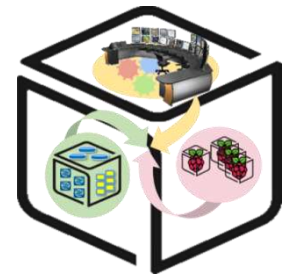


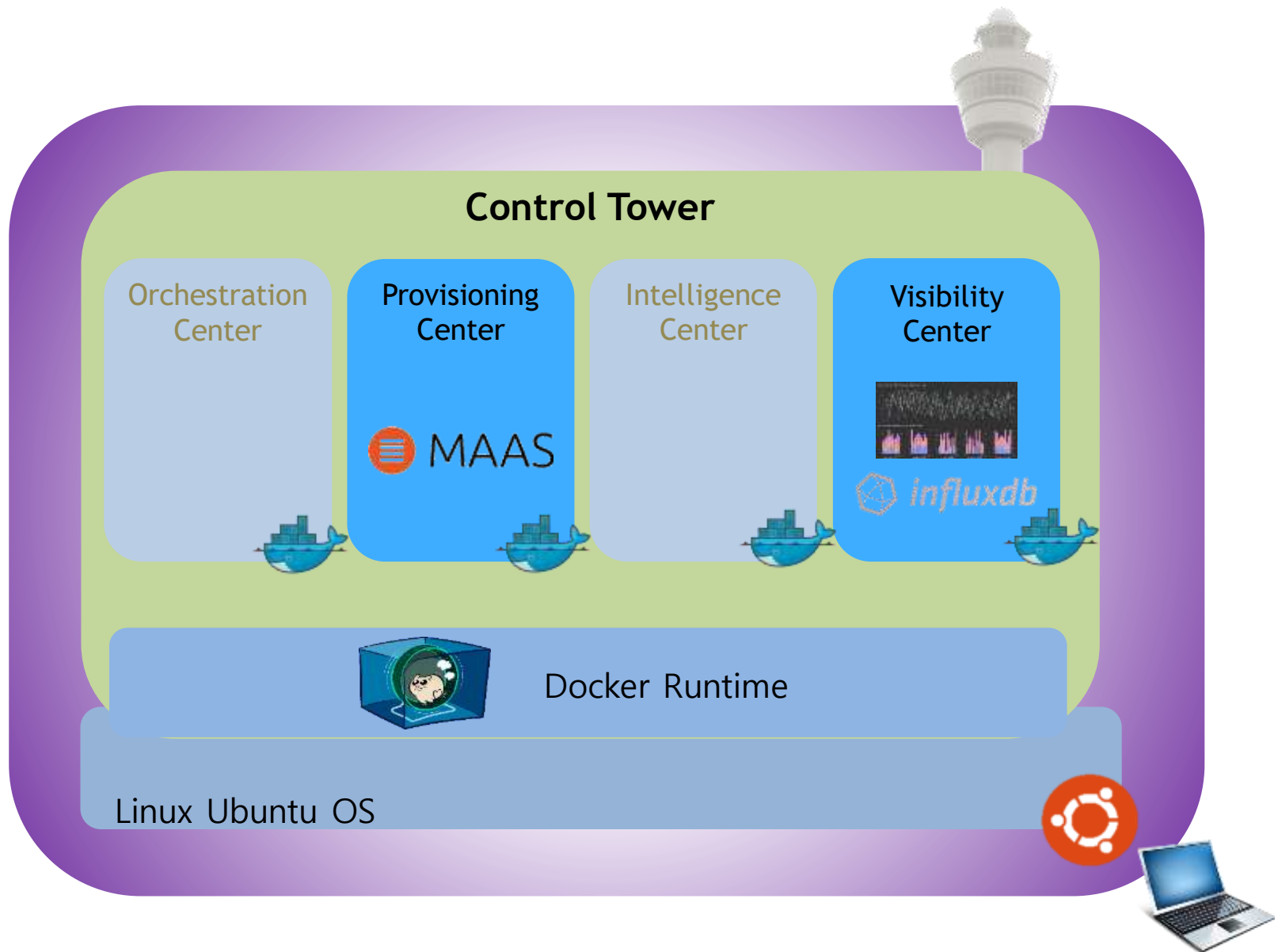
Computer Systems for AI-inspired Cloud **Theory & Lab.**

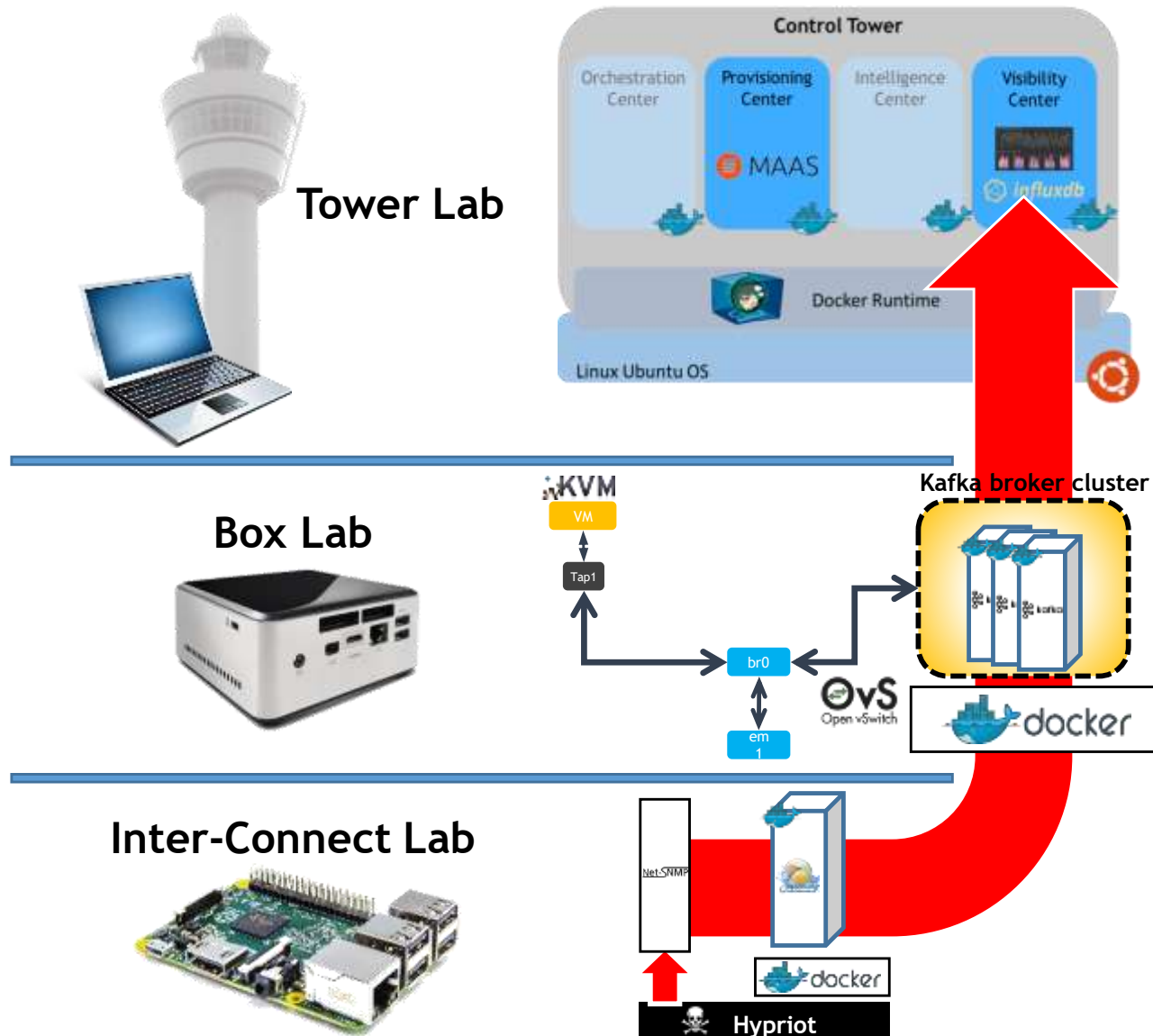
SmartX Labs – Mini (MOOC Selection)

Tower (Lab#3)



