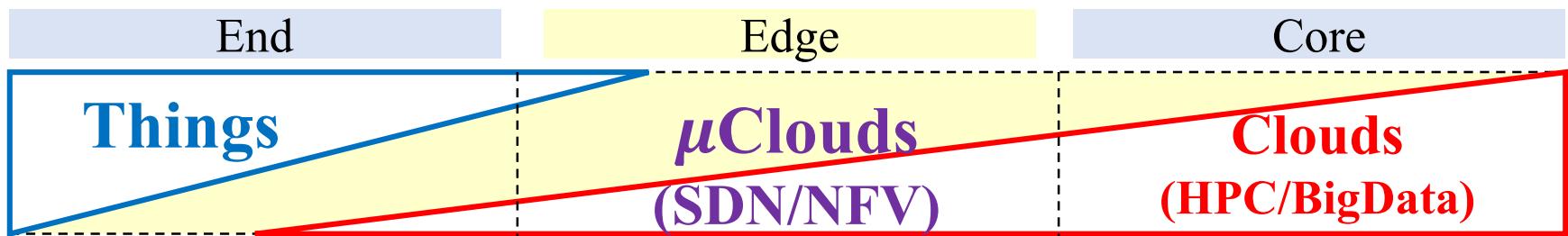
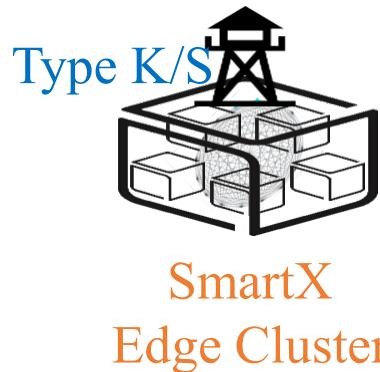
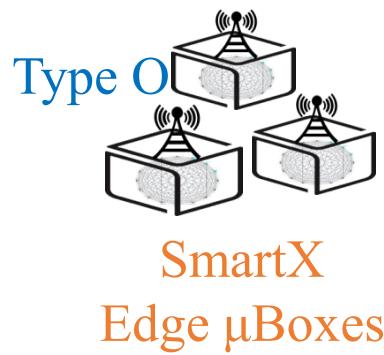
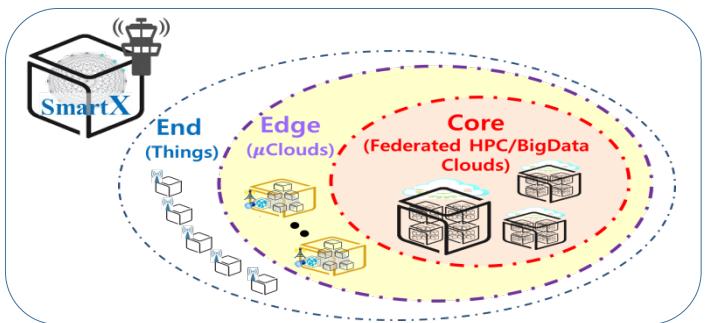
Theory



SmartX Automation Framework

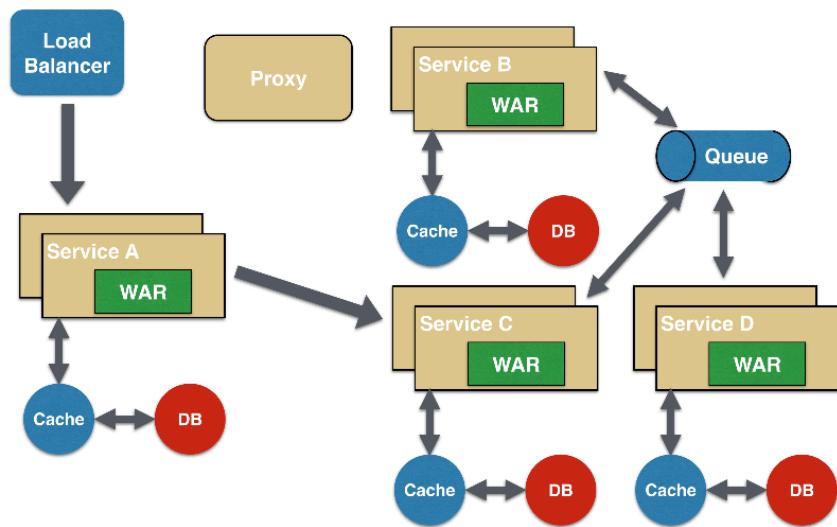
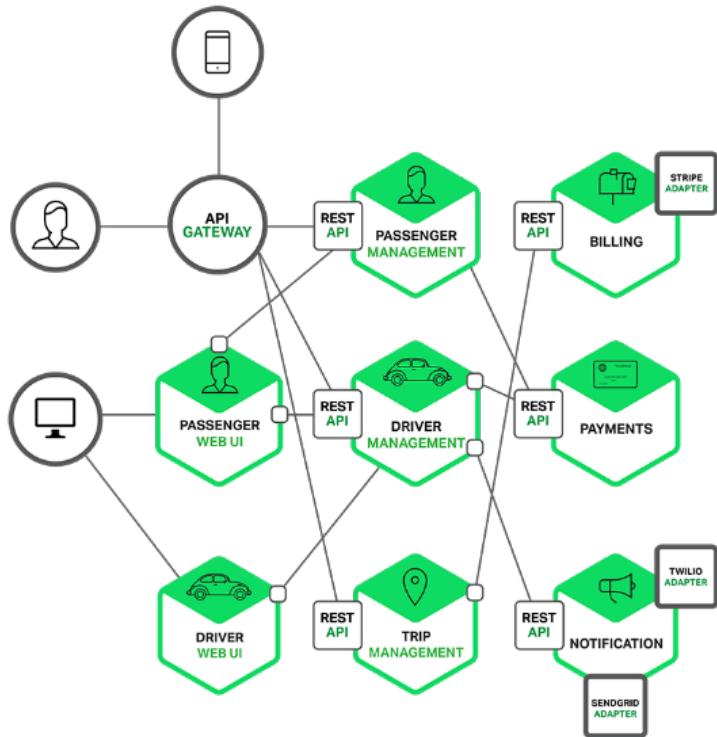


SmartX Composable Playground & Boxes



Container-based MSA (MicroServices Architecture)

- Software development technique based on **Collection of loosely coupled small-size services (i.e., functions)**
- Fine-grained services and lightweight protocols to improve modularity, create applications easier, and helps resiliency against architecture erosion



Visibility: TSDB (Time Series Database)



