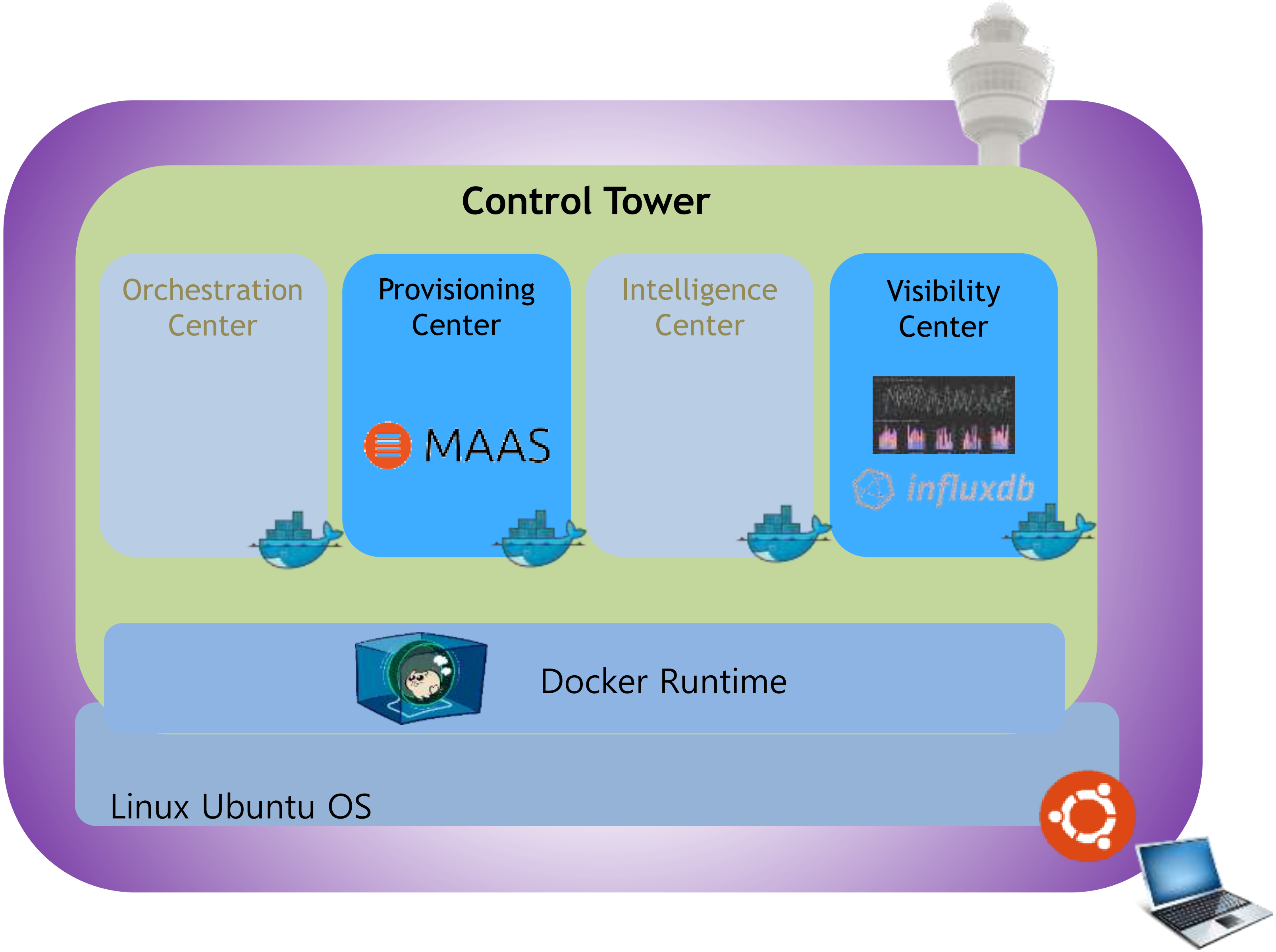
Computer Systems for AI-inspired Cloud Theory & Lab.

SmartX Labs – Mini (MOOC Selection)

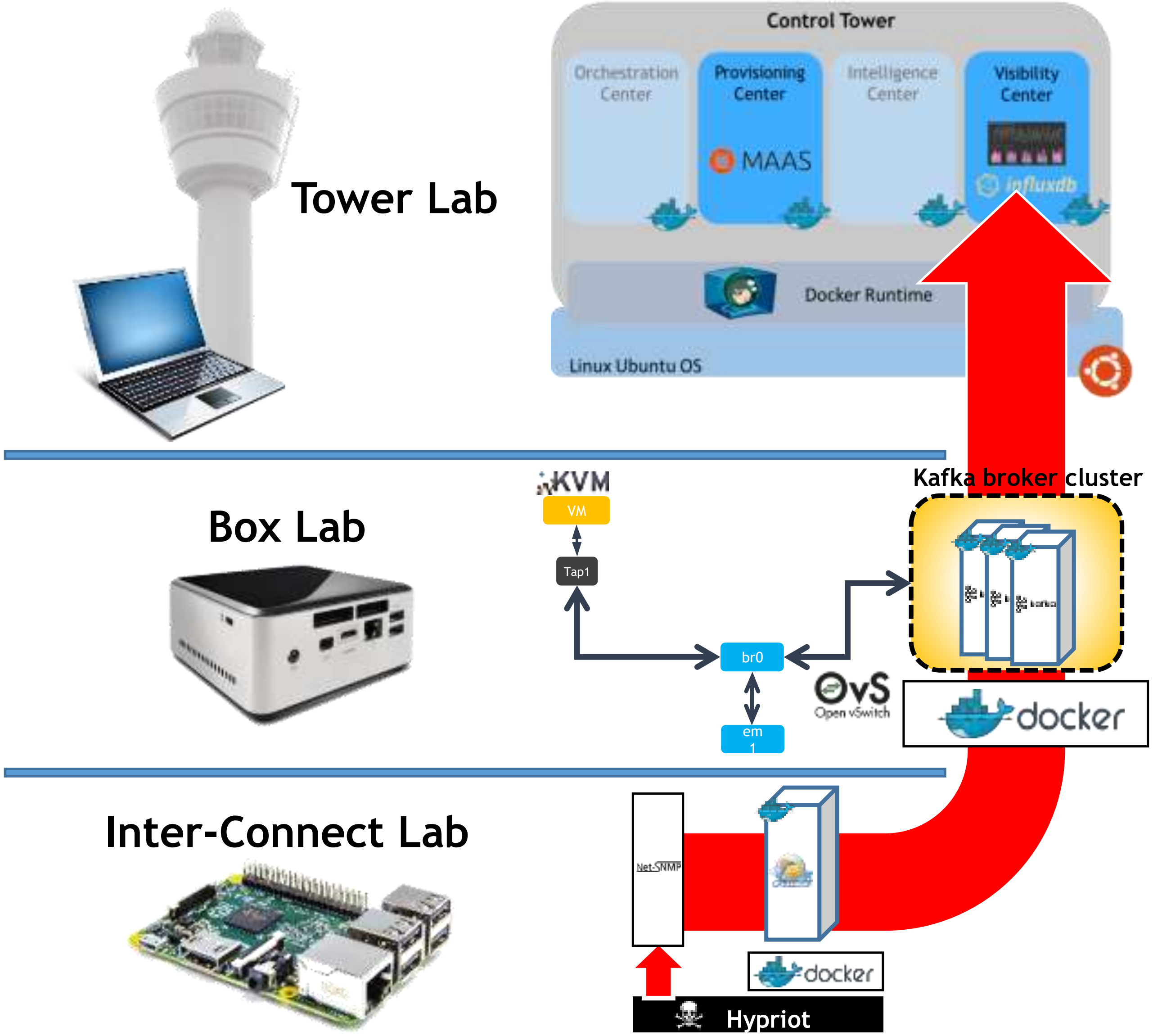
Tower (Lab#3)

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

Tower Lab: Concept

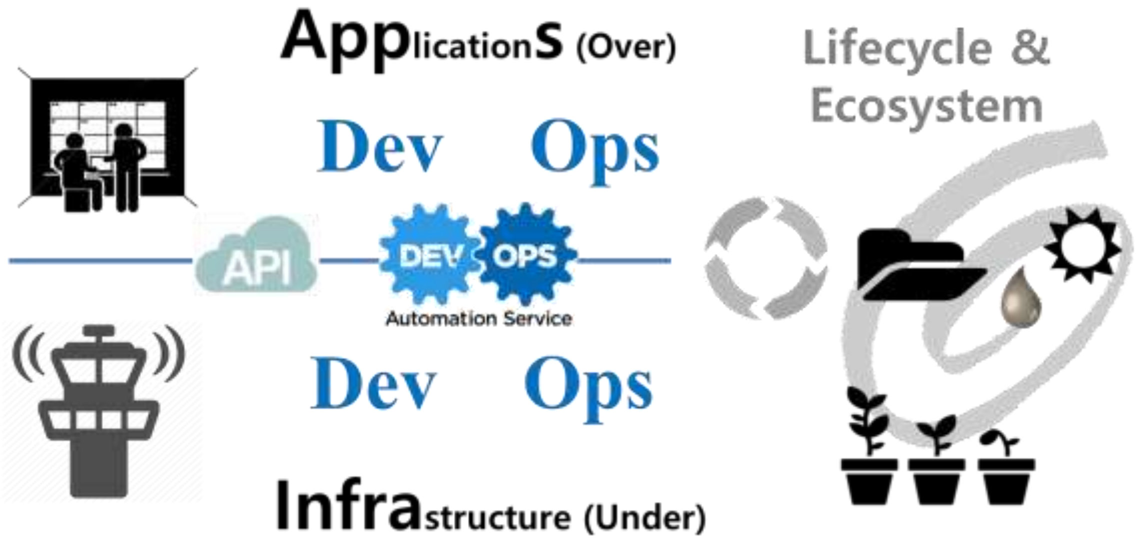


SmartX Labs #1~#3: Relationship



Lab Theory

SmartX Automation Framework



ApplicationS (Over)