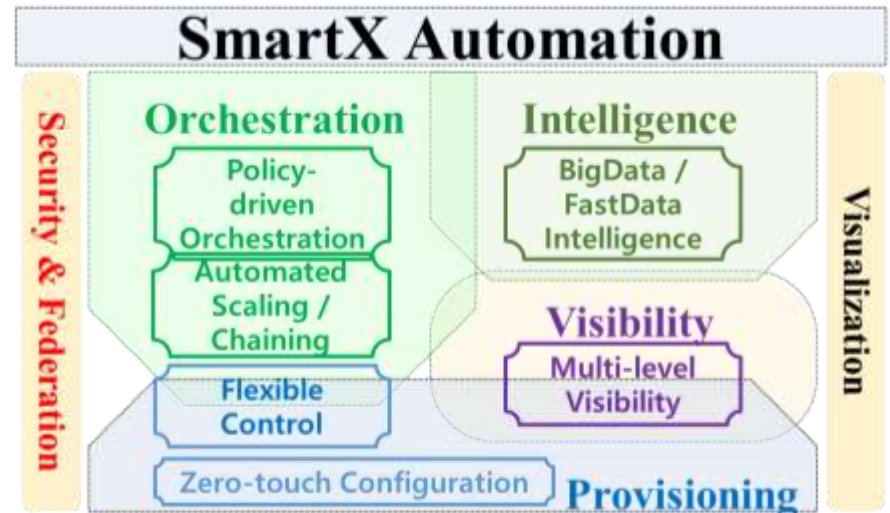
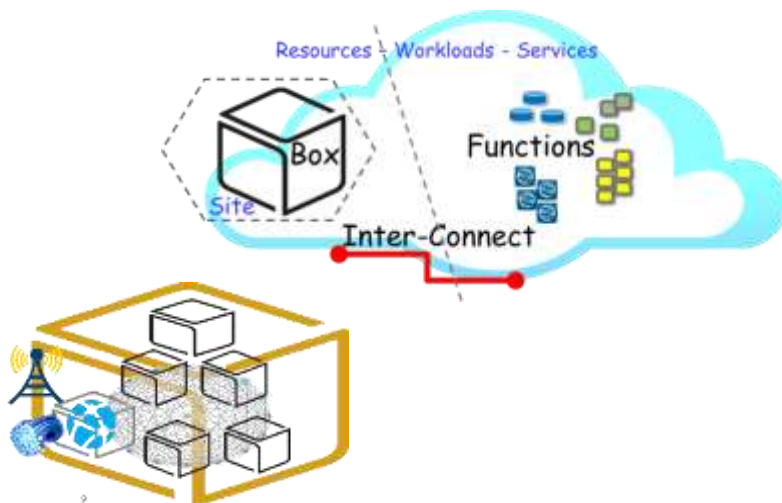
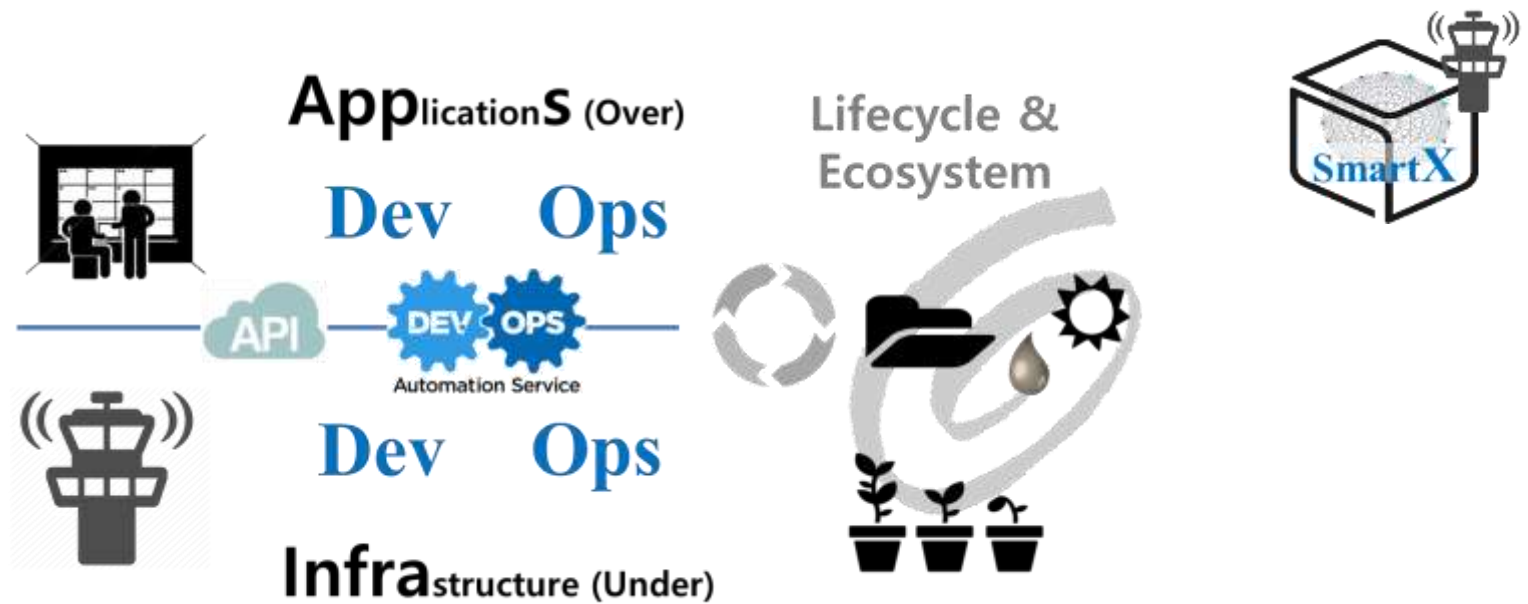
<https://github.com/SmartX-Labs/SmartX-Mini-MOOC>

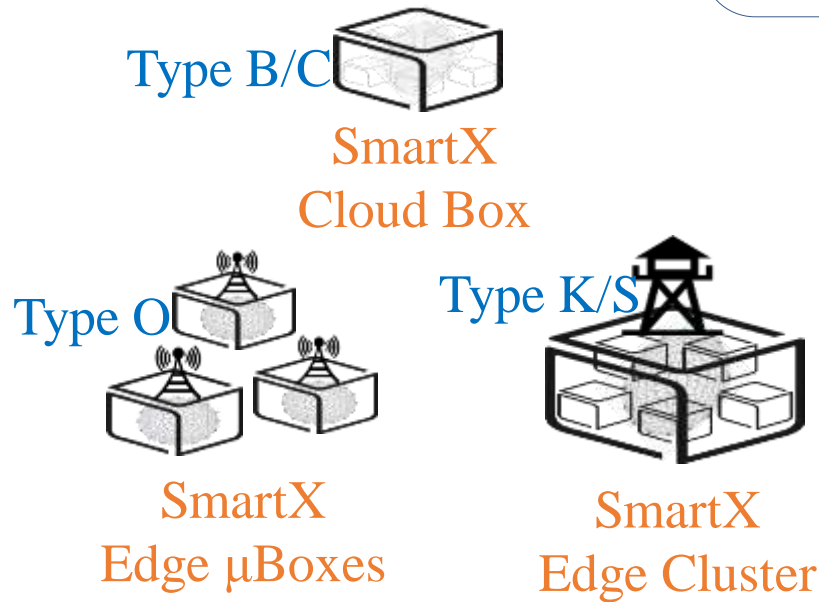
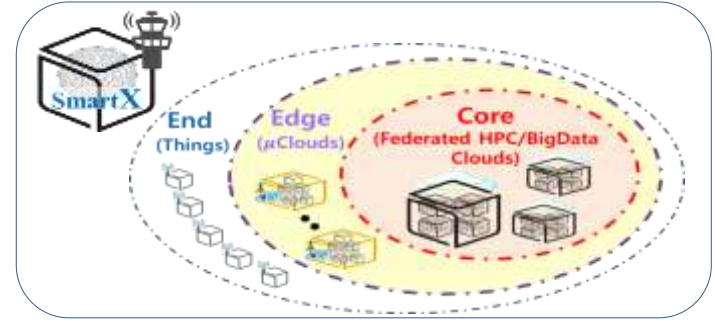