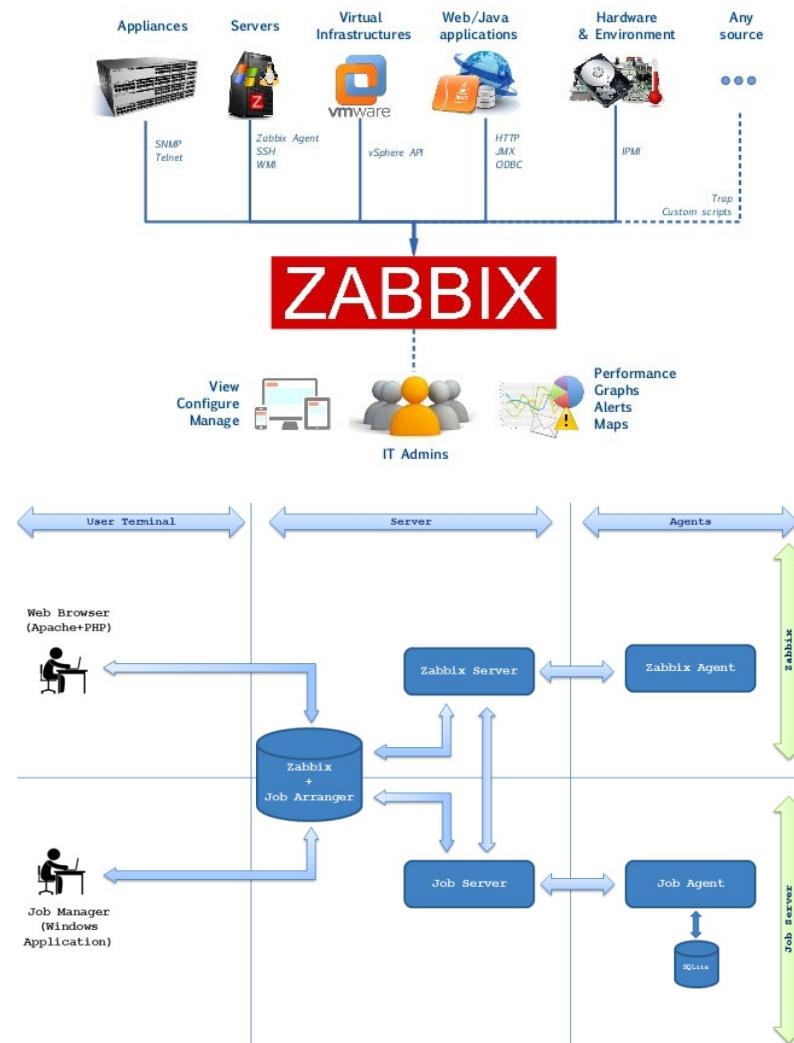
- Time series data is arrays of numbers indexed by time.
- In some fields these time series are called profiles, curves, or traces.



Visibility: Zabbix Distributed Monitoring

- Adopts a flexible notification mechanism
- User can configure & watch graph easily via Web GUI
- Consists of structured server and client
 - Client collects the monitoring data and send it to the Zabbix server
 - Server visualizes the data that is collected by the Zabbix Agent

ZABBIX



Zabbix Server Agent structure

Ubuntu boot procedure

Ubuntu boot up phases

1. BIOS

- ▶ When the computer begins execution, it starts by executing the firmware, and obtain the boot loader.

2. Boot loader

- ▶ The job of the boot loader is to begin the next phase, loading the kernel and an initial ram disk filesystem.

3. Kernel

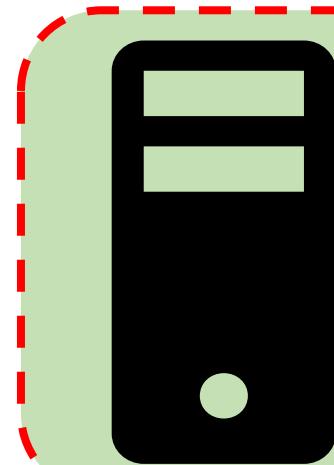
- ▶ The kernel launches the init script inside the initrd file system, which loads hardware drivers and finds the root partition.

4. Upstart

- ▶ After the kernel is running, the remainder of the operating system is brought online.

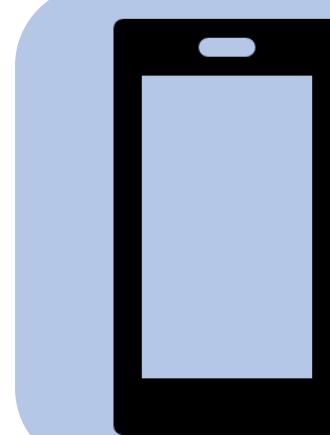
Remote OS installation on machines

Remote OS installation targets



Bare metal

- OS absent in the hardware
- For remote OS installation, the host doesn't have a decision-making power.



Mobile device

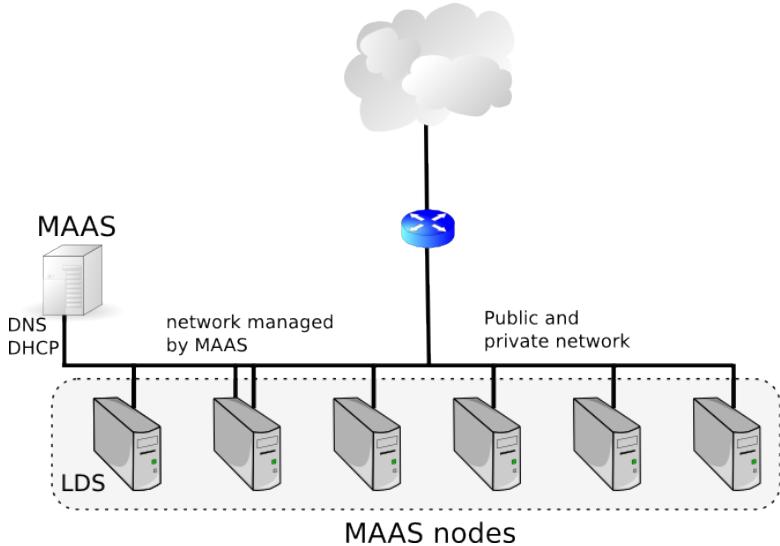
- OS already activated in the hardware
- From the standpoint of user, OS is installed automatically by host.



**MAAS(Metal As A Service) is
better suited to the bare metal.**



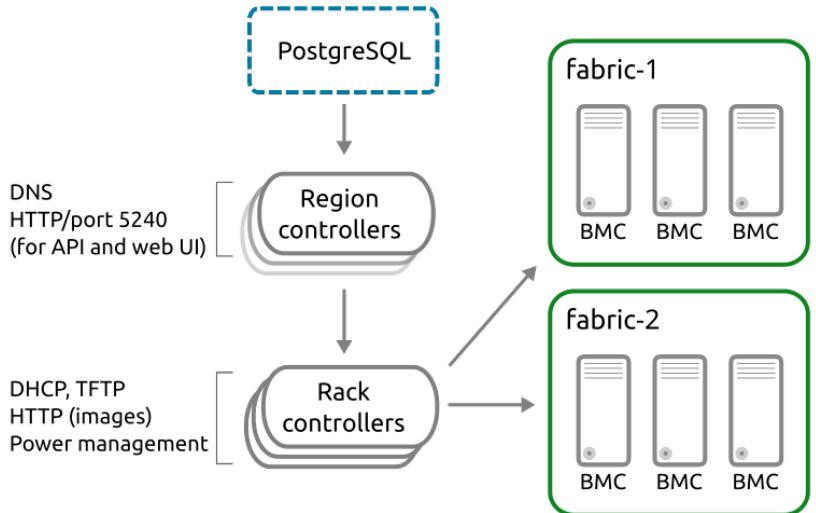
Provisioning: Ubuntu MAAS (Metal as a Service)