Dev Ops

API

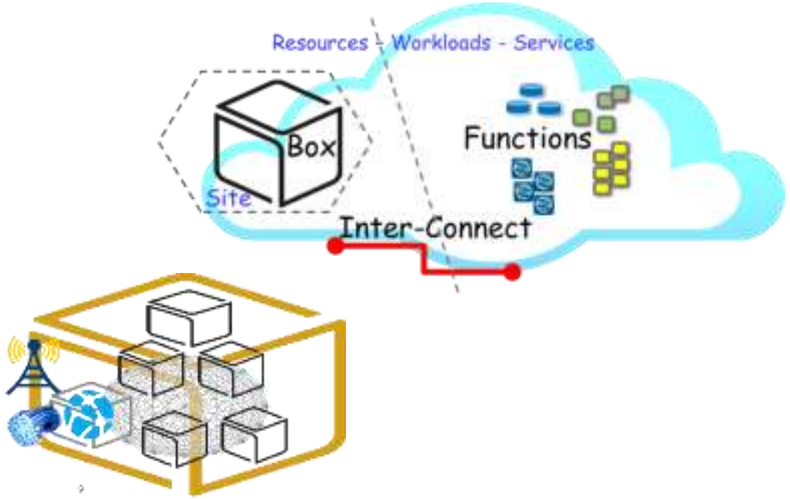
DEV OPS

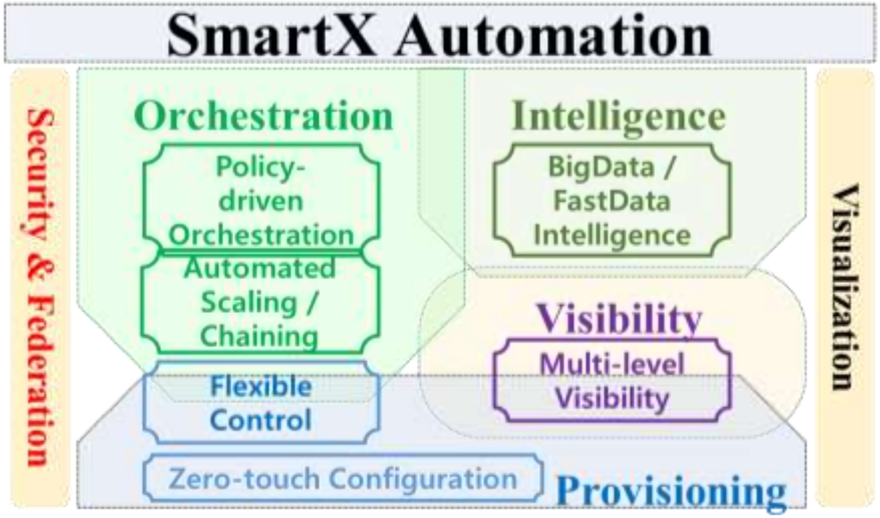
Automation Service

Dev Ops

Infrastructure (Under)