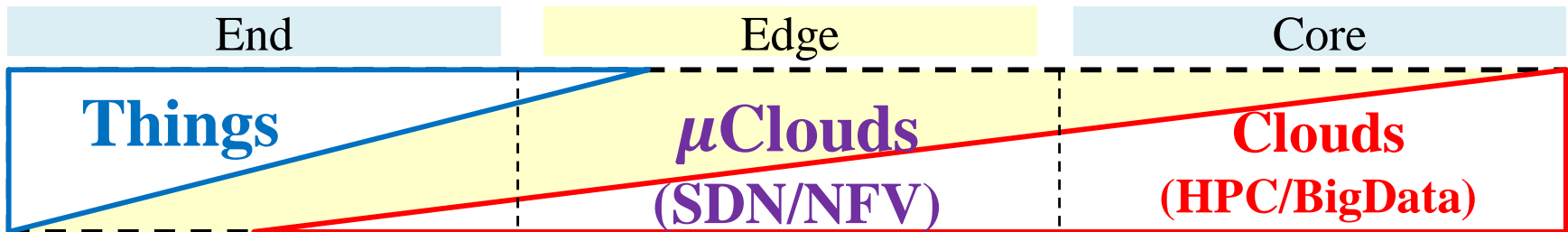


Lab Theory

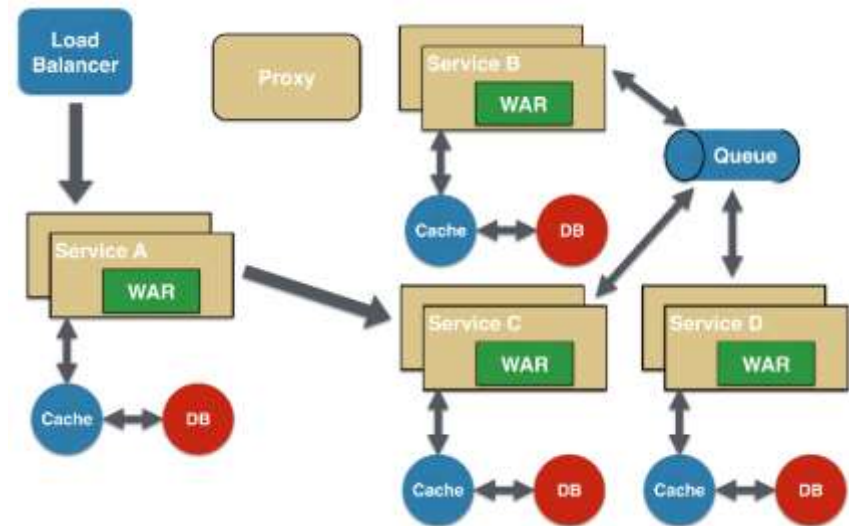
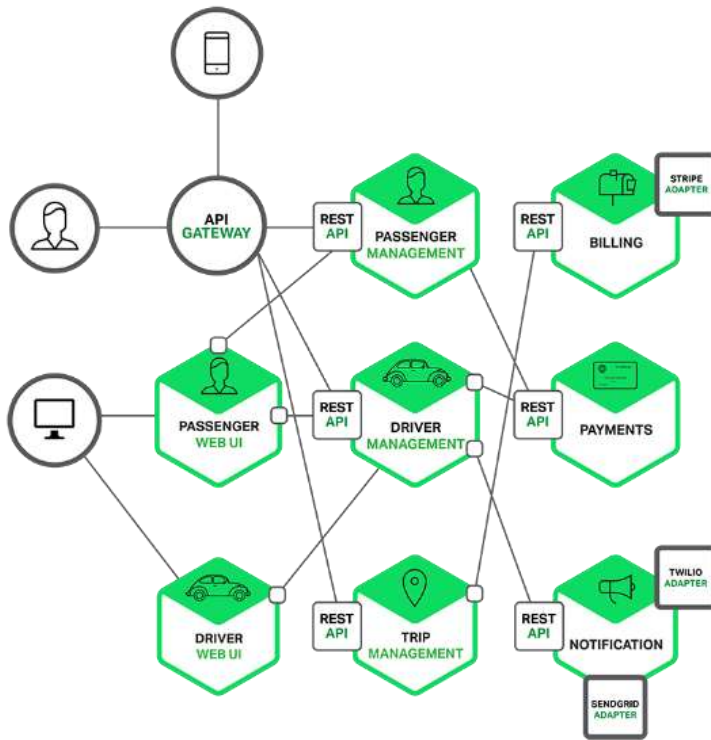




SmartX
DevOps Tower
Cloud
with DataLake

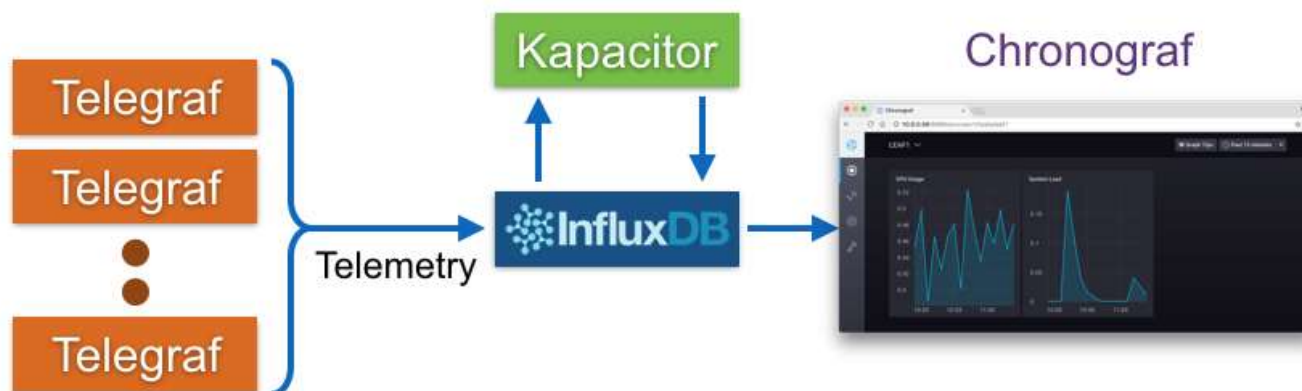


- Software development technique based on **Collection of loosely coupled small-size services (i.e., functions)**
- Fine-grained services and lightweight protocols to improve modularity, create applications easier, and helps resiliency against architecture erosion



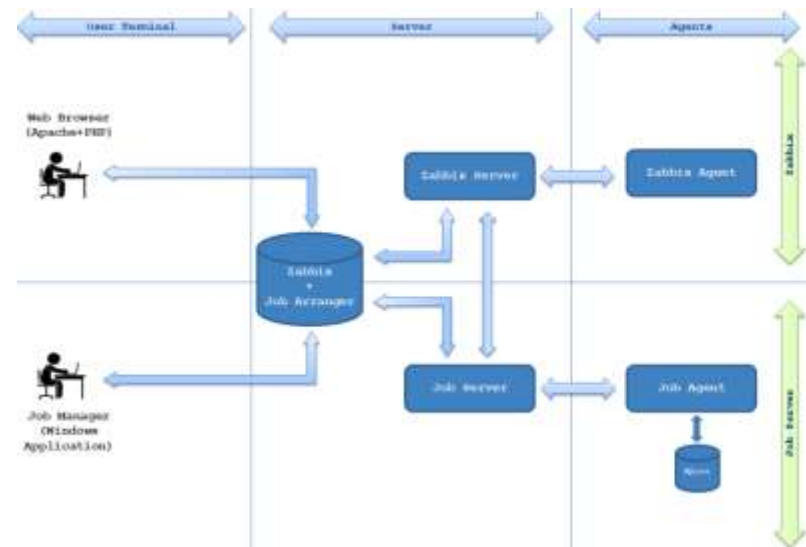


- Time series data is arrays of numbers indexed by time.
- In some fields these time series are called profiles, curves, or traces.



- 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 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 targets

Bare metal

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

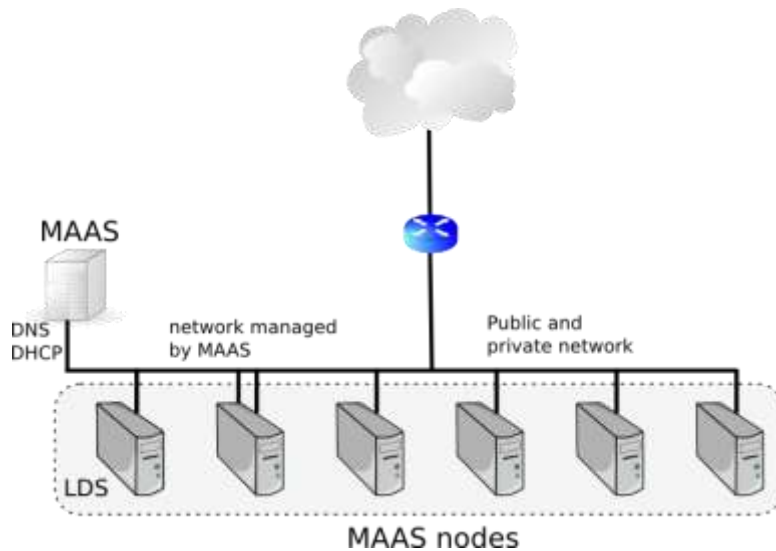
Mobile device

- 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.