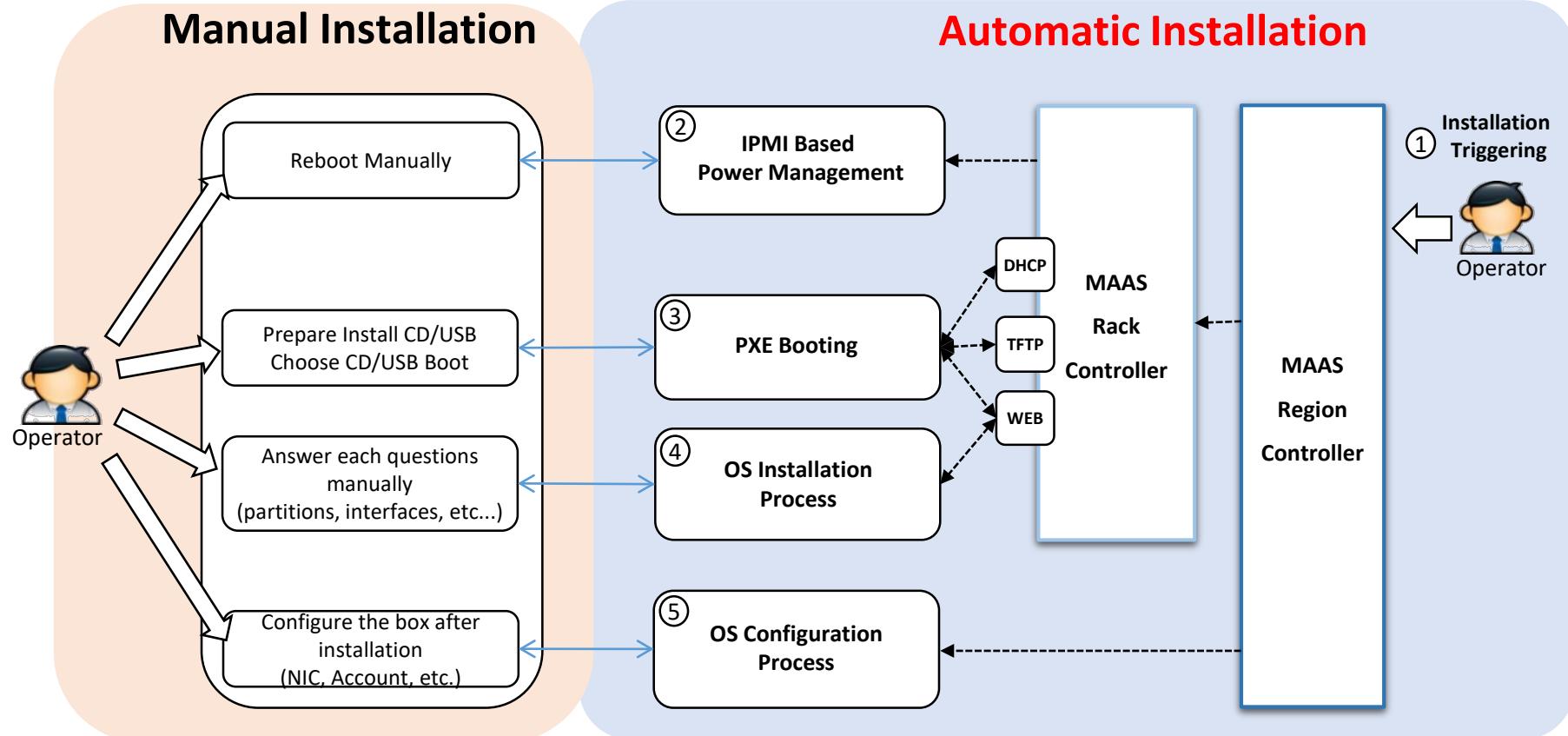
MAAS Controller Architecture

- Region Controller: Deals with operator requests
- Rack Controller: Provide the high bandwidth services to multiple server racks + Cache OS install images

- Bear-metal machines can be quickly provisioned and destroyed; MAAS provides management of a large number of physical machines by creating a single resource pool
- MAAS can act as a standalone PXE services, provides Web GUI, supports various Linux distribution installation, ...



MAAS: OS Automated Installation



Warning!

Box Hardware Requirements for Automated Installation

- *IPMI, Intel AMT, IBM HMC, ...
- PXE bootable with DHCP option
- Two Ethernet interfaces

*IPMI: The intelligent Platform Management Interface. Remote hardware health monitoring and management system that defines the interfaces for use in monitoring

MAAS: Remote OS installation process

Lab #3: Tower 14

Remote OS installation Process

1. DHCP server contacted
2. Kernel, initrd received over TFTP
3. Machine boots
4. Initrd mounts a squashfs image over HTTP



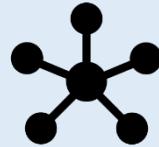
Enlistment

5. cloud-init runs enlistment scripts
6. Machine shuts down



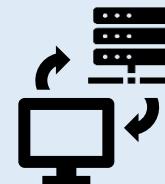
Commissioning

5. cloud-init runs commissioning scripts
6. Machine shuts down



Deployment

5. cloud-init triggers deployment
 - Curtin installation script run
 - Squashfs image placed on disk



Practice



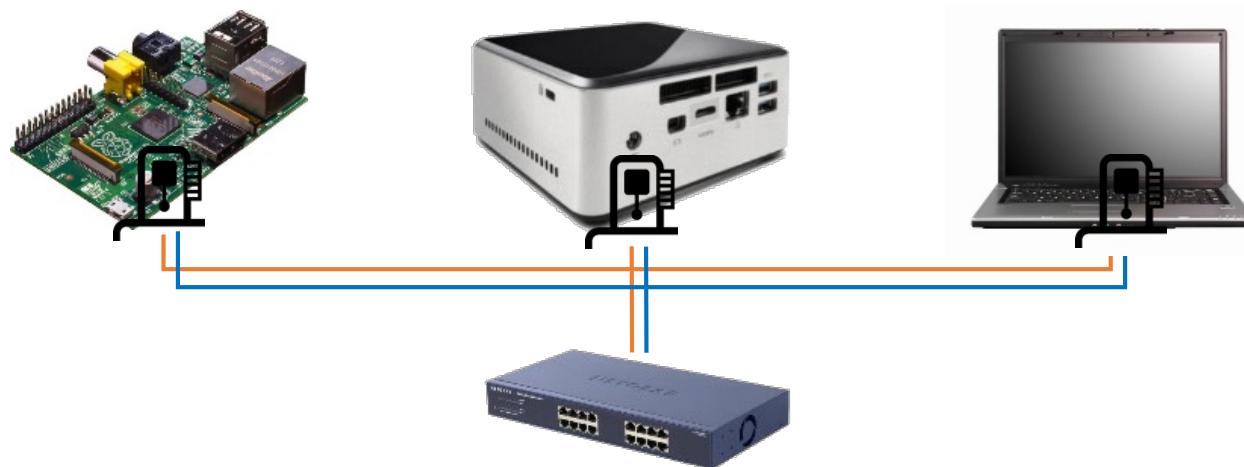
#0 Lab Preparation (1/2)

Wired connection

NAME: Raspberry Pi Model B (Pi)
CPU: ARM Cortex A7 @900MHz
CORE: 4
Memory: 1GB
SD Card: 32GB