Lifecycle & Ecosystem





SmartX Automation

Security & Federation

Orchestration

Policy-driven Orchestration

Automated Scaling / Chaining

Flexible Control

Intelligence

BigData / FastData Intelligence

Visibility

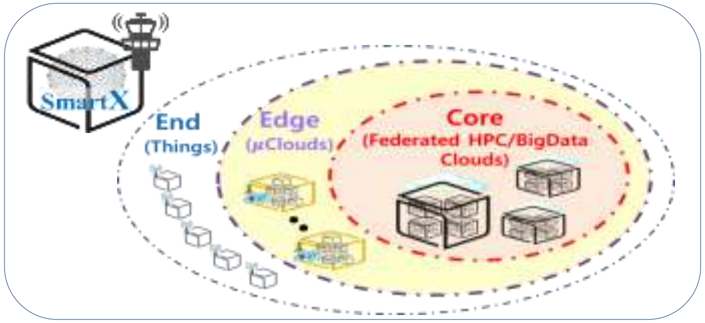
Multi-level Visibility

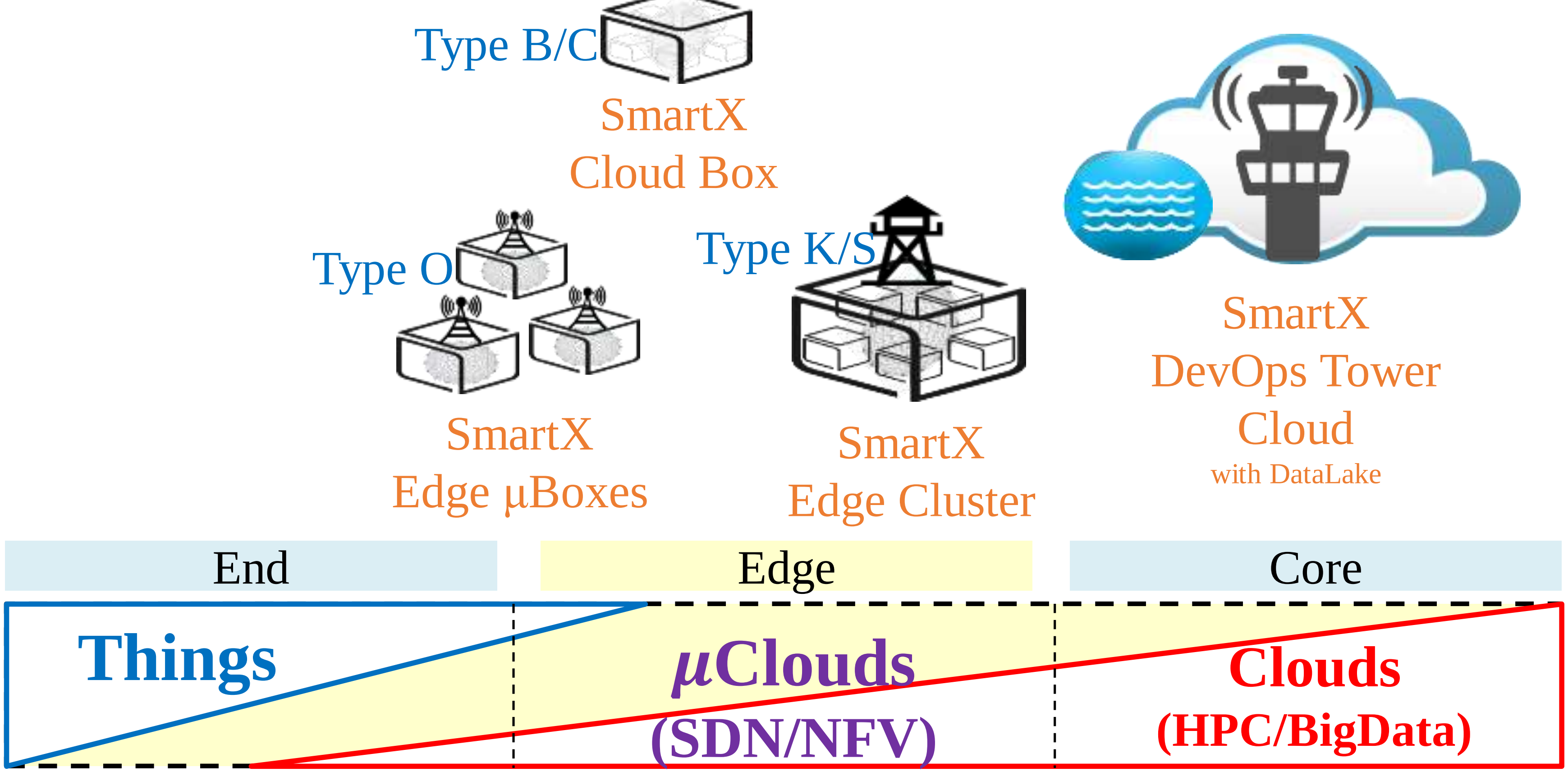
Zero-touch Configuration

Provisioning

Visualization

SmartX Composable Playground & Boxes





Type B/C

SmartX
Cloud Box

Type O

Type K/S

SmartX
DevOps Tower
Cloud
with DataLake

SmartX
Edge μBoxes

SmartX
Edge Cluster

End

Edge

Core

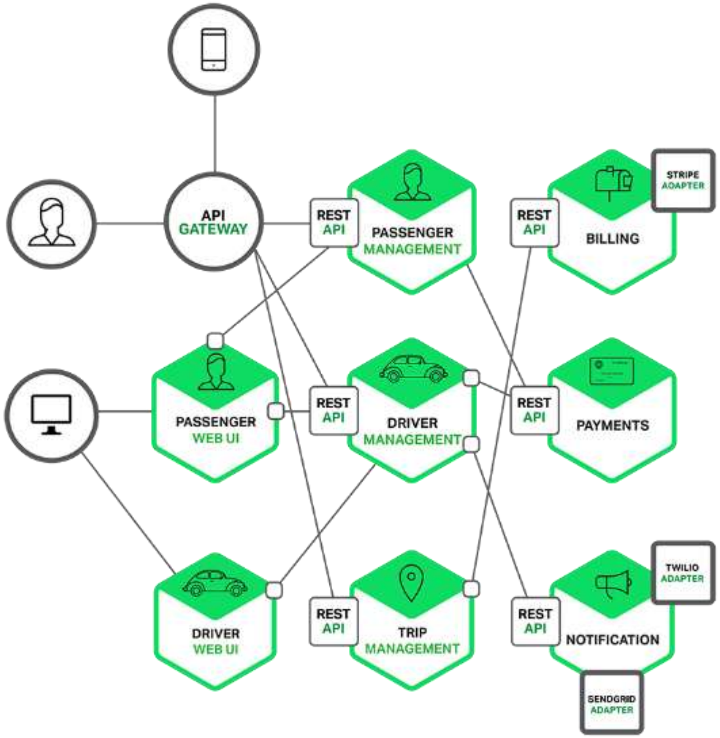
Things

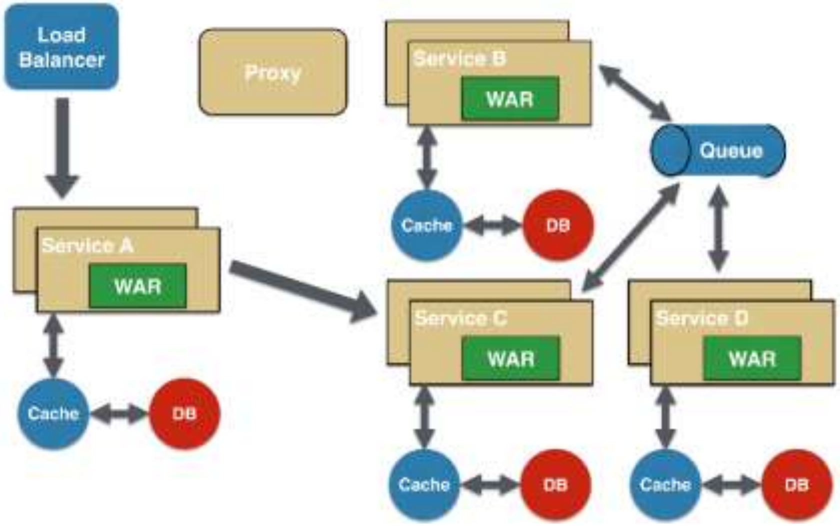
μClouds
(SDN/NFV)

Clouds
(HPC/BigData)

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

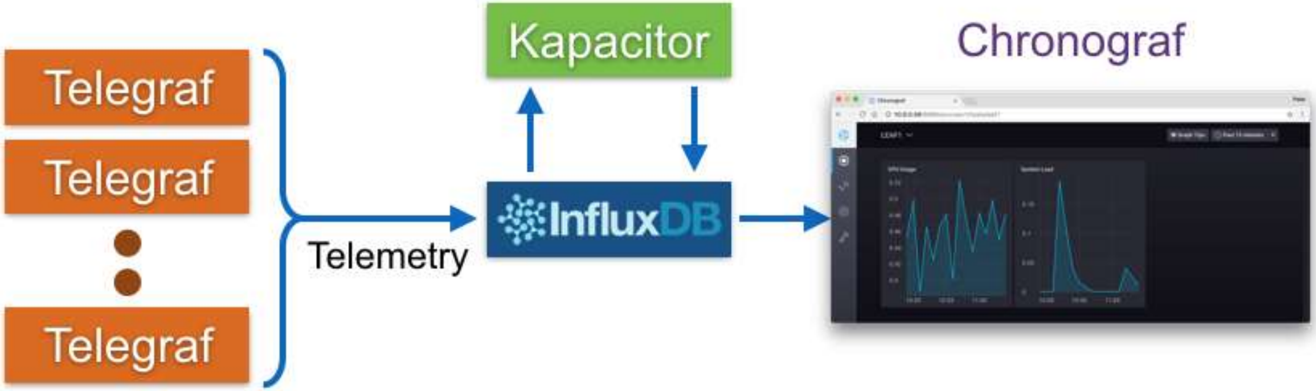




Visibility: TSDB (Time Series Database)

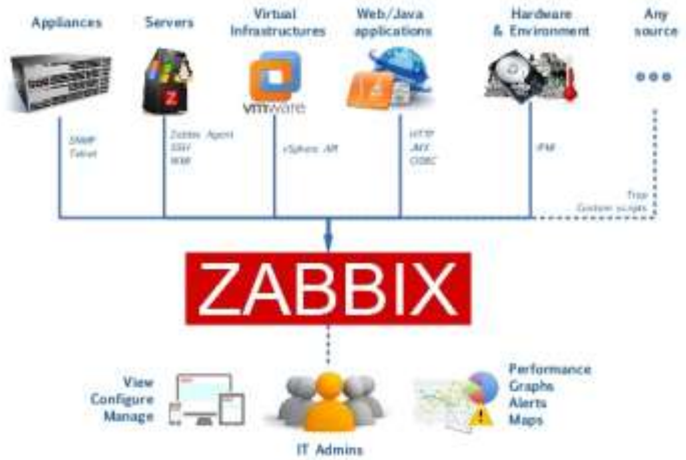
influxdb
cassandra
druid
riakTS
InfluxDB
elasticsearch

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

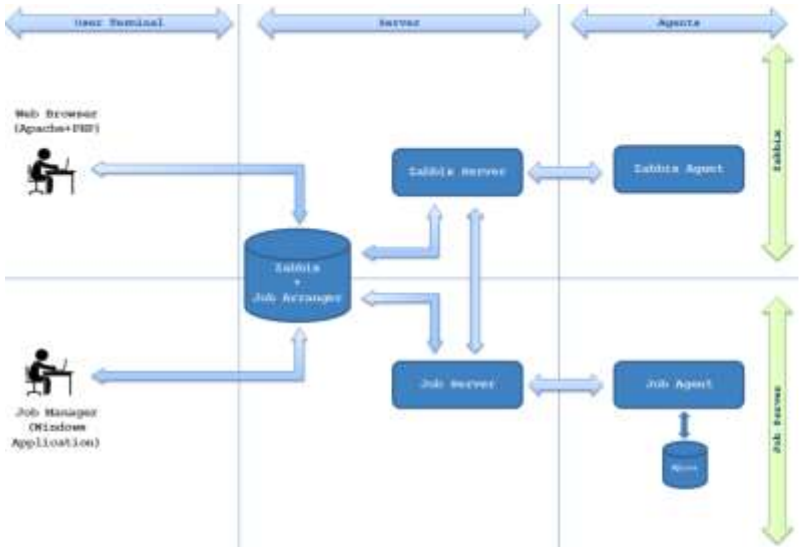


Visibility: Zabbix Distributed Monitoring

- Adopts a flexible notification mechanism
-
- User can configure & watch graph easily via Web GUI



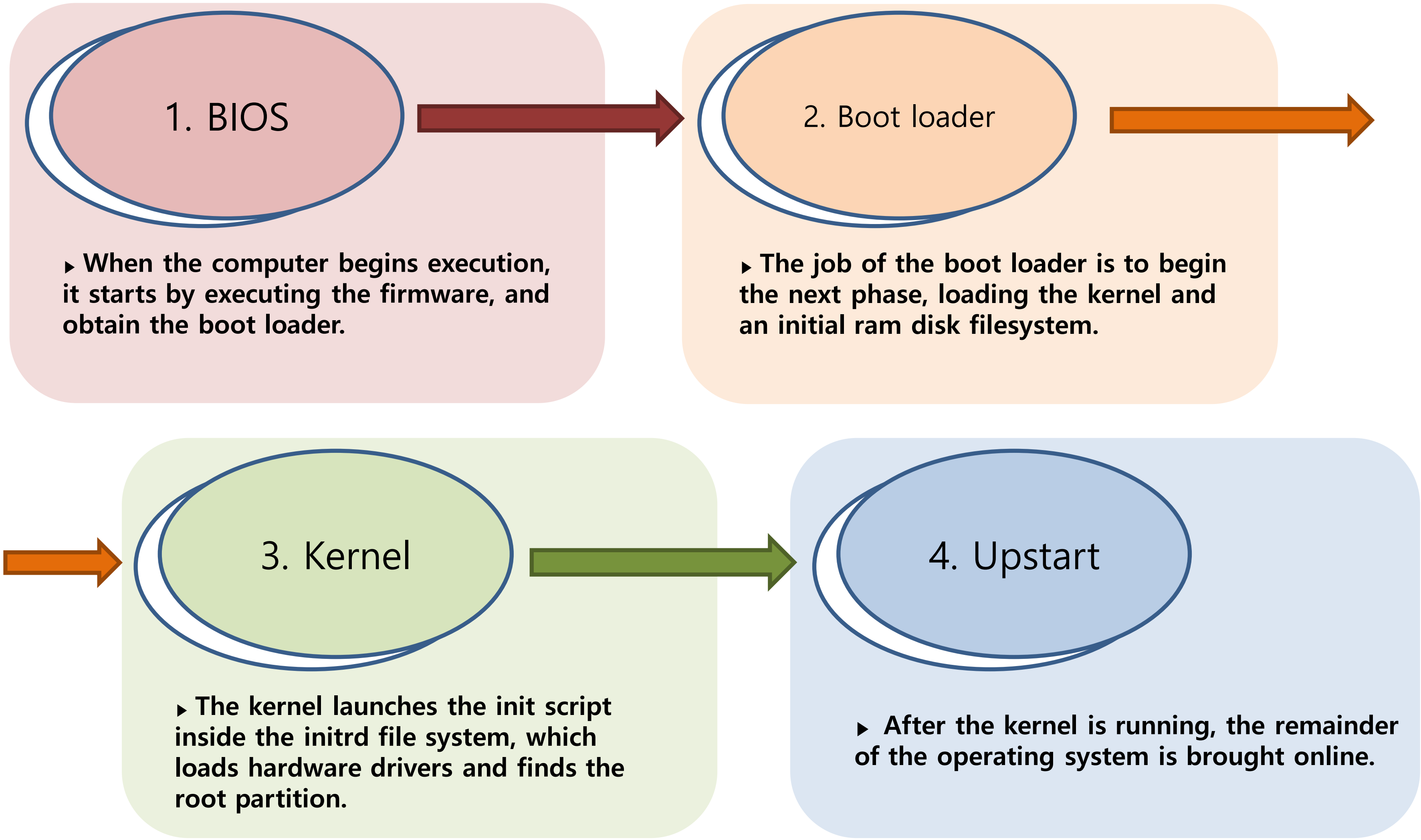
- Consists of structured server and client
  - Client collects the monitoring data and send it to the Zabbix server
  - Server visualizes the data that is collected by the Zabbix Agent