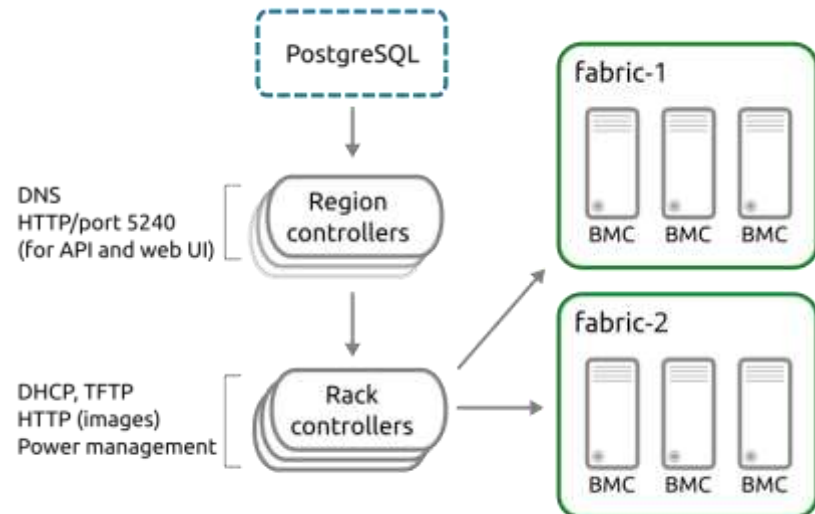


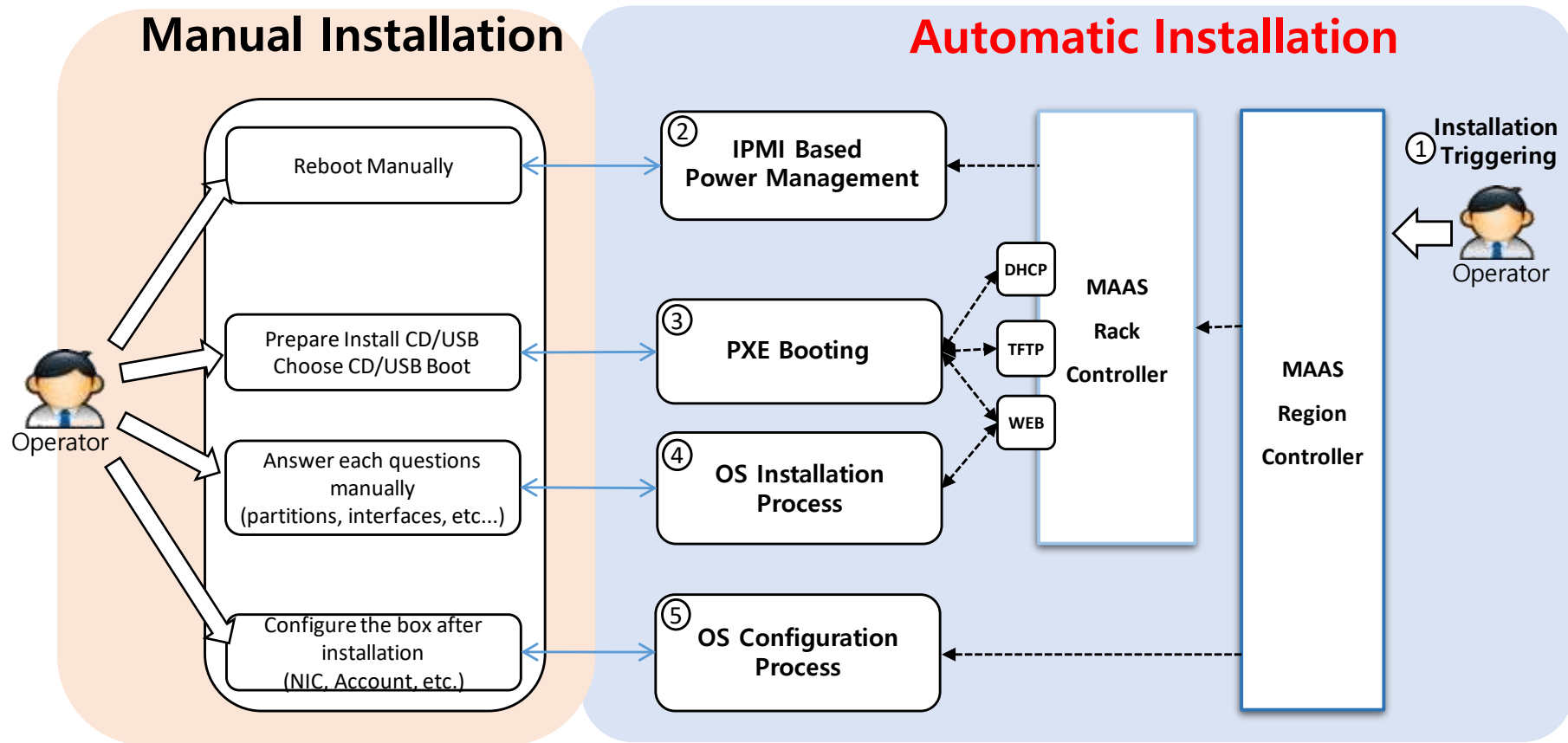


- Bare-metal machines can be quickly provisioned and destroyed; MAAS provides management of a large number of physical machines by creating a single resource pool
- MAAS can act as a standalone PXE services, provides Web GUI, supports various Linux distribution installation, ...