NAME: NUC5i5MYHE (NUC PC)
CPU: i5-5300U @2.30GHz
CORE: 4
Memory: 16GB DDR3
HDD: 94GB

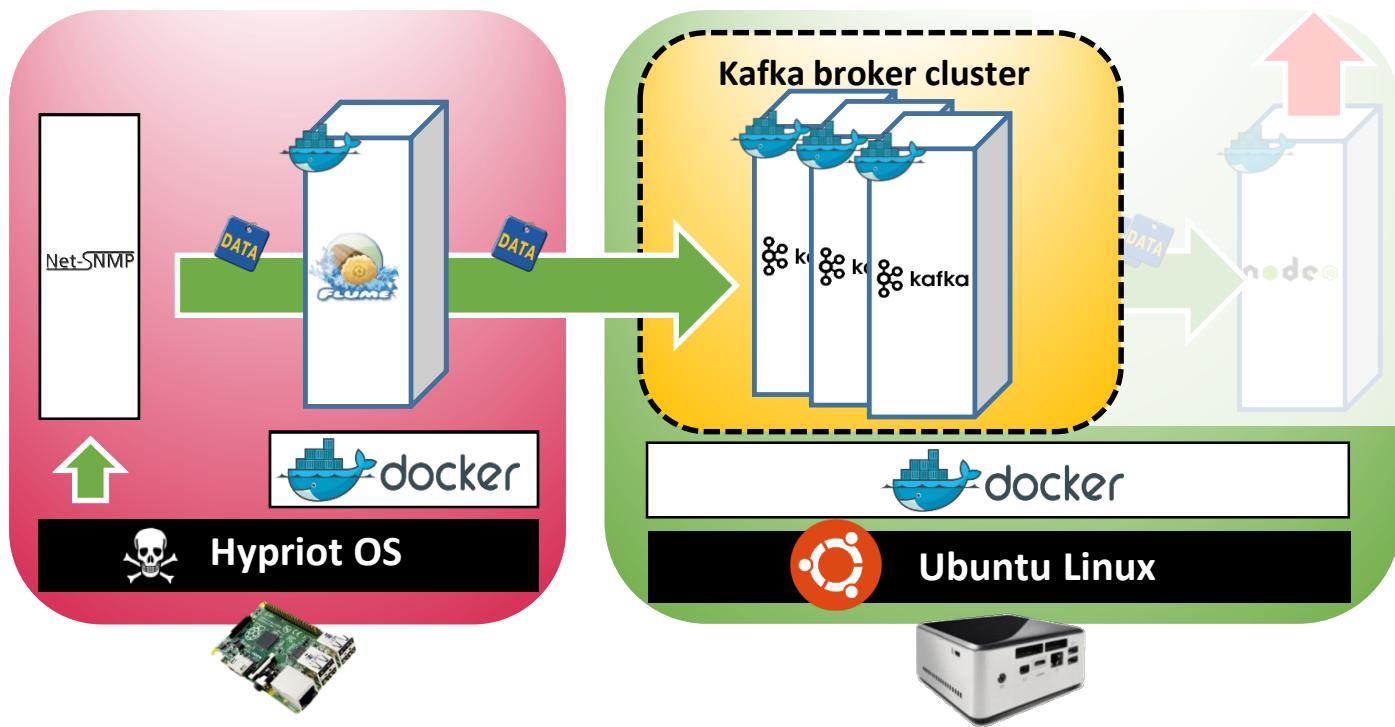
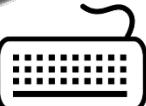
NAME: NT900X3A
CPU: i5-2537U @1.40GHz
CORE: 2
Memory: 4GB DDR3
HDD: 128GB



NAME: netgear prosafe 16 port gigabit switch(Switch)
Network Ports: 16 auto-sensing 10/100/1000 Mbps Ethernet ports

#0 Lab Preparation (2/2)

- Verify Inter-Connect Lab's configuration

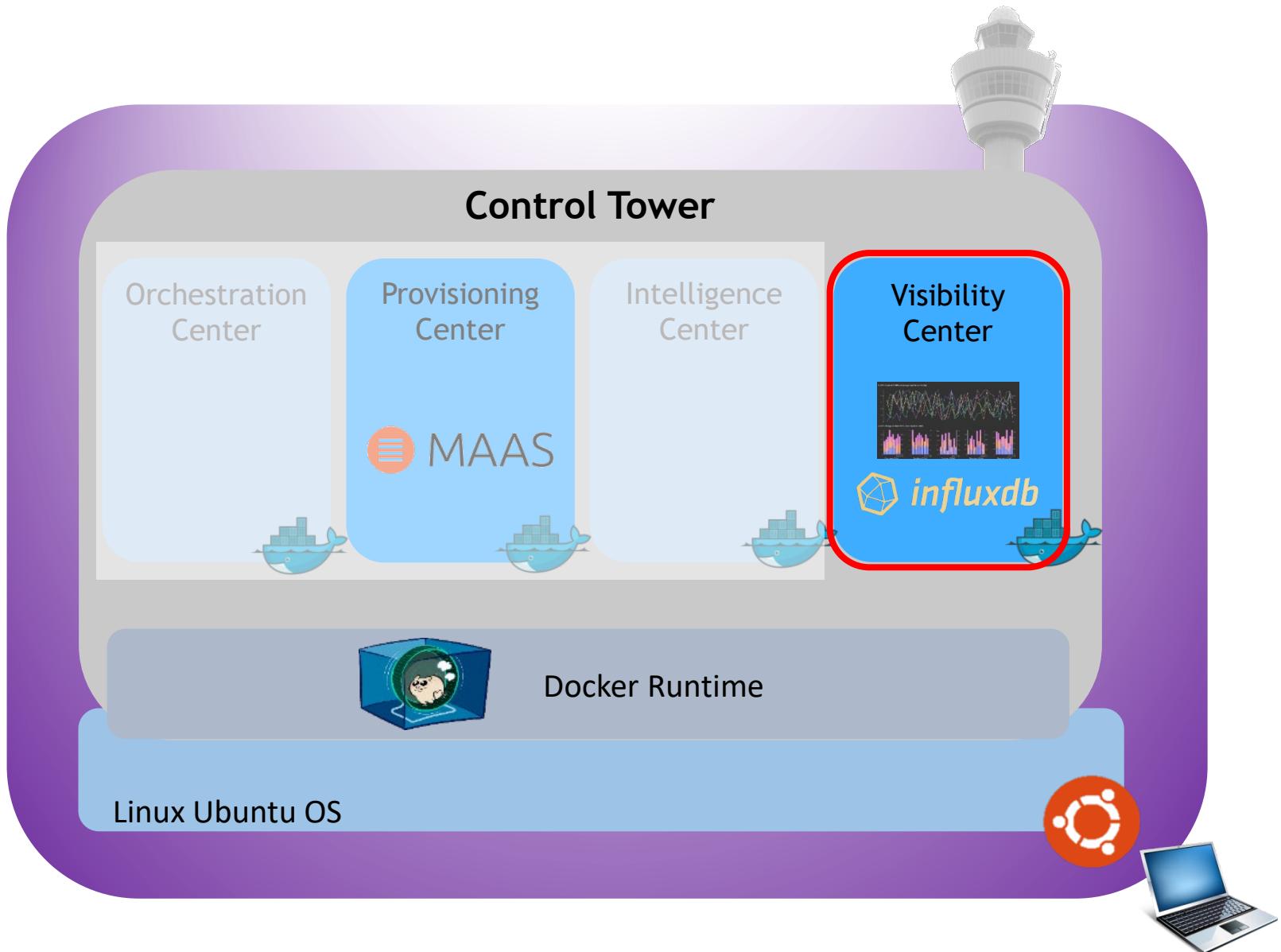


Are they working?

If you can see logs of resource status on console consumer, go ahead!

Visualization of Resource Visibility

Lab #3: Tower 18



#1 Run InfluxDB & Chronograf Containers on NUC



- Run InfluxDB Container

```
$ sudo docker run -d --name=influxdb --net=host influxdb:1.7
```

- Make and run Chronograf container

```
$ sudo docker run -p 8888:8888 --net=host chronograf --influxdb-  
url=http://<NUC IP>:8086
```

```
netai@master1:~$ sudo docker run -d --name=influxdb --net=host influxdb:1.7
```

```
netai@master1:~$ sudo docker run -p 8888:8888 --net=host chronograf --influxdb-  
url=http://210.        :8086
```

Type without a space!