Zabbix 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

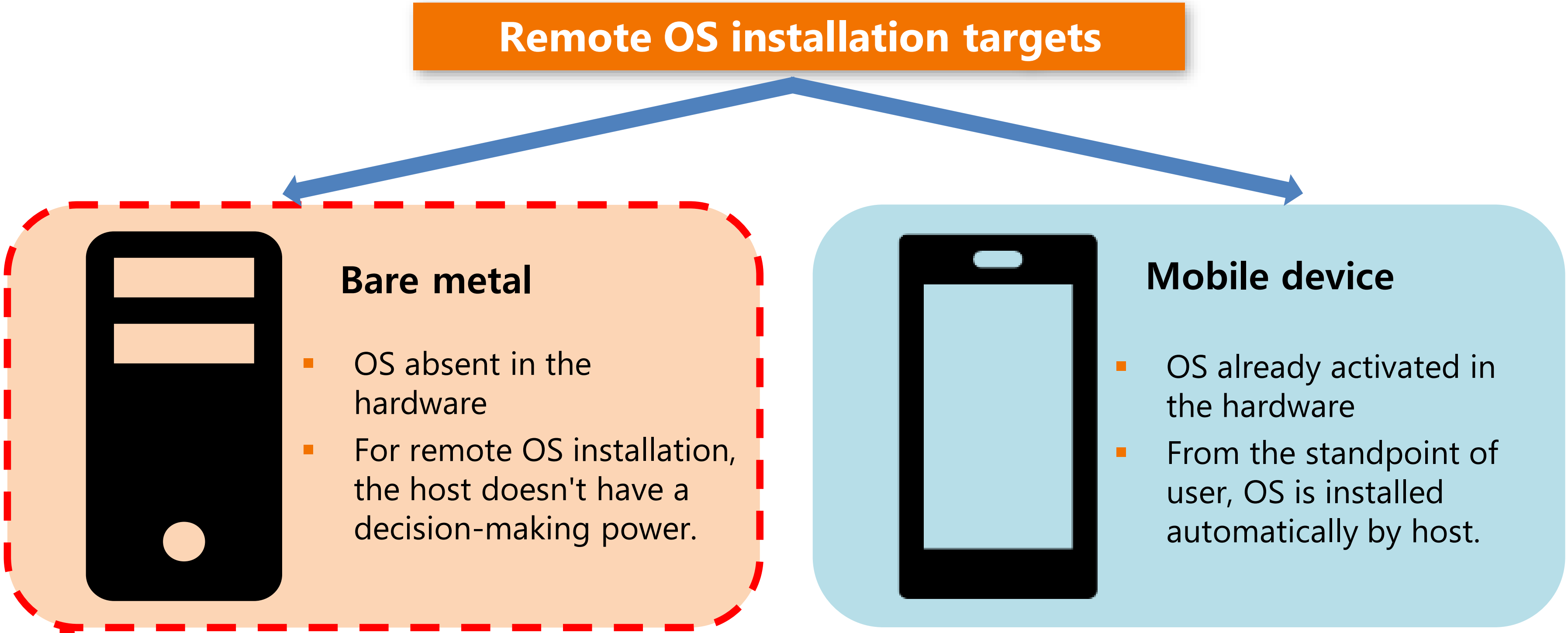
- The kernel launches the init script inside the initrd file system, which loads hardware drivers and finds the root partition.

4. Upstart

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

Reference: https://wiki.ubuntu.com/Booting

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

MAAS(<u>Metal</u> As A Service) is better suited to the <u>bare metal</u>.

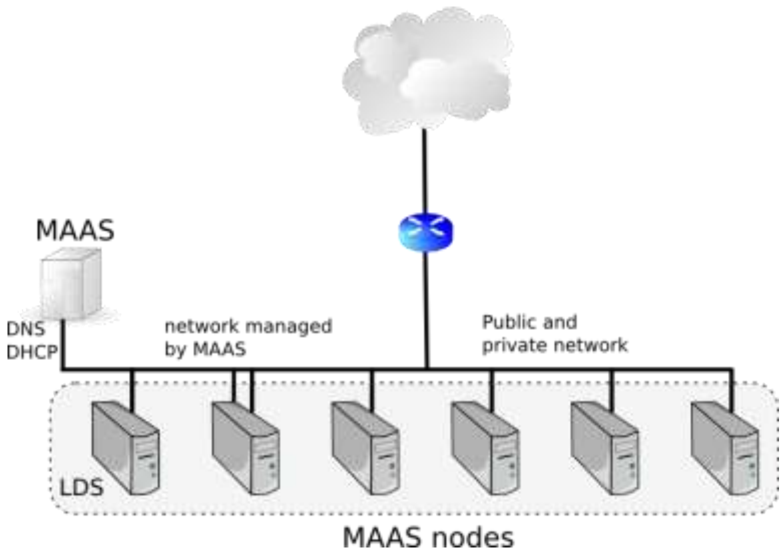
puppet

cobbler

stacki

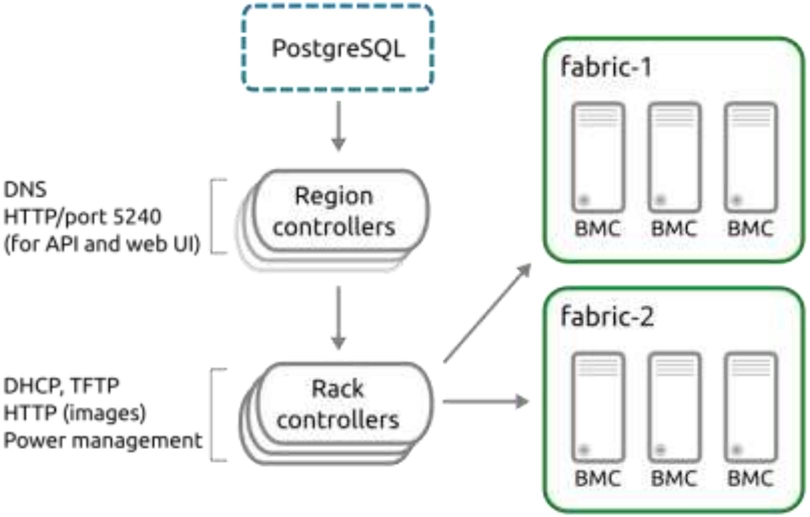
Provisioning: Ubuntu MAAS (Metal as a Service)