MAAS Controller Architecture

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





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

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





Lab Practice

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

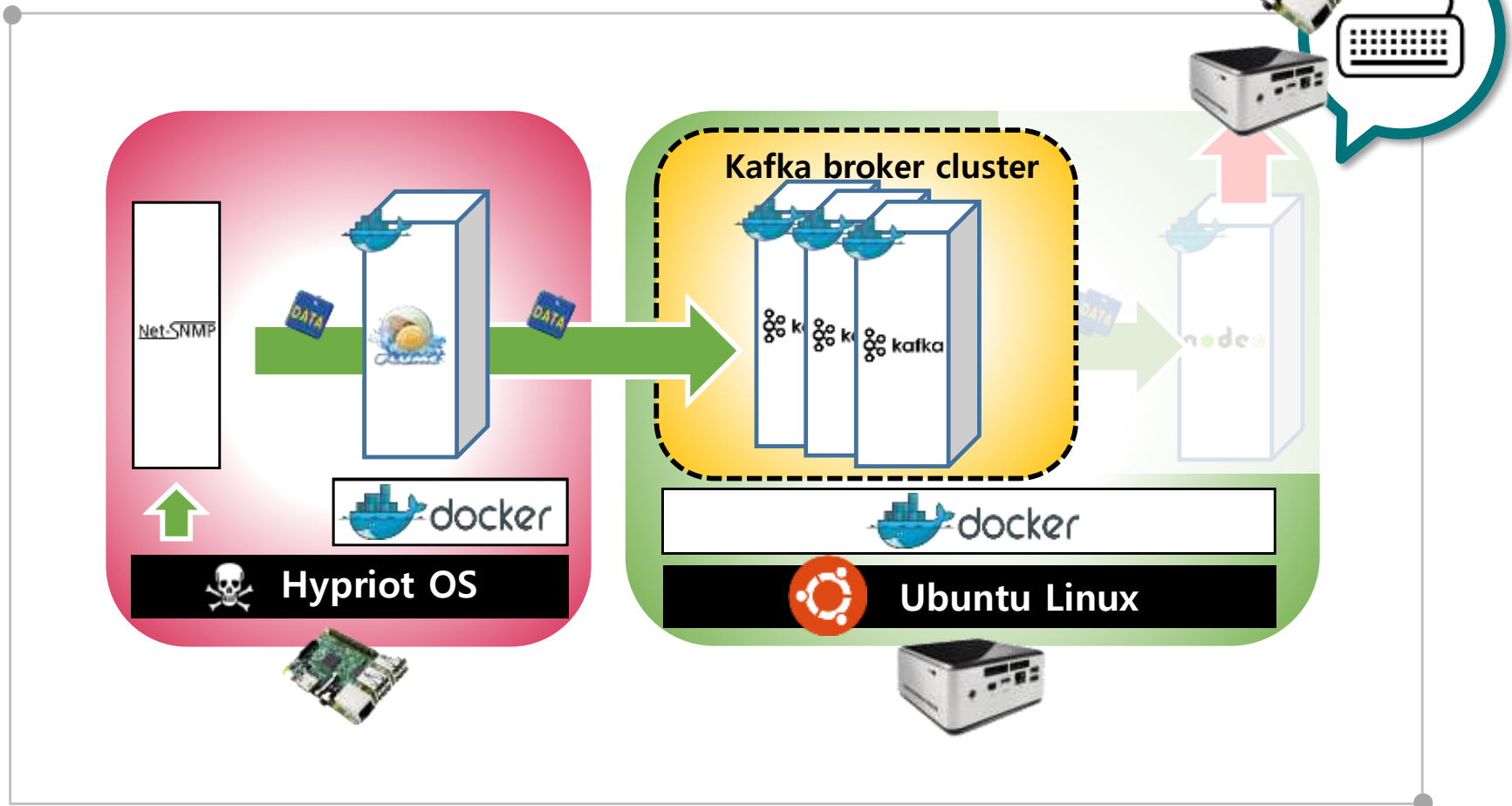


NAME: NT900X3A
CPU: i5-2537U @1.40GHz
CORE: 2
Memory: 4GB DDR3
HDD: 128GB



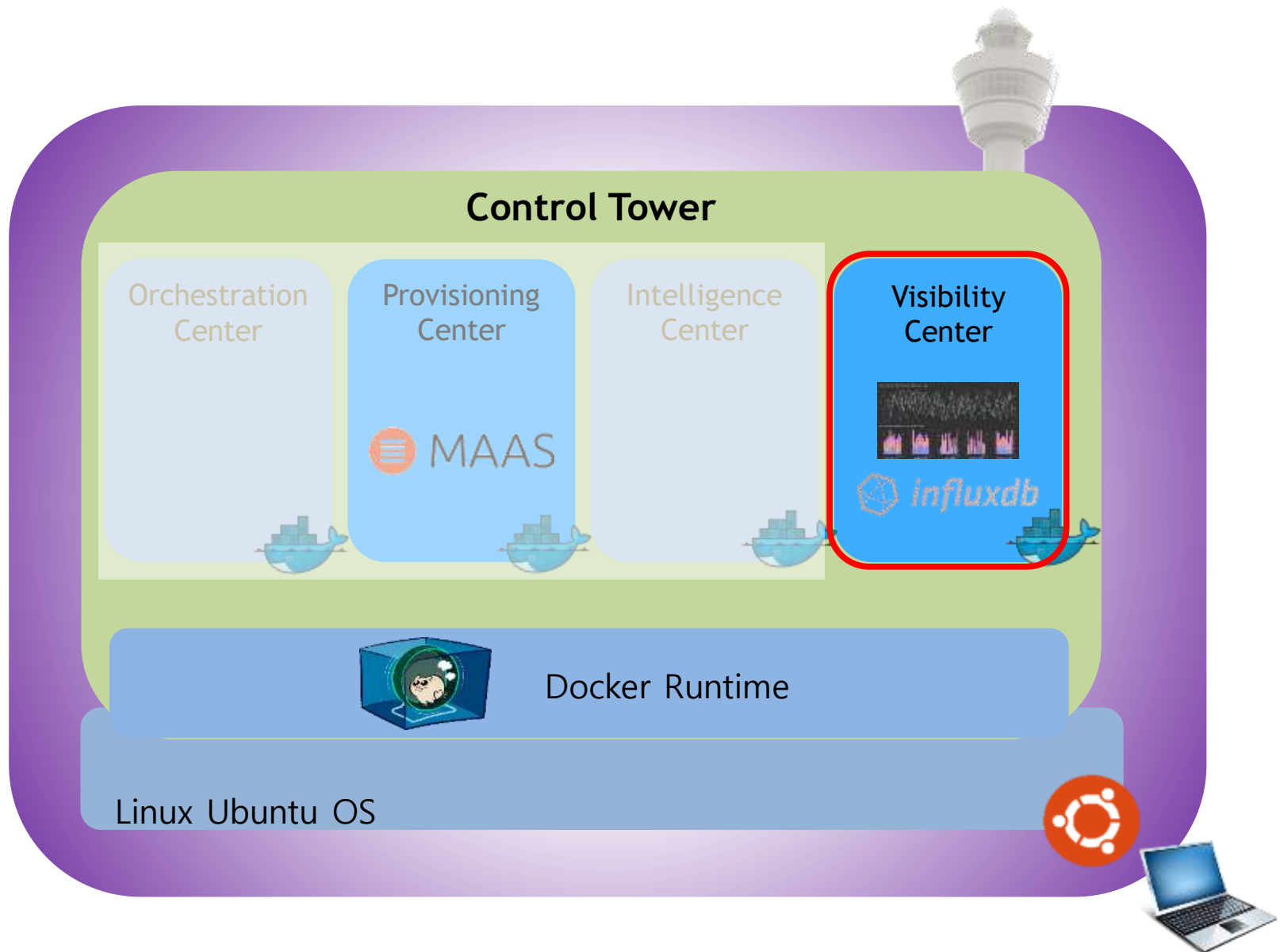
NAME: netgear prosafe 16 port gigabit switch(Switch)
Network Ports: 16 auto-sensing 10/100/1000 Mbps
Ethernet ports

- Verify Inter-Connect Lab's configuration



Are they working?

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



- Run InfluxDB Container

```
$ docker run -d --name=influxdb --net=host influxdb
```