#2 Install python packages:for python Kafka consumer

- Install python-pip

```
$ sudo apt-get install -y libcurl3 openssl curl  
$ sudo apt-get install -y python2.7 python-pip  
$ sudo apt-get install -y python3-pip
```

- Install python package

```
$ sudo pip install requests  
$ sudo pip install kafka-python  
$ sudo pip install influxdb  
$ sudo pip install msgpack
```



#3 Broker to InfluxDB code:Modify & Run python code

- Open 'broker_to_influxdb.py' code

```
$ vi ~/SmartX-mini/ubuntu-kafkatodb/broker_to_influxdb.py
```

```
cmd = "curl -XPOST 'http://localhost:8086/query' --data-urlencode 'q=CREATE DATABASE 'Labs''"
subprocess.call([cmd], shell=True)

timeout = 100
actual_data=[]

consumer = KafkaConsumer('resource',bootstrap_servers=[[REDACTED]:9091])
partitions = consumer.poll(timeout)
while partitions == None or len(partitions) == 0:
    consumer = KafkaConsumer('resource', bootstrap_servers=[[REDACTED]:9091])
    message = next(consumer)
    print(message.value)
```

Modify to your nuc IP Address

Modify to your nuc IP Address

- Run python code

```
$ sudo sysctl -w fs.file-max=100000
```

```
$ ulimit -S -n 2048
```

```
$ python ~/SmartX-mini/ubuntu-kafkatodb/broker_to_influxdb.py
```



#4 Configure Chronograf Dashboard (1/4)

- Open Web browser and connect to Chronograf Dashboard

<http://<NUC IP>:8888>

The screenshot shows the Chronograf dashboard interface. On the left, there is a sidebar with various icons: a gear, a magnifying glass, a checkmark, a grid, an alert triangle, a hand, a crown, and a wrench. The main area has two large panels: "Alert Events per Day – Last 30 Days" and "Alerts – Last 30 Days". Both panels display the message "No Results". Below these panels, a message states "The current source does not have an associated Kapacitor instance" and a blue button labeled "Configure Kapacitor". To the right, there is a "News Feed" section with a single article about InfluxDays SF 2018, followed by a "Getting Started" section with a purple box titled "Welcome to Chronograf!", a "TICK Stack" section, and a "Guides" section with links to "Create a Dashboard", "Create a Kapacitor Alert", "Configure Kapacitor Event Handlers", "Transition from InfluxDB's Web Admin Interface", and "Dashboard Template Variables".

#4 Configure Chronograf Dashboard (2/4)



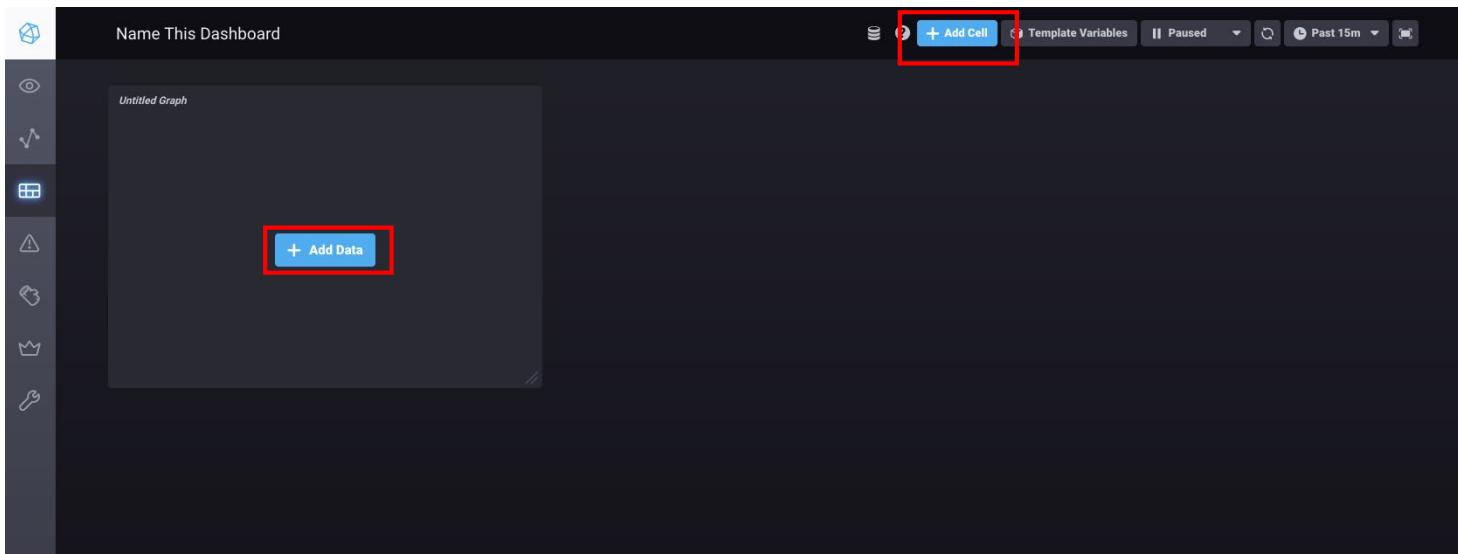
- Create Dash board

A screenshot of the Chronograf interface, specifically the 'Dashboards' section. The left sidebar has several icons: a gear (Metrics), an eye (Logs), a checkmark (Metrics), a grid (Dashboards, highlighted with a red box), a warning triangle (Logs), a hand (Metrics), a crown (Logs), and a wrench (Metrics). The main area is titled 'Dashboards' and shows '0 Dashboards'. It includes a search bar ('Filter by Name...'), an 'Import Dashboard' button, and a prominent blue '+ Create Dashboard' button. Below the button, a message says 'Looks like you don't have any dashboards'.

#4 Configure Chronograf Dashboard (3/4)



- Add Cell & Data



#4 Configure Chronograf Dashboard (4/4)



- Add Data

The screenshot shows the Chronograf interface with the following details:

- Queries Tab:** A query editor window displays the following SQL-like query:

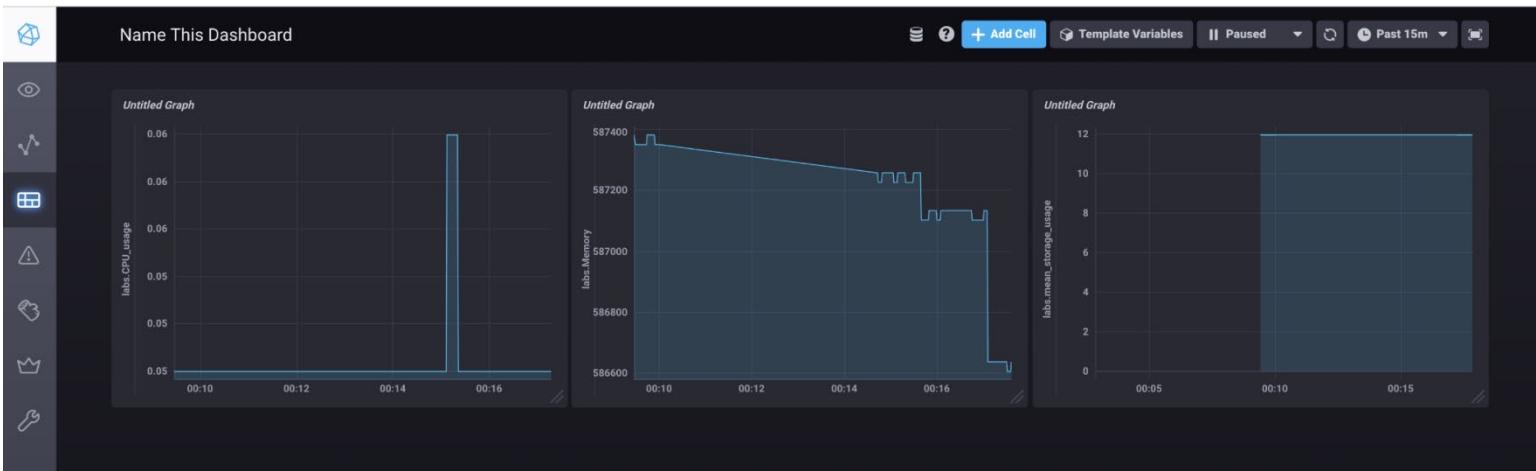
```
SELECT "Memory" FROM "Labs"."autogen"."la...  
SELECT "Memory" FROM "Labs"."autogen"."labs" WHERE time > :dashboardTime;
```

A green checkmark icon with a red border is located in the top right corner of the query editor.
- Success Message:** Below the query editor, a green checkmark icon with the text "Success!" is displayed.
- DB.RetentionPolicy:** On the left, there's a tree view of database retention policies. One node under "Labs.autogen" is expanded, showing sub-nodes like "host - 1" and "region - 1". Other nodes include "str3", "str4", "str7", "str8", "test", and "timestamp".
- Measurements & Tags:** This section contains a search bar labeled "Filter" and a list of measurements and tags. Under "Fields", the "Memory" measurement is selected.
- Fields Section:** This section lists various fields: CPU_Load, CPU_usage, storage_usage, timestamp, mean, median, count, min, max, sum, first, last, spread, stddev, and stddev.
- Buttons:** At the bottom right of the interface, there is a red-bordered "Apply" button.