- 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, …

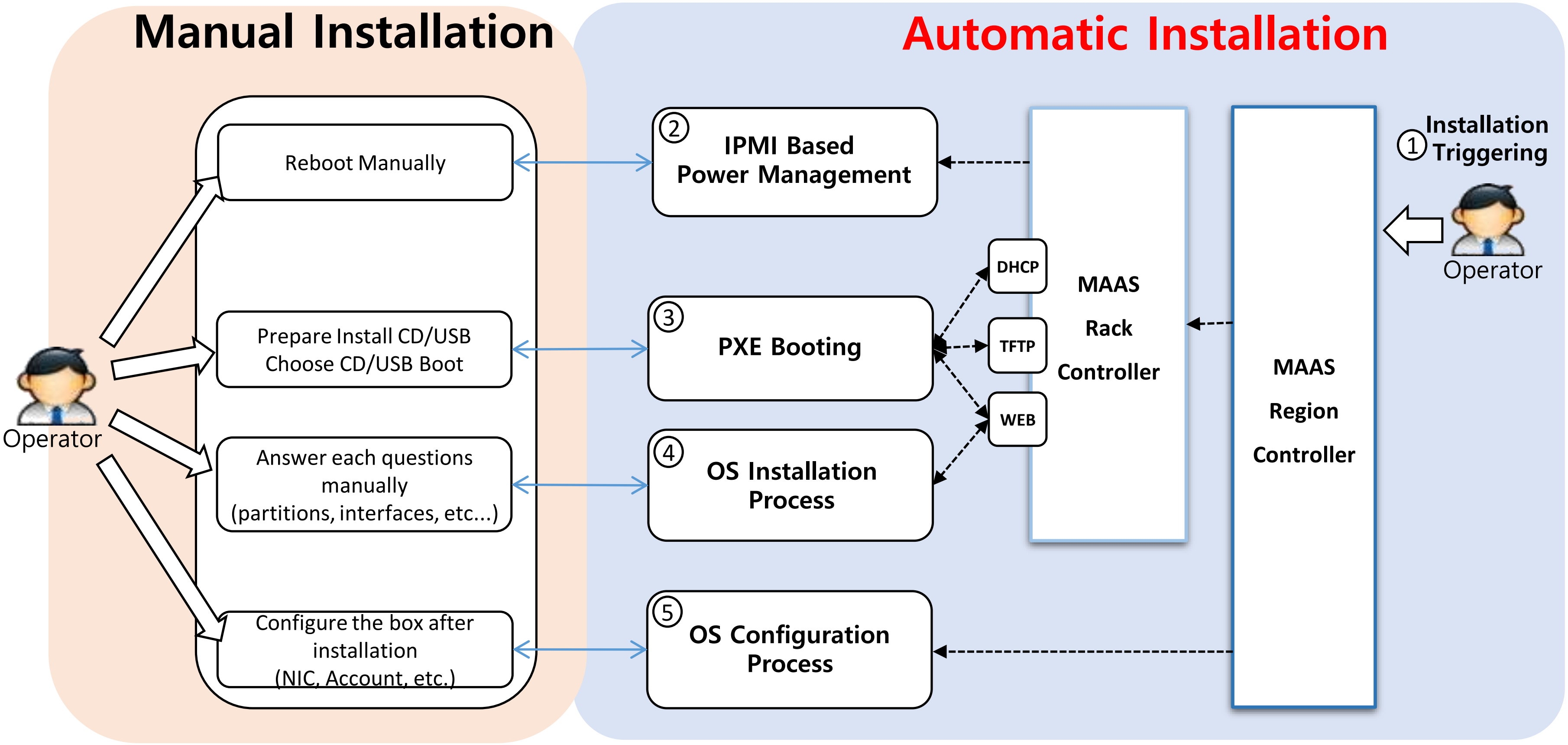


MAAS Controller Architecture

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



MAAS: OS Automated Installation



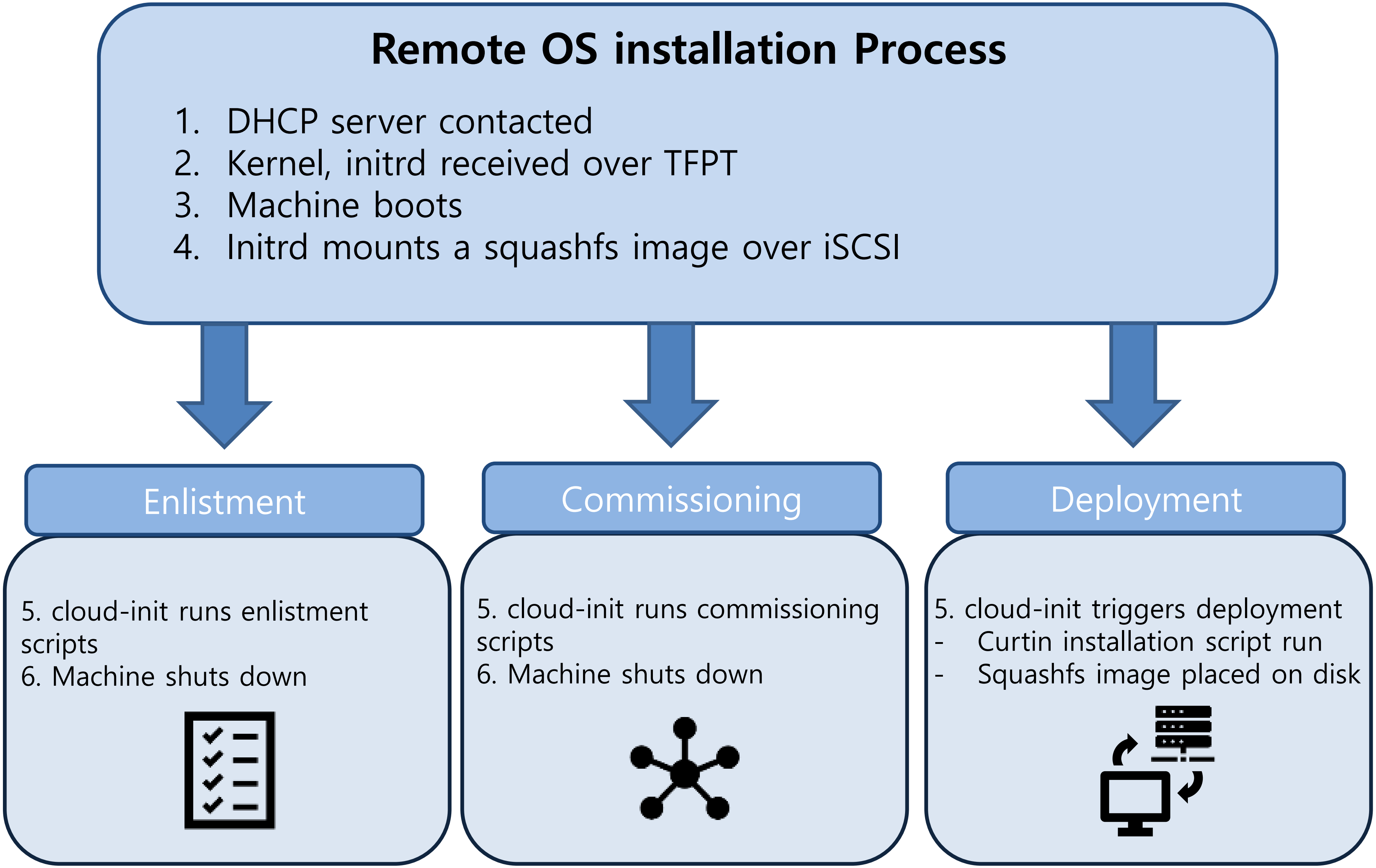
Warning!

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



Remote OS installation Process

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

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

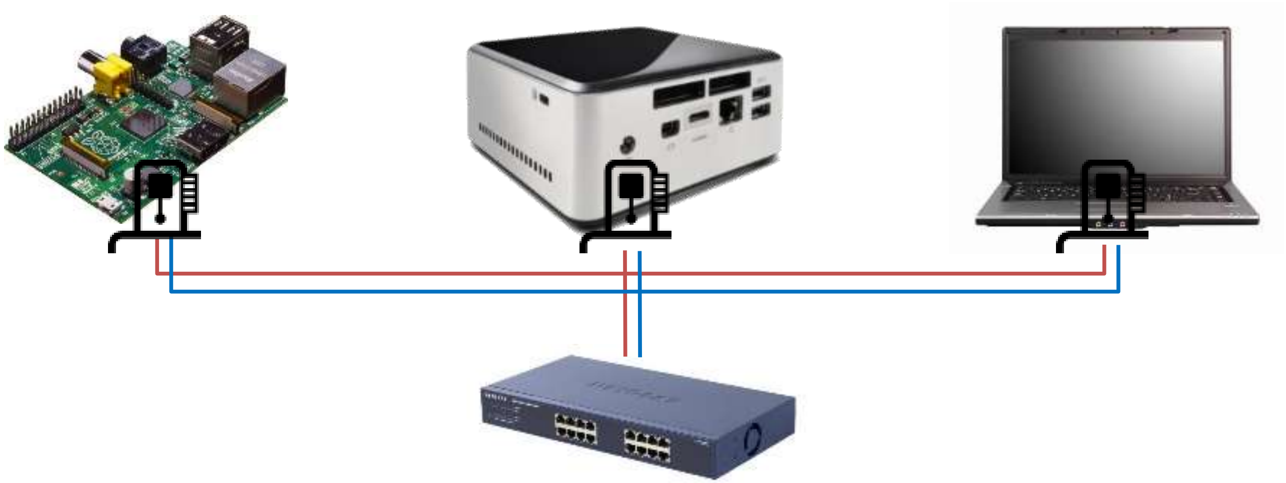
#0 - Lab Preparation (1/2)