- Make and run Chronograf container

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



- 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
```





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

Modify to your nuc IP Address

Modify to your nuc IP Address

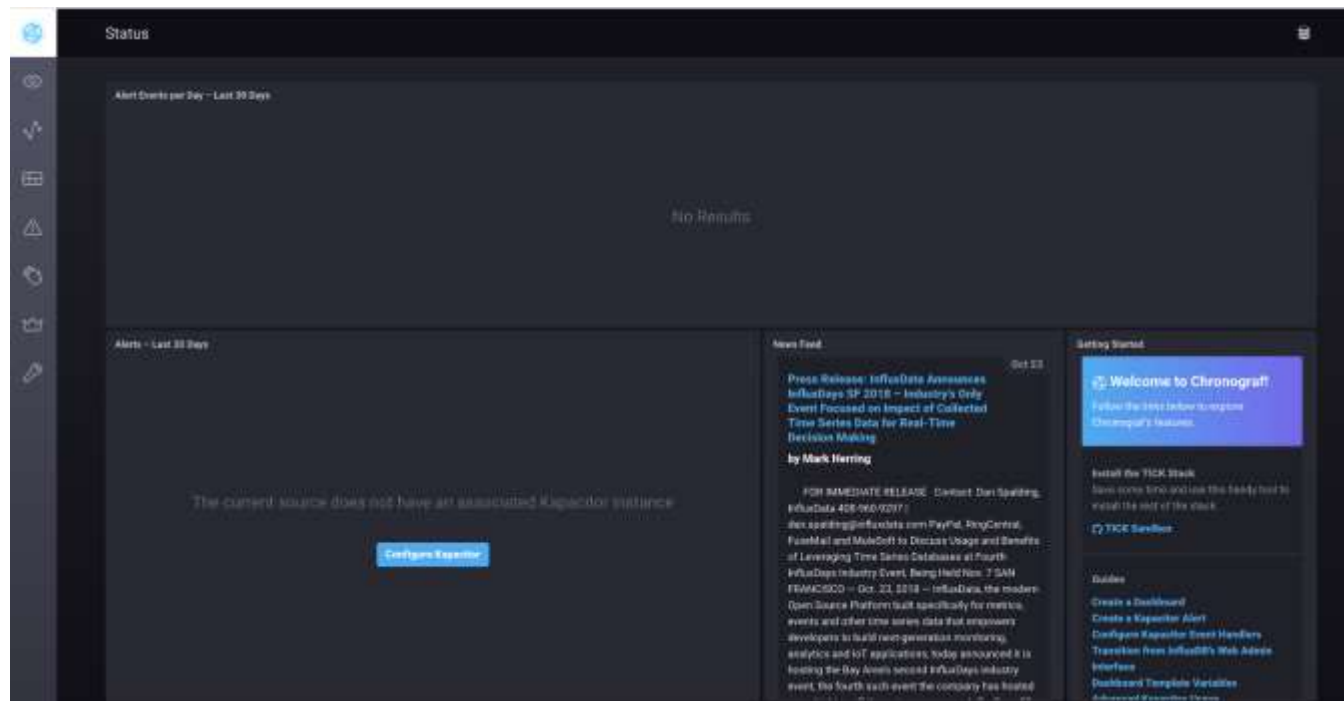
- Run python code

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

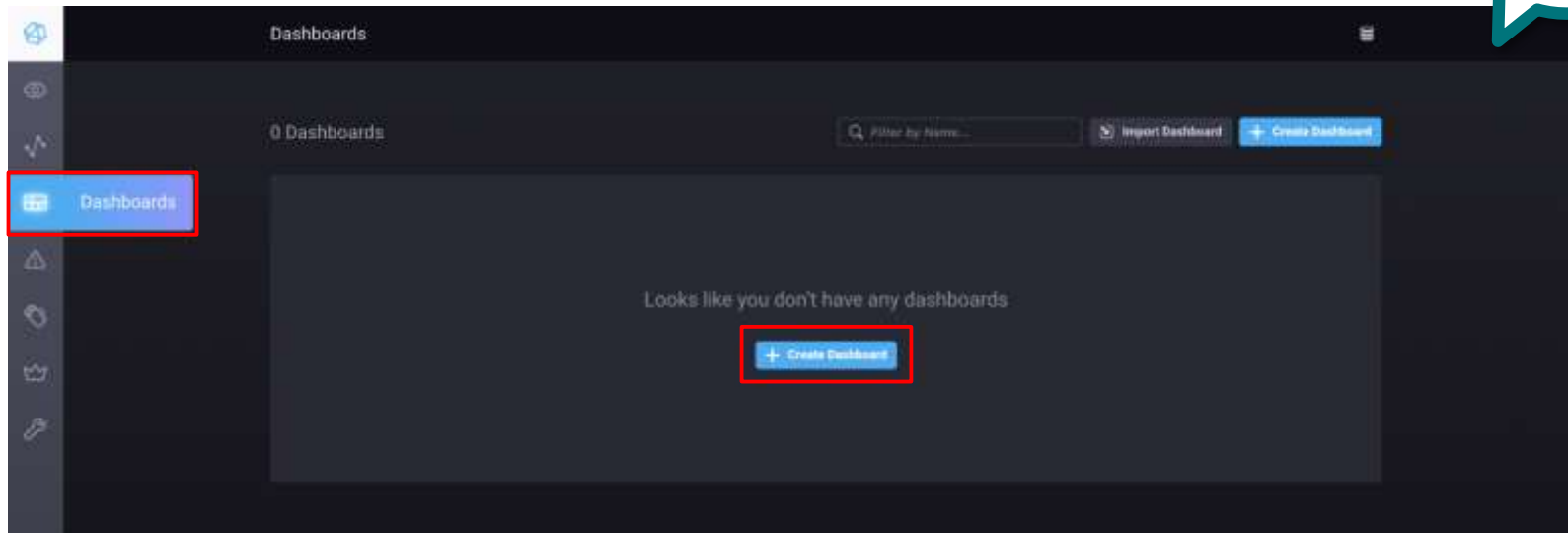
\$ ulimit -S -n 2048

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

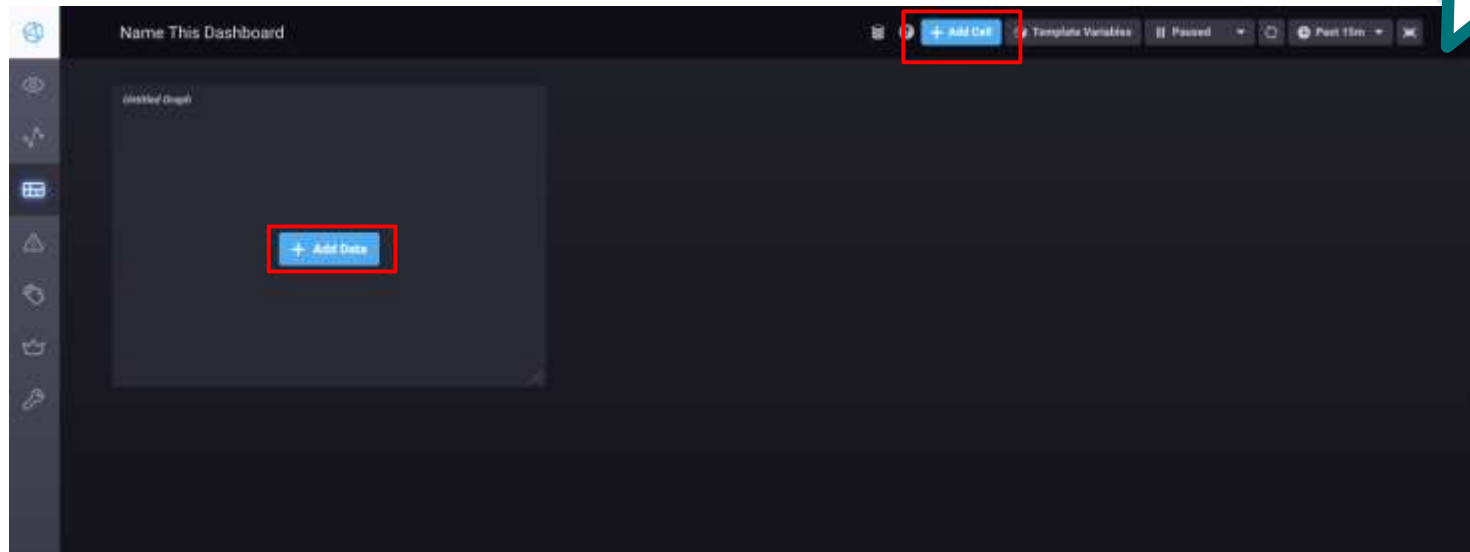
- Open Web browser and connect to Chronograf Dashboard
<http://<NUC IP>:8888>



- Create Dash board



- Add Cell & Data





•Add Data

Queries Visualization

SELECT "Memory" FROM "Labs"."autogen"."la..."

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

✓ Success!

DB Retention Policy

Interval: monitor

Labs.autogen

Measurements & Tags

Filter

Tags

- host = 1
- region = 1

Fields

CPU_load

CPU_usage

Memory

Select functions below

mean	median	count	min
max	sum	first	last
spread	stdev		

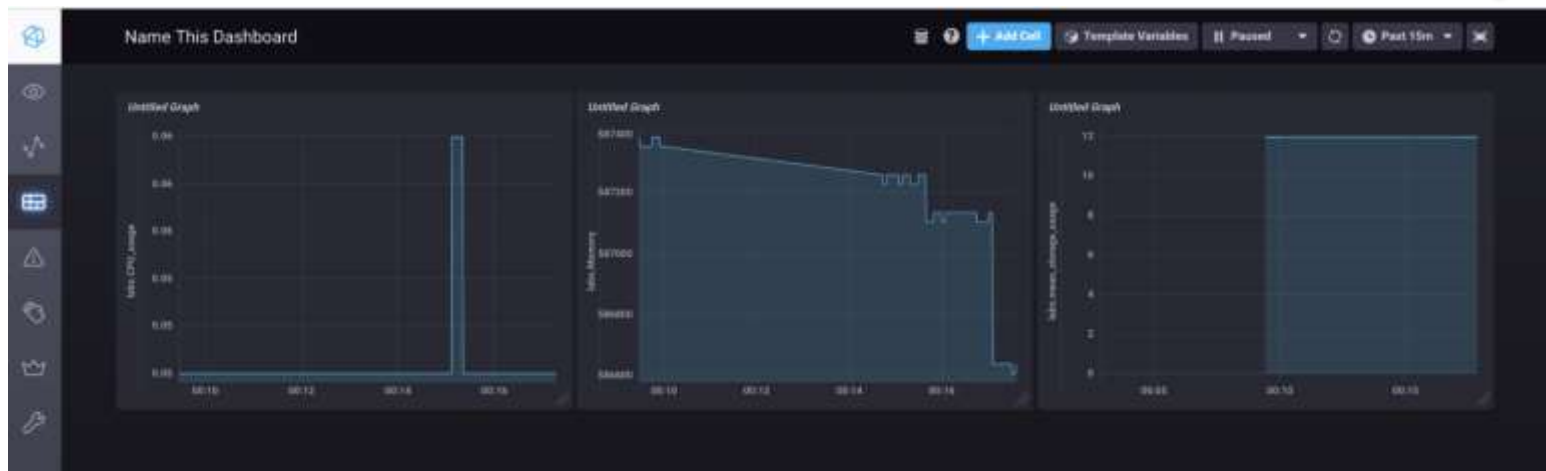
storage_usage

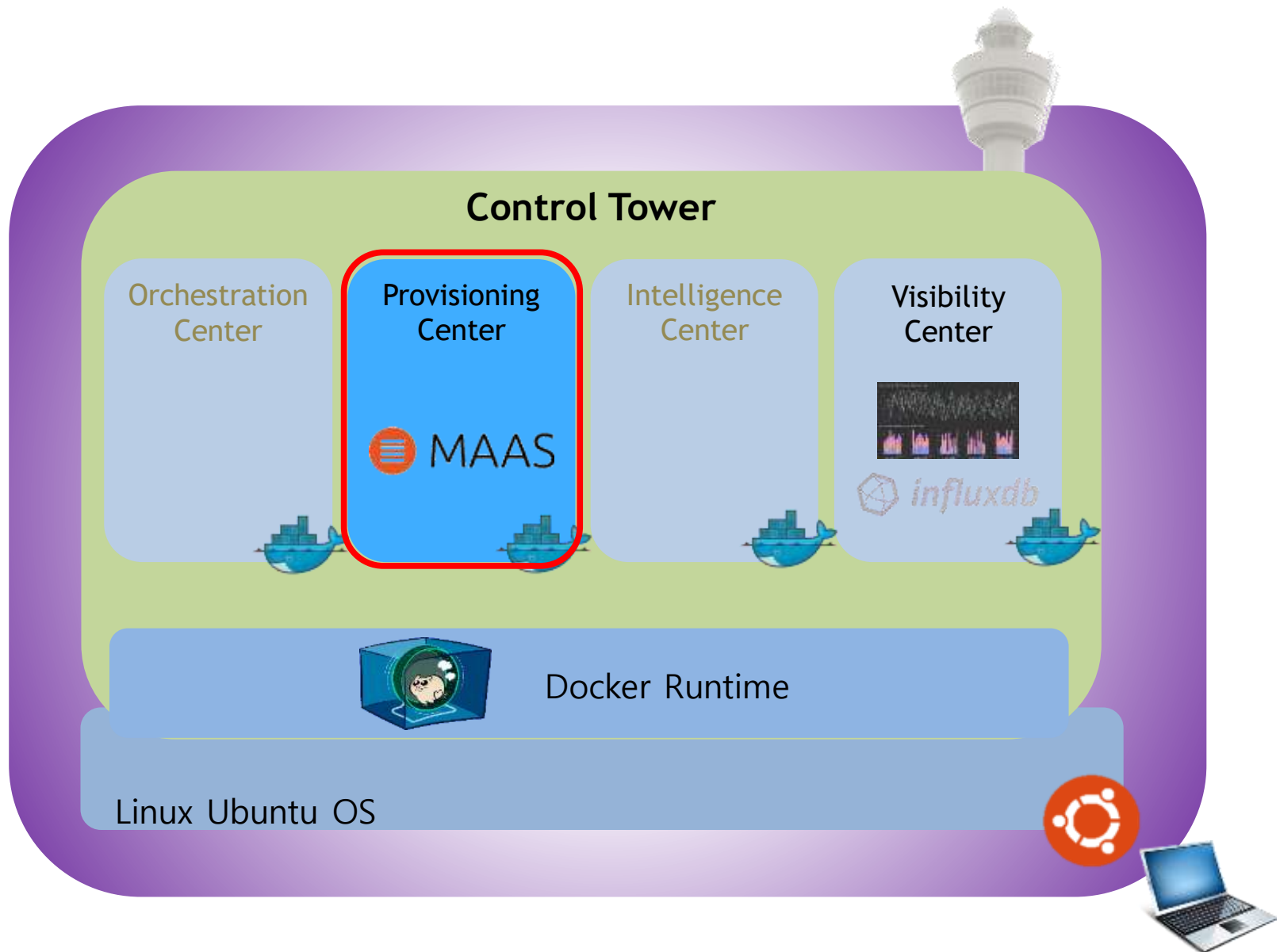
timestamp

Success

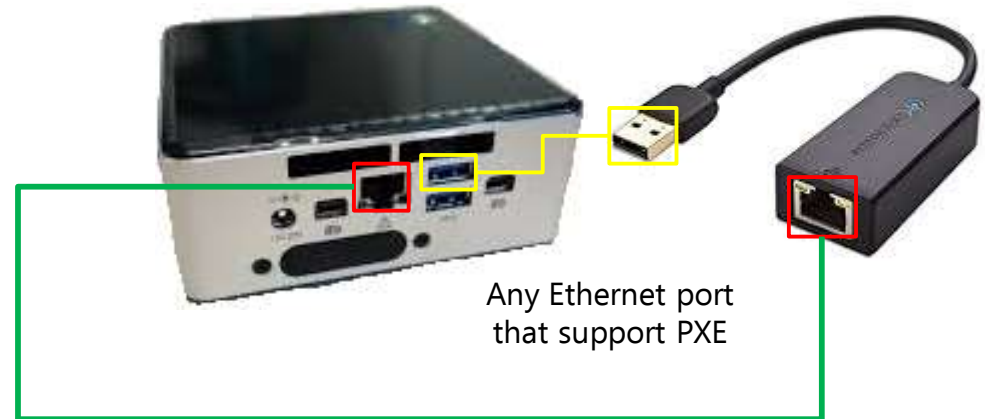
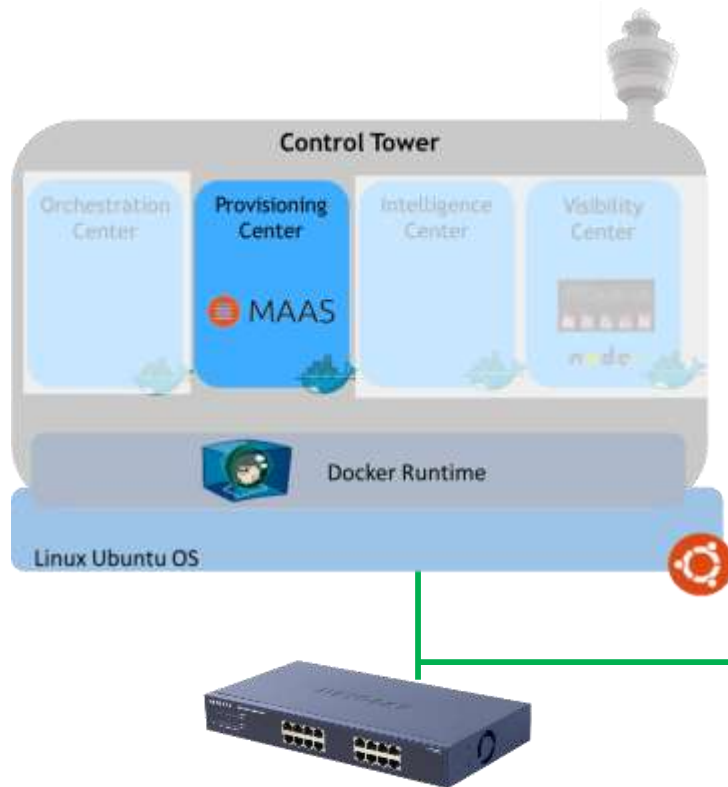


- We can see the changes of values from database





Management & Control



Requirements for manual Installation

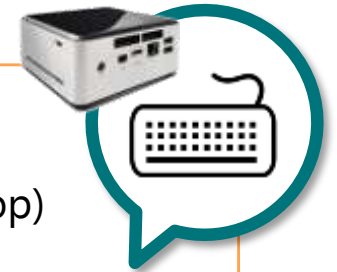
- DHCP PXE bootable
- USB to ethernet connector

Note: Typical NUC does not have a IPMI port.



- 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"

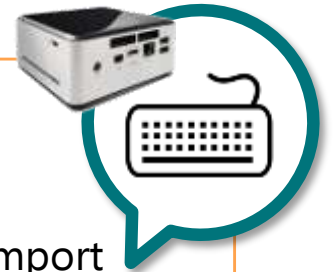
*region : Organizational unit. Contains all information about all machines running



- 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"



- Create SSH key