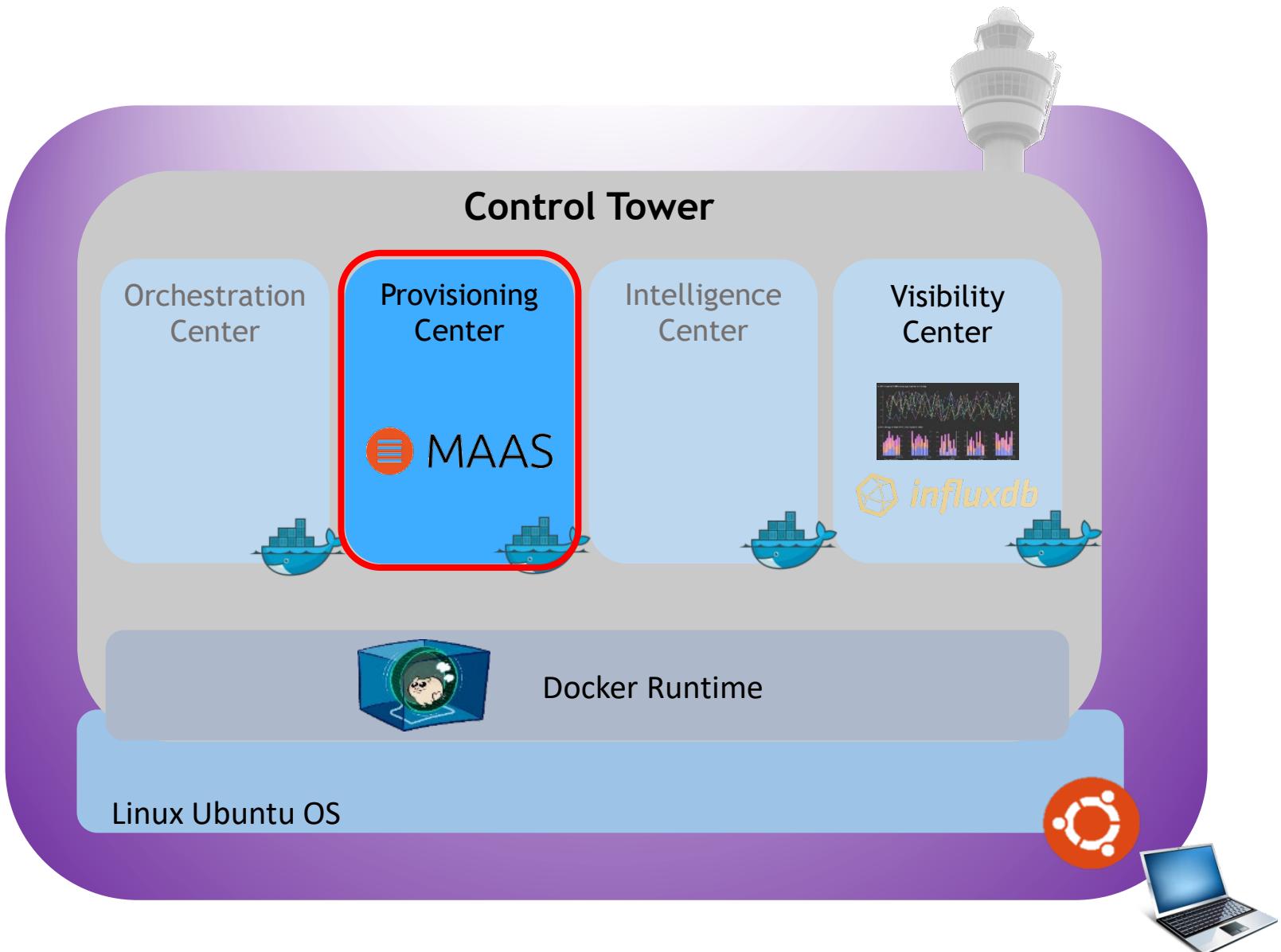


#5 – Check Chronograf Dashboard

- We can see the changes of values from database

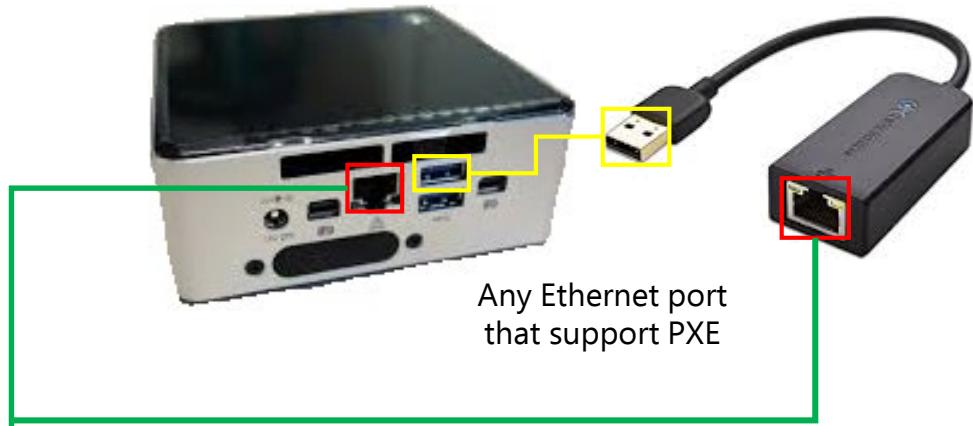
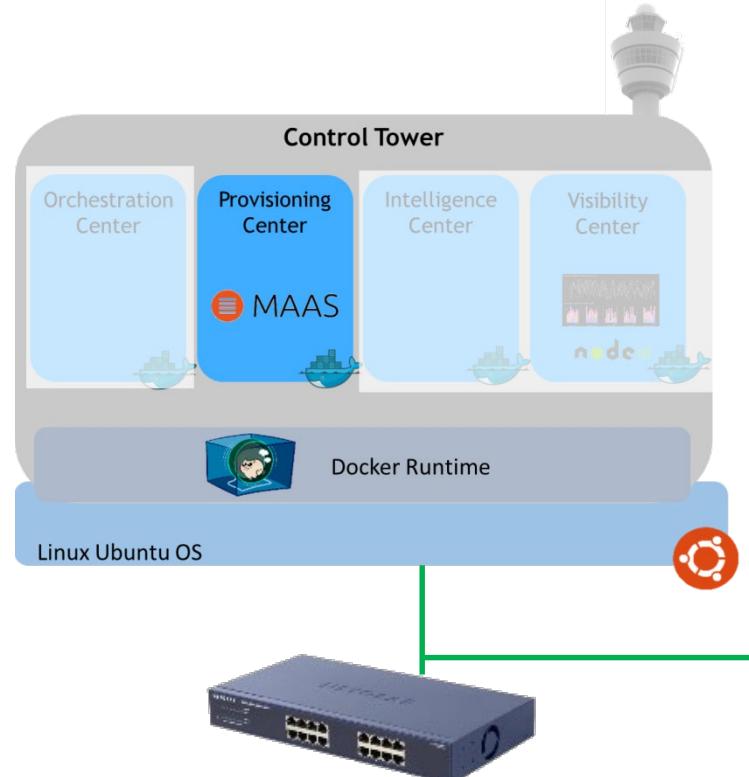


Automated Provisioning & Visibility



#6-1 OS manual Installation: Establish physical interconnect

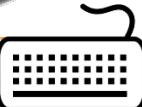
Management & Control



Requirements for manual Installation

- DHCP PXE bootable
- USB to ethernet connector

Note: Typical NUC does not have a IPMI port.