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

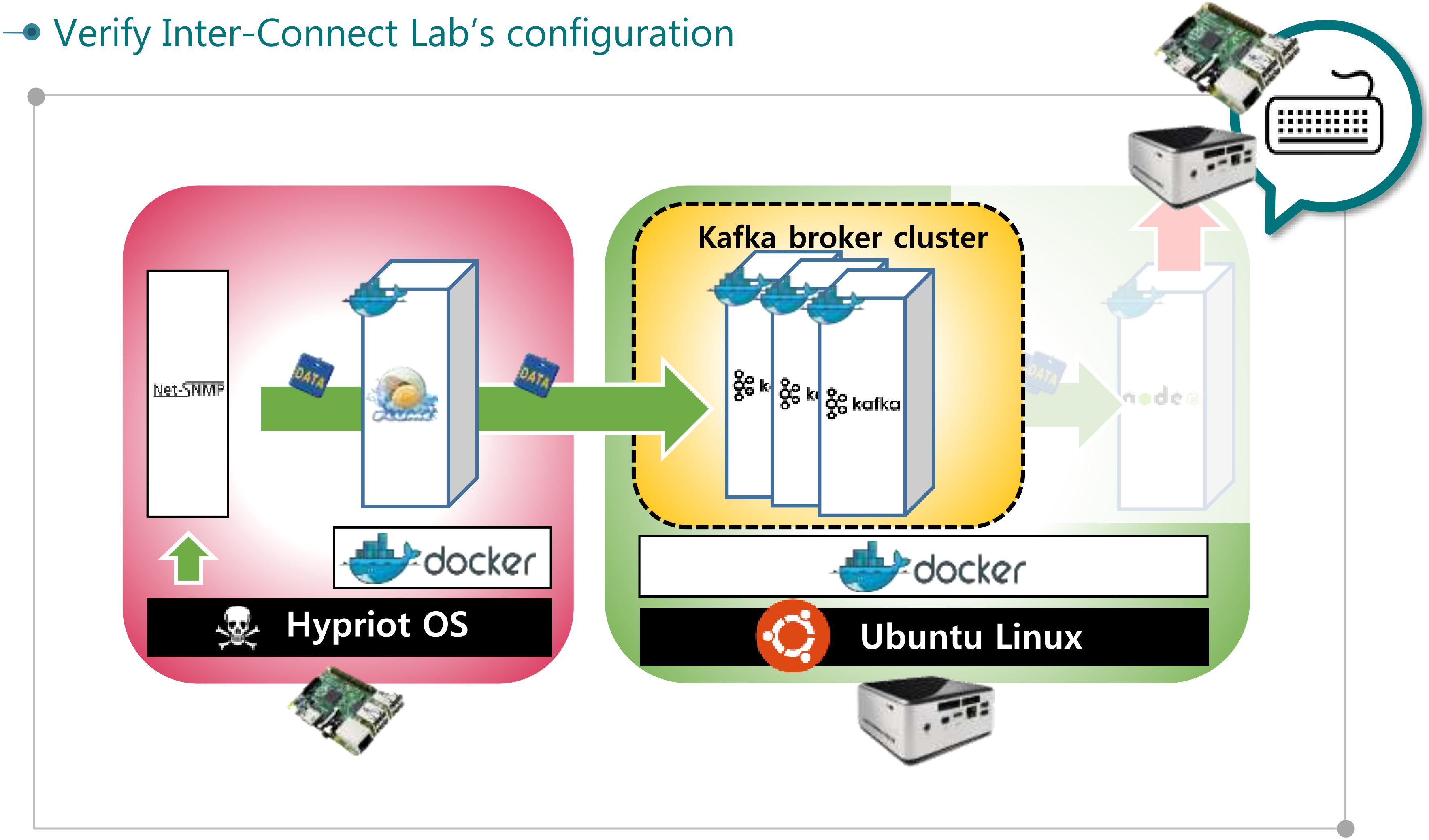
NAME: NT900X3A
CPU: i5-2537U @1.40GHz
CORE: 2
Memory: 4GB DDR3
HDD: 128GB



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

#0 - Lab Preparation (2/2)

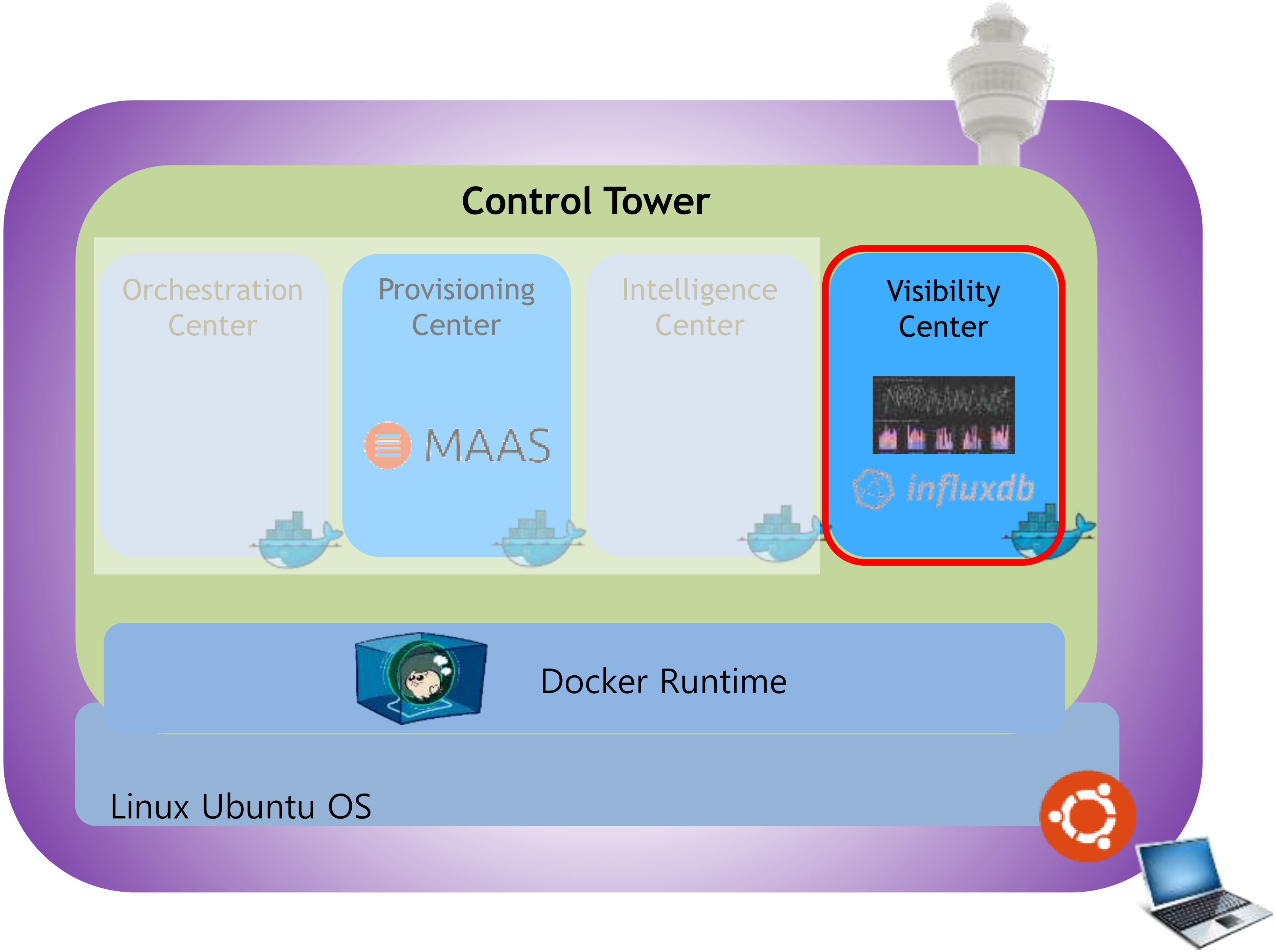
Verify Inter-Connect Lab's configuration



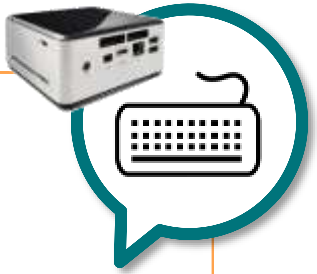
Are they working?

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

Visualization of Resource Visibility



#1 Run InfluxDB & Chronograf Containers on NUC



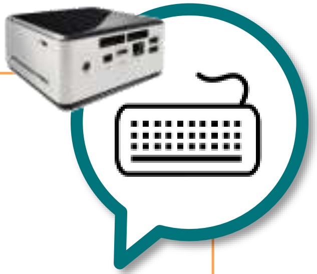
- Run InfluxDB Container

```
$ 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
```