```
$ sudo ssh-keygen
```

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

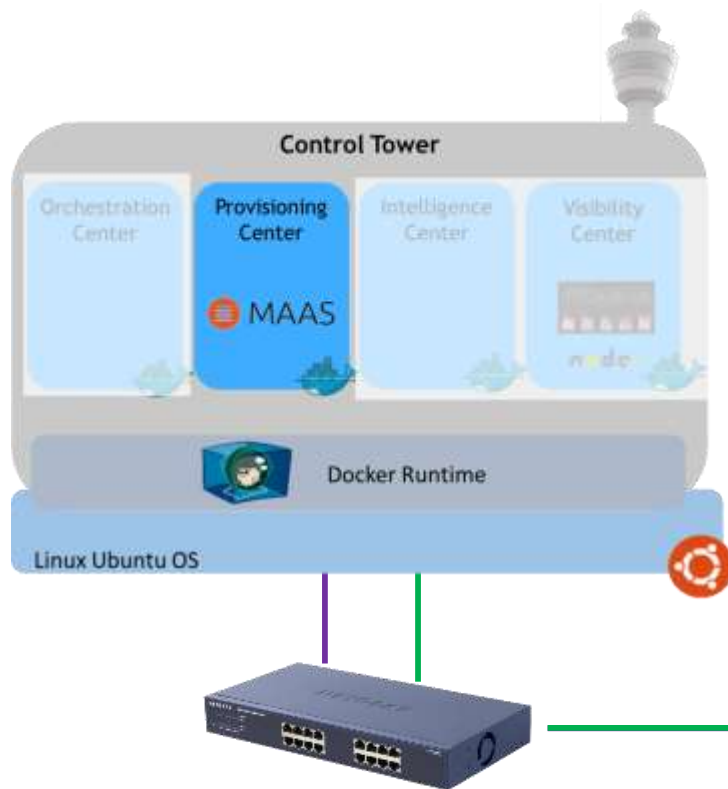
Copy the outcome , press "upload" and paste it on "User ID" and import

- 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**

— Power
— Management & Control



IPMI port

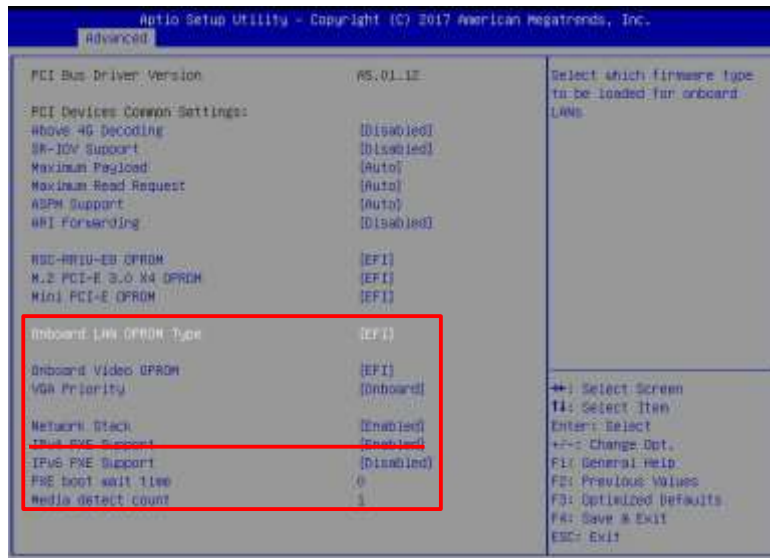


Any Ethernet port
that support PXE

Requirements for Automated Installation

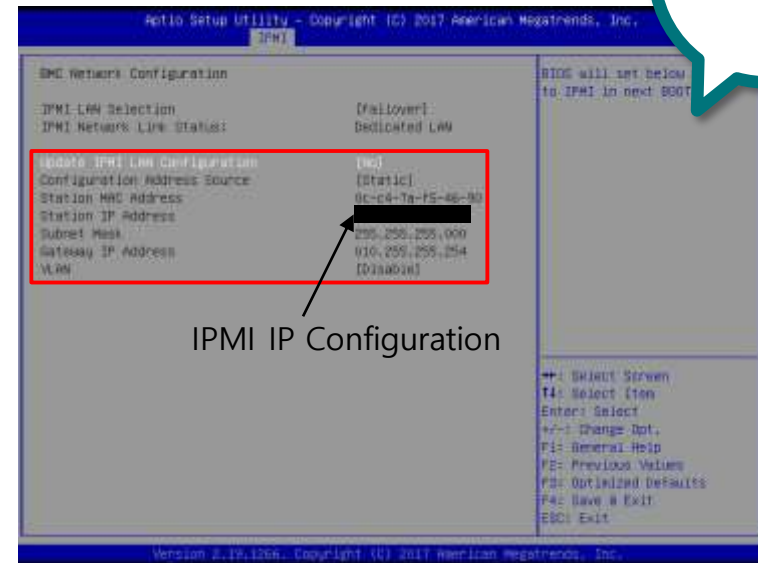
- IPMI, Intel AMT, IBM HMC ... and so on.
- DHCP PXE bootable

Note: Typical NUC does not satisfy the above requirement!



<BIOS PXE Configuration>

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



<BIOS IPMI Configuration>

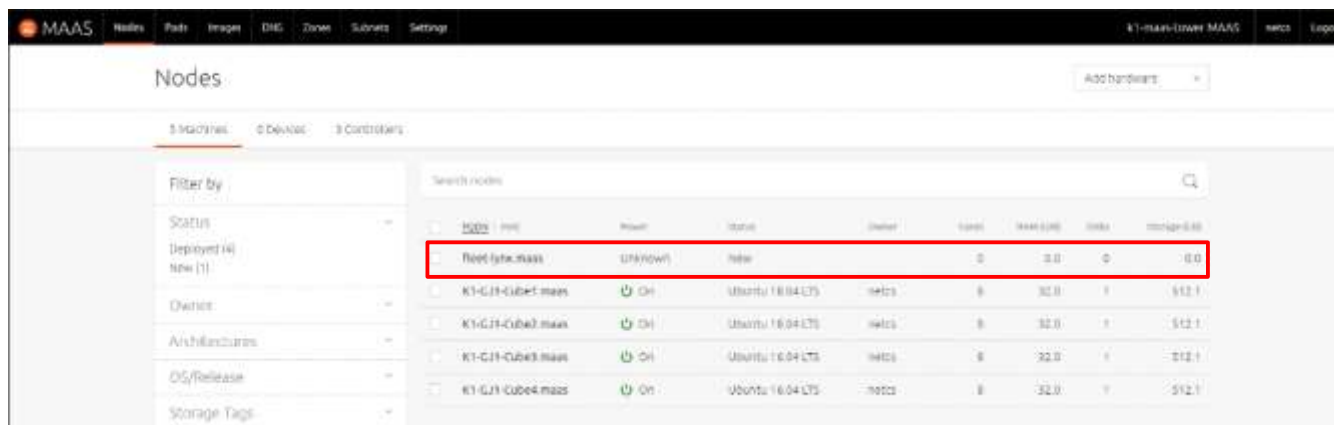
Booting under MAAS direction...

Check this
message