#6-2 Install MAAS server

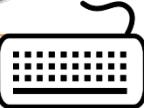
- From target NUC, go to BIOS, turn on PXE and set network boot priority
- NUC reboot (to apply BIOS changes)
- Install MAAS server

```
$ sudo apt update  
$ sudo apt install maas
```

- Initiate MAAS server
- \$ sudo maas init

- Login to the MAAS UI at:
 <http://<your.maas.ip>:5240/MAAS>
- From the MAAS UI, you need to make user configurations
 - *Region name
 - Ubuntu images
- Turn on DHCP
 - Go to the “Subnets” tab, select the VLAN for which you want to enable DHCP
 - From the “Take action” button select “Provide DHCP”

*region : Organizational unit. Contains all information about all machines running



#6-3 Enlist, Commission and deploy the box

- Enlist the NUC
 - Set all the servers to PXE boot
 - NUC reboot (When hardware is initialized, all software operations stop)
 - Check the NUC appear in MAAS
 - If the NUC does not support IPMI based BMC, edit them and enter their BMC details
- Commission the NUC
 - Go to "machine interface", "configuration" and set "power type" to "manual"
 - From the "take action" button, select "commission"
 - NUC reboot (When a new kernel is installed, the box must be rebooted as to prevent the removed kernel from being loaded which will halt the NUC operation)
- Deploy the NUC
 - From the "take action" button, choose "deploy" and click "view this page"
 - From SSH keys, choose source and click upload
 - Set "id_rsa.pub" as a public key and click "import"

#6-4 Deploying the box



- Create SSH key

```
$ sudo ssh-keygen
```

```
$ sudo cat ~/.ssh/id_rsa.pub
```

Copy the outcome , press “upload” and paste it on “User ID” and import

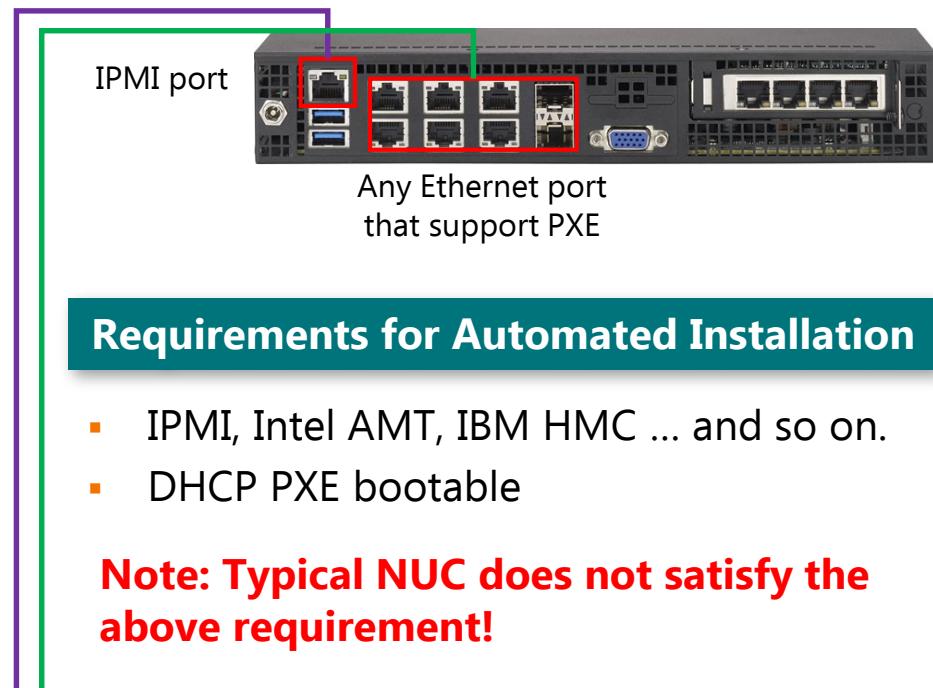
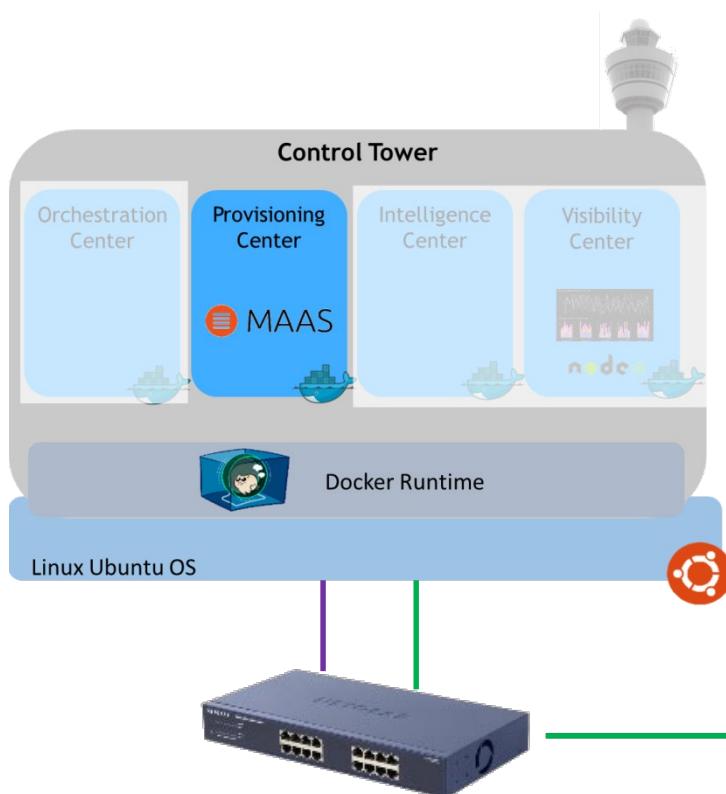
- Deploy

- From “take action”, choose “deploy”
- OS remote Install complete

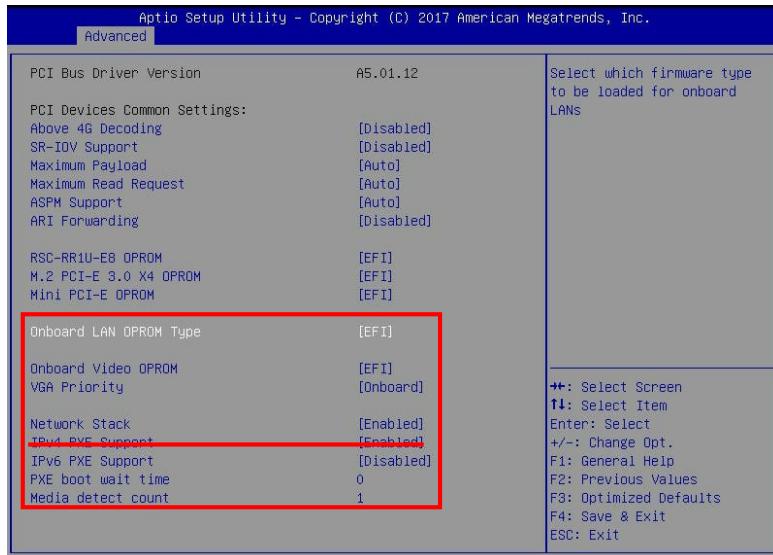
Appendix

Remaining Lab practice requires
special Box resources with
**IPMI or similar Remote Power
Management, PXE Boot support**