#2 Install python packages:for python Kafka consumer



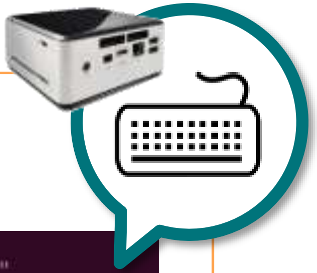
- Install python-pip

```
$ 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

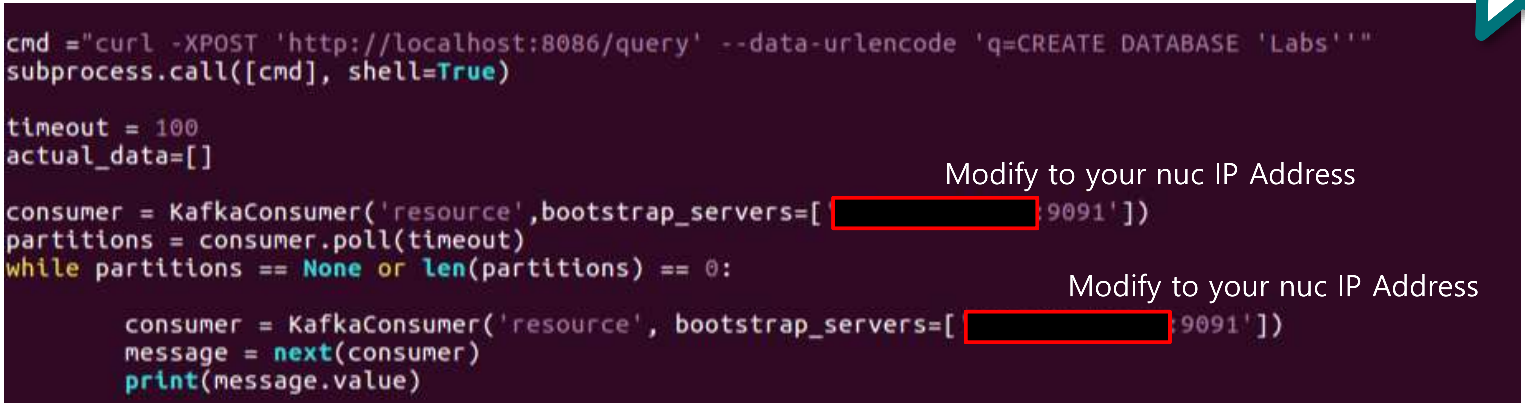
```
$ sudo pip install requests
$ sudo pip install kafka-python
$ sudo pip install influxdb
$ sudo pip install msgpack
```

#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

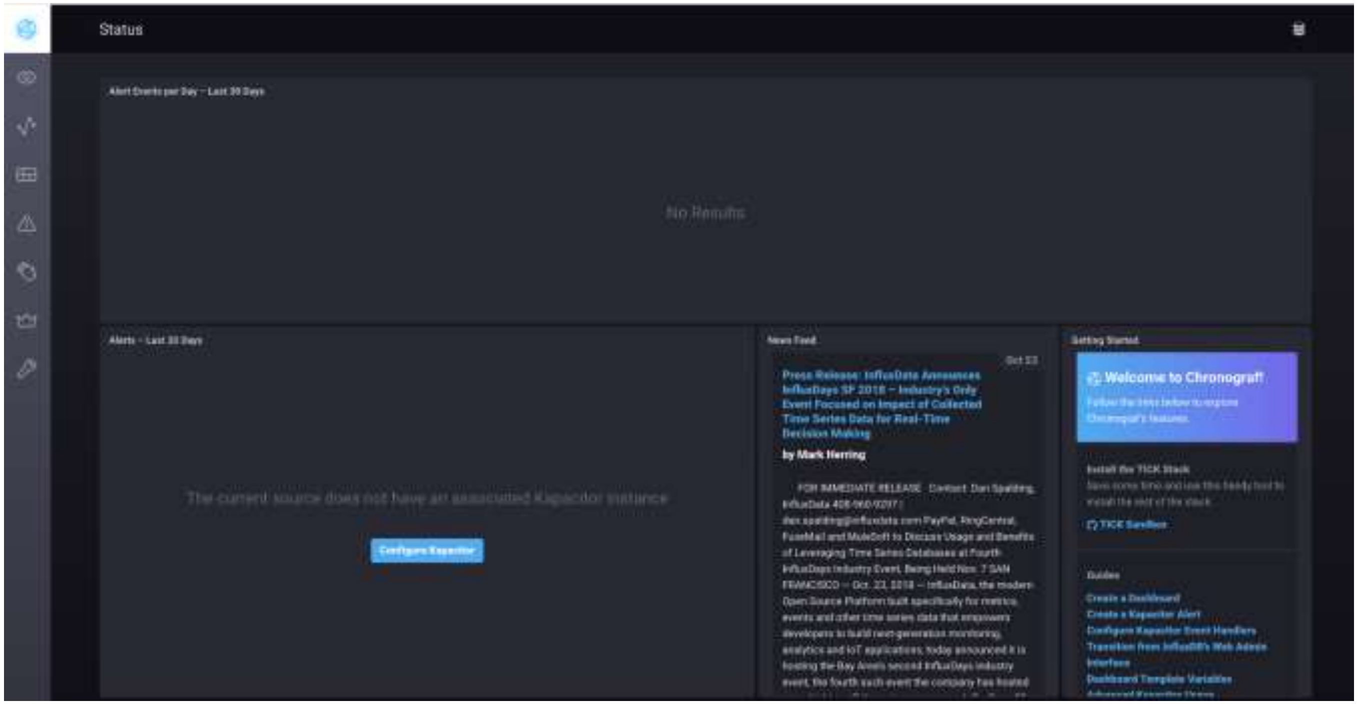
- Run python code

$ 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



Status

Alert Events per Day – Last 30 Days

No Results

Alerts – Last 30 Days

The current source does not have an associated Kapacitor instance

Configure Kapacitor

News Feed

Oct 23

Press Release: InfluxData Announces InfluxDays SF 2018 – Industry's Only Event Focused on Impact of Collected Time Series Data for Real-Time Decision Making

by Mark Herring

FOR IMMEDIATE RELEASE Contact: Dan Spalding, InfluxData 408-960-0297 | dan.spalding@influxdata.com PayPal, RingCentral, FuseMail and MuleSoft to Discuss Usage and Benefits of Leveraging Time Series Databases at Fourth InfluxDays Industry Event, Being Held Nov. 7 SAN FRANCISCO — Oct. 23, 2018 — InfluxData, the modern Open Source Platform built specifically for metrics, events and other time series data that empowers developers to build next-generation monitoring, analytics and IoT applications, today announced it is hosting the Bay Area's second InfluxDays industry event, the fourth such event the company has hosted

Getting Started

Welcome to Chronograf!

Follow the steps below to explore Chronograf's features.

Install the TICK Stack

Save some time and use this handy tool to install the rest of the stack.

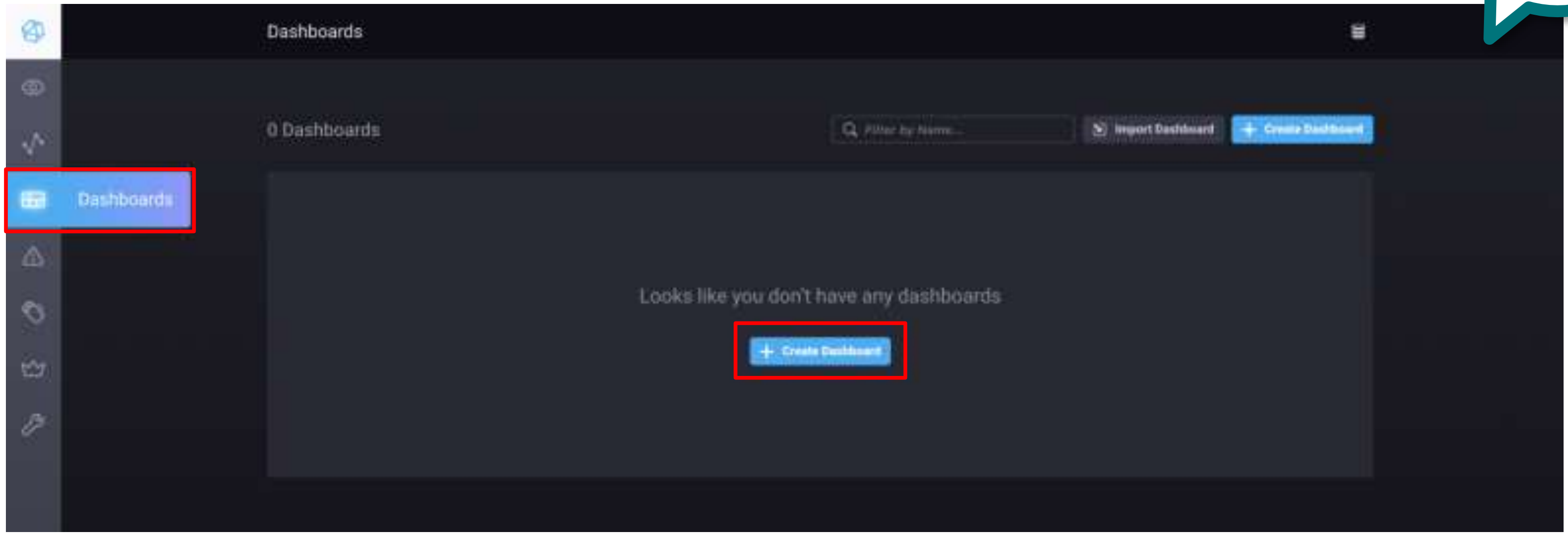
TICK Sandbox

Guides

Create a Dashboard
Create a Kapacitor Alert
Configure Kapacitor Event Handlers
Transition from InfluxDB's Web Admin Interface
Dashboard Template Variables
Advanced Kapacitor Usage

#4 Configure Chronograf Dashboard (2/4)

- Create Dash board

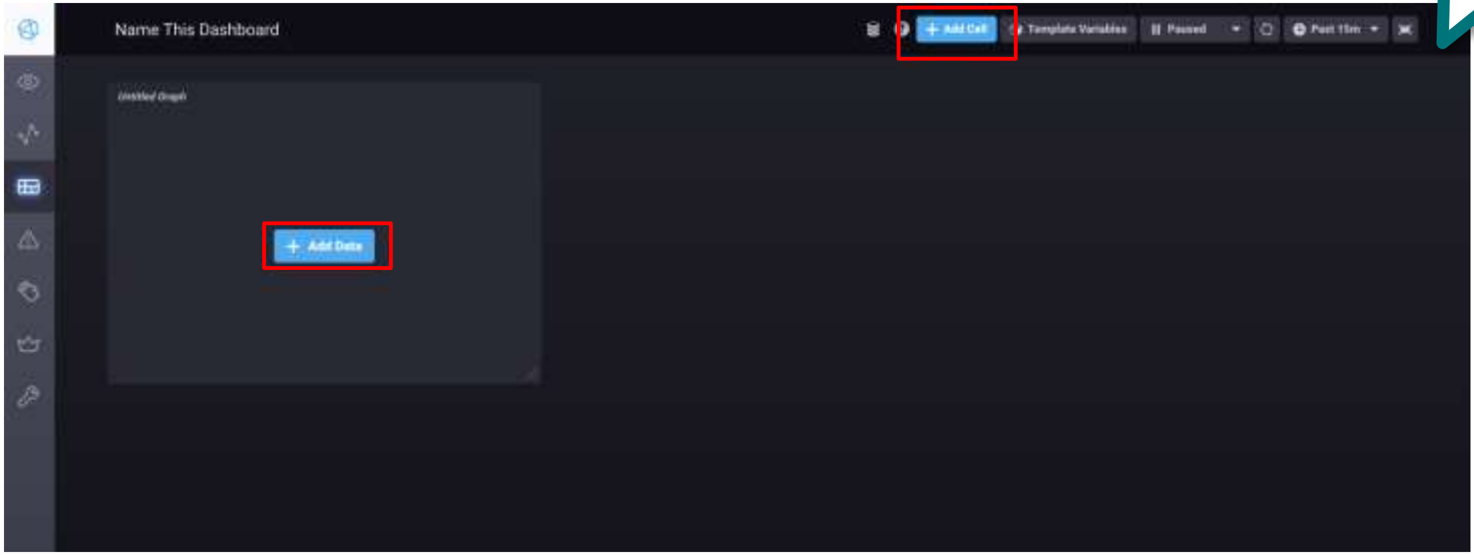


0 Dashboards

Looks like you don't have any dashboards

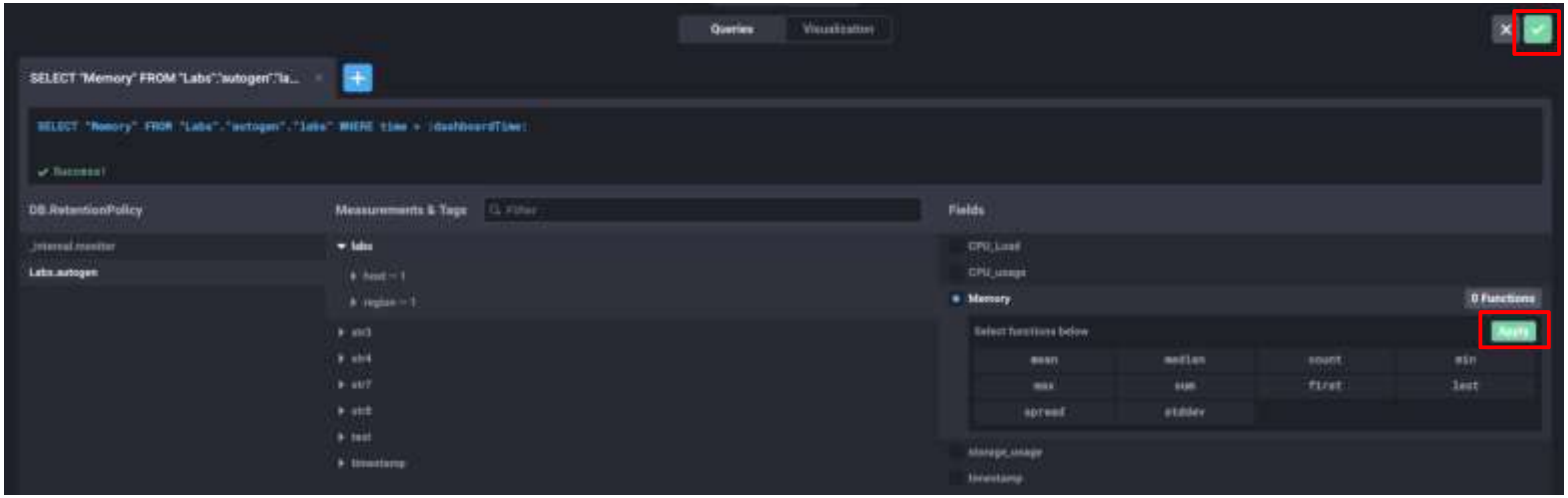
#4 Configure Chronograf Dashboard (3/4)

- Add Cell & Data



#4 Configure Chronograf Dashboard (4/4)

- Add Data

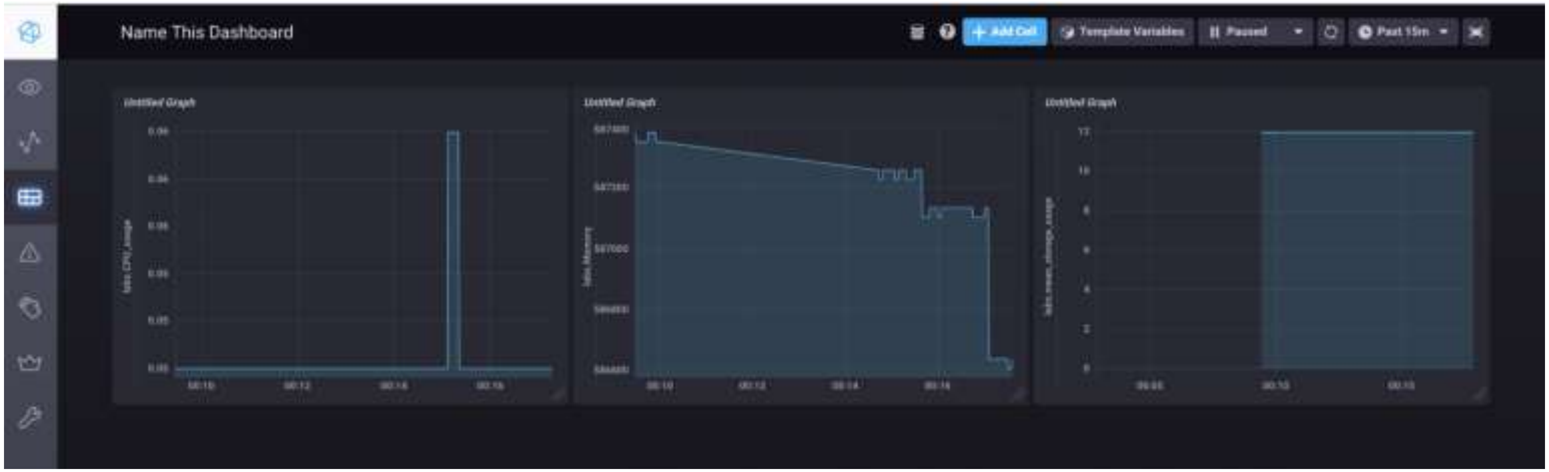