```
OK ] Reached target Local Encrypted Volumes.
OK ] Started Confit a transient machine-id on disk.
OK ] Started etables 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.
S3.126408] cloud-init[570]: Cloud-init v. 18.3-9-g2e62cb8a-ubuntu18.04.2 running 'init-local'
at Tue, 16 Oct 2018 12:20:59 +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>

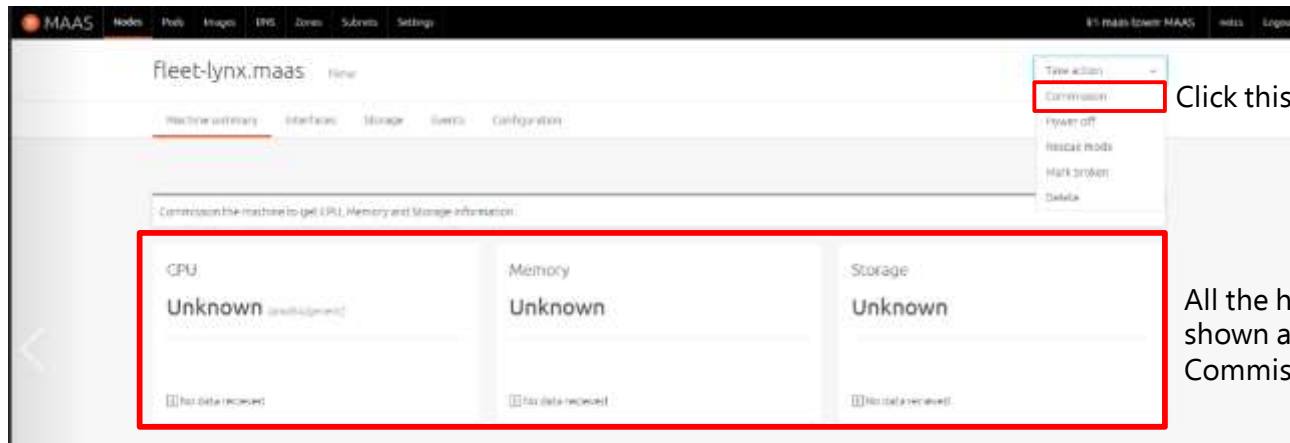
<Enlistment after PXE booting>



The screenshot shows the MAAS (Metal as a Service) web interface. The 'Nodes' tab is selected, displaying a list of nodes. The first node, 'Node (yhc.maas)', is highlighted with a red box. The table columns include Name, Power, Status, Owner, Capacity, Memory, Disk, and Storage. The 'Node (yhc.maas)' row shows a status of 'PENDING' and a capacity of 0.0.

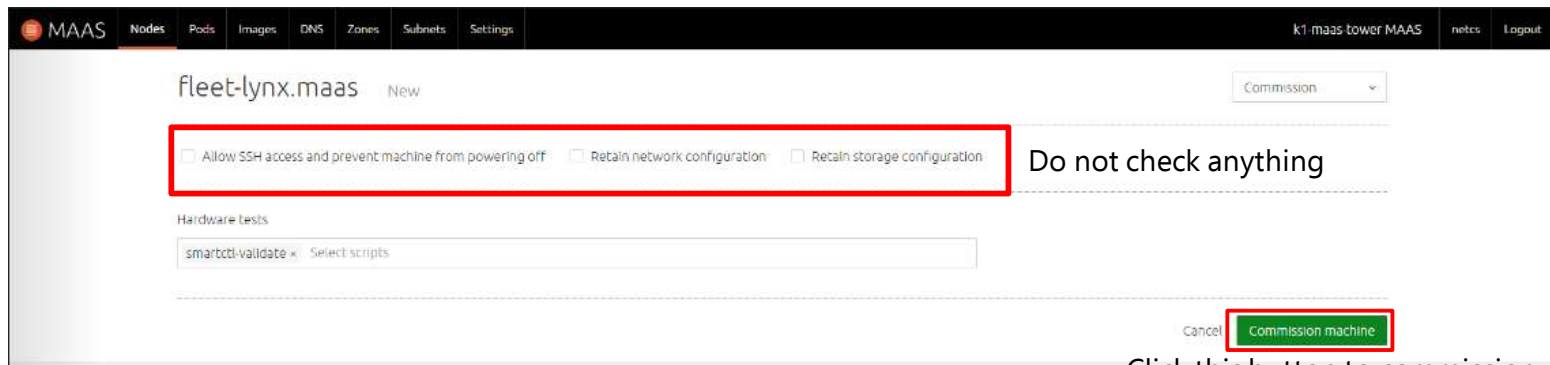
Name	Power	Status	Owner	Capacity	Memory	Disk	Storage
Node (yhc.maas)	UNKNOWN	PENDING		0.0	0.0	0	0.0
K1-GJ1-Cube2.maas	ON	OK	Ubuntu 16.04 LTS	8	32.0	1	512.1
K1-GJ1-Cube3.maas	ON	OK	Ubuntu 16.04 LTS	8	32.0	1	512.1
K1-GJ1-Cube4.maas	ON	OK	Ubuntu 16.04 LTS	8	32.0	1	512.1

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



Click this button to commission

All the hardware information shown as 'Unknown' before Commissioning



Do not check anything

Click this button to commission

- It takes about 10 minutes



MAAS Nodes Pods Images DNS Zones Subnets Settings k1-maas-tower MAAS neto

fleet-lynx.maas Ready Power off check now

Machine summary Interfaces Storage Commissioning Configuration

CPU 12 cores (amd64/generic) Memory 32.0 GiB Storage 512.1 GB over 1 disk

No tests have been run. No tests have been run.

1x512GB (ssd) 1 test has passed. See storage

Take action Commission Acquire Deploy Test hardware Rescue mode Mark broken Delete

Click this button to Deploy

You can see all the Hardware information After commissioning

MAAS Nodes Pods Images DNS Zones Subnets Settings k1-maas-tower MAAS neto Logout

fleet-lynx.maas Ready Power off check now

Choose your image Ubuntu Ubuntu 18.04 LTS "Bionic Beaver" Default kernel

Cancel Deploy machine

Click this button to Deploy

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



```
Martha: 10.04.1 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)>



Lab Review

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

01

Visibility Center function to **enable 'distributed monitoring'** over remote Boxes and to **store 'monitoring information'** to time-size DB.

02

Provisioning Center function to **enable remote 'installation & configuration'** (of OS and others) of distributed Boxes.

Thank You for Your Attention

Any Question?

Mini@smartx.kr