#7-1 OS Automated Installation: Establish physical interconnect

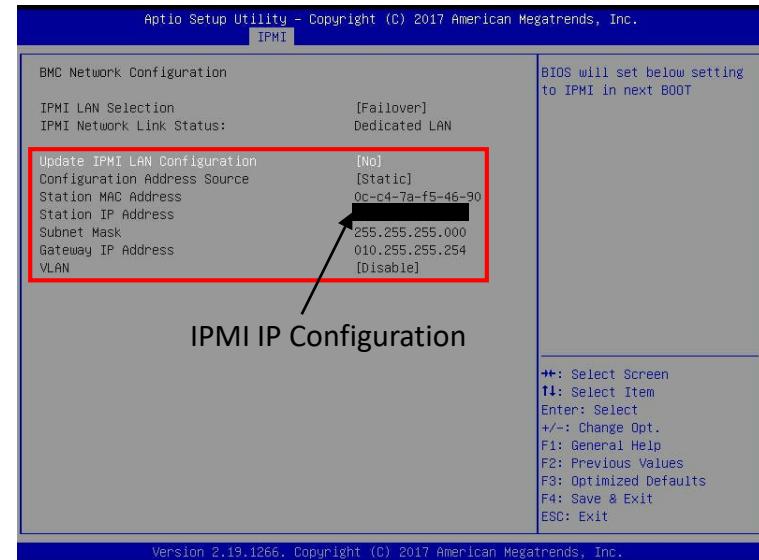


#7-2 OS Automatic Installation: BIOS Configuration



<BIOS PXE Configuration>

- After then **Save Configuration and Exit**
- And then the PXE booting sequence is stated



<BIOS IPMI Configuration>

#8-1 OS Automatic Installation:Enlistment



Booting under MAAS direction...

Check this message

```
[ OK ] Reached target Local Encrypted Volumes.
[ OK ] Started Commit a transient machine-id on disk.
[ OK ] Started ebtables ruleset management.
[ OK ] Started Network Time Synchronization.
[ OK ] Listening on Load/Save RF Kill Switch Status /dev/rfkill Watch.
[ OK ] Reached target System Time Synchronized.
[ 53.726408] cloud-init[570]: Cloud-init v. 18.3-9-g2e62cb8a-0ubuntu1~18.04.2 running 'init-local' at Tue, 16 Oct 2018 12:20:53 +0000. Up 53.25 seconds.
[ OK ] Started Initial cloud-init Job (pre-networking).
[ OK ] Reached target Network (Pre).
Starting Network Service...
[ OK ] Started Network Service.
Starting Wait for Network to be Configured...
Starting Network Name Resolution...
[ OK ] Started Network Name Resolution.
[ OK ] Reached target Host and Network Name Lookups.
[ OK ] Reached target Network.
[ *** ] A start job is running for Wait for Network to be Configured (2min 4s / no limit)
```

<Succeed PXE booting>

<Enlistment after PXE booting>

FQDN MAC	Power	Status	Owner	Cores	RAM (GB)	Disk	Storage (GB)
fleet-lynx.maas	Unknown	New		0	0.0	0	0.0
K1-GJ1-Cube1.maas	On	Ubuntu 18.04 LTS	netcs	8	32.0	1	512.1
K1-GJ1-Cube2.maas	On	Ubuntu 18.04 LTS	netcs	8	32.0	1	512.1
K1-GJ1-Cube3.maas	On	Ubuntu 16.04 LTS	netcs	8	32.0	1	512.1
K1-GJ1-Cube4.maas	On	Ubuntu 16.04 LTS	netcs	8	32.0	1	512.1

<After Enlistment (you can check on Web GUI) >



#8-2 OS Automatic Installation:Commissioning

The screenshot shows the MAAS Nodes interface for a machine named 'fleet-lynx.maas'. The 'Machine summary' tab is selected. In the hardware section, the CPU, Memory, and Storage status are all listed as 'Unknown'. Below each status, there is a note indicating 'No data received'. On the right side of the screen, a dropdown menu under 'Take action' has the 'Commission' option highlighted with a red box. A callout text 'Click this button to commission' points to the 'Commission' button.

Click this button to commission

All the hardware information shown as 'Unknown' before Commissioning

The screenshot shows the MAAS Nodes interface for the same machine 'fleet-lynx.maas'. In the commissioning configuration section, three checkboxes are present: 'Allow SSH access and prevent machine from powering off', 'Retain network configuration', and 'Retain storage configuration'. All three checkboxes are empty. Below this, there is a 'Hardware tests' section with a text input field containing 'smartctl-validate' and a 'Select scripts' link. At the bottom right of the commissioning dialog, there is a green 'Commission machine' button with a white border, which is also highlighted with a red box. A callout text 'Click this button to commission' points to this button.

Commission

Do not check anything

Cancel Commission machine

Click this button to commission

- It takes about 10 minutes



#8-3 OS Automatic Installation: Deployment (1/2)

You can see Ready state and power on/off state after commissioning.

Click this button to Deploy

You can see all the Hardware information After commissioning

Click this button to Deploy

- It is a **OS Installation procedure**
- It takes about 15 minutes

#8-3 OS Automatic Installation:Deployment (2/2)



```
Ubuntu 18.04.1 LTS fleet-lynx tty1
fleet-lynx login: _
```

<Complete Deploying (OS booted)>

fleet-lynx.maas

Deployed Power on check now

Machine state is changed to
'Deployed'

Take action

Machine summary

Interfaces

Storage

Commissioning

Hardware tests

Logs

Events

Configuration

<Complete Deploying (On Web GUI)>

Review

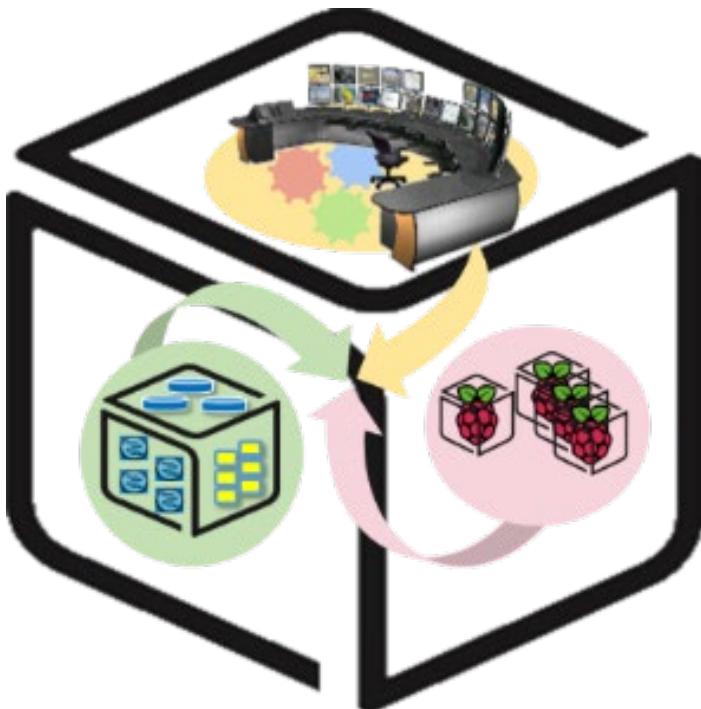


Lab Summary

With Tower Lab, you have experimented selected roles of Monitor/Control (관제) Tower

1. Visibility Center function to **enable ‘distributed monitoring’** over remote Boxes and to **store ‘monitoring information’** to time-size DB.
2. Provisioning Center function to **enable remote ‘installation & configuration (of OS and others)’** of distributed Boxes.

Thank You for Your Attention Any Questions?



mini@smartx.kr