```
SELECT "Memory" FROM "Labs"."autogen"."labs" WHERE time > :dashboardTime:
```

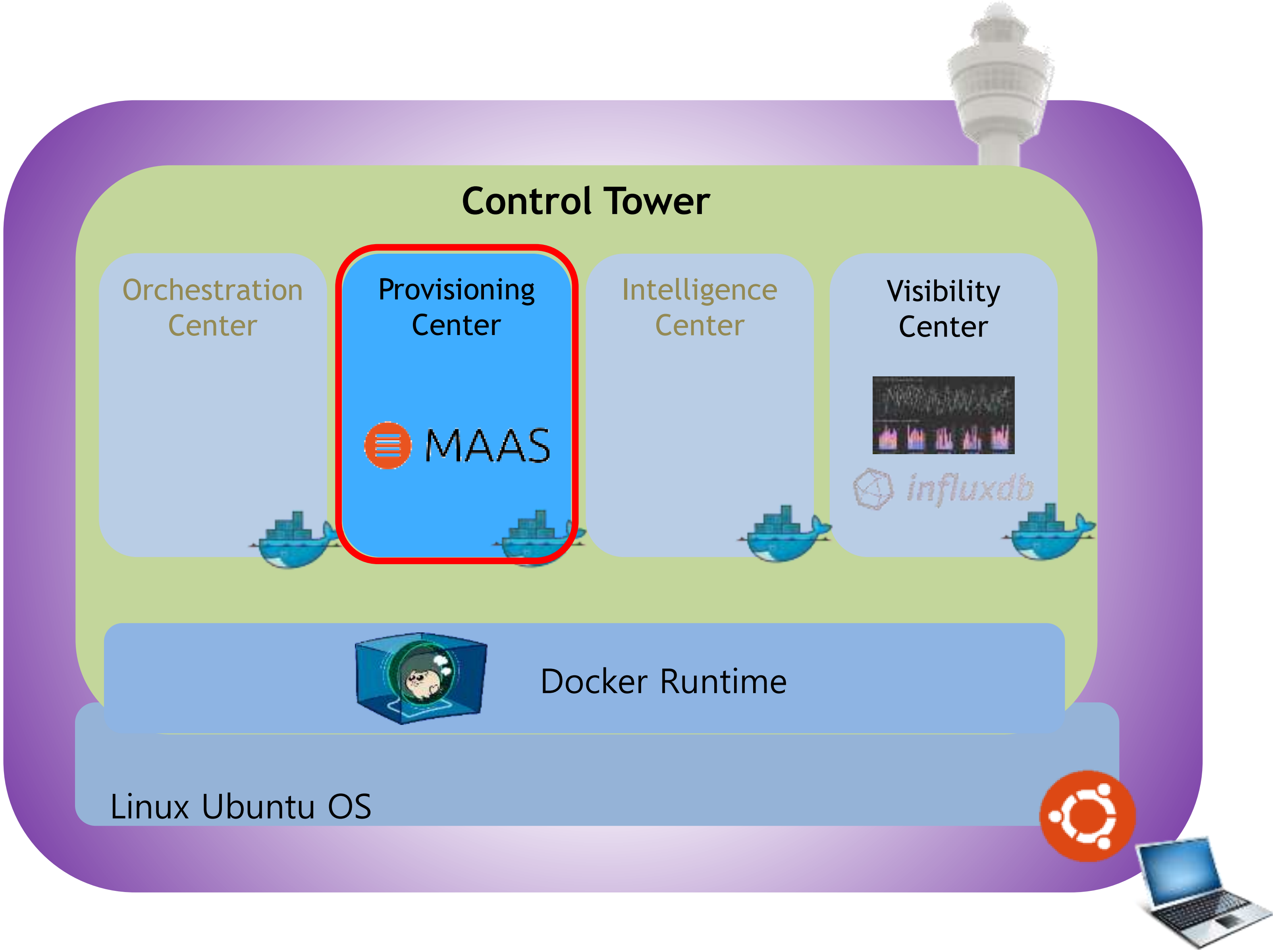
✓ Success!

#5 – Check Chronograf Dashboard

- We can see the changes of values from database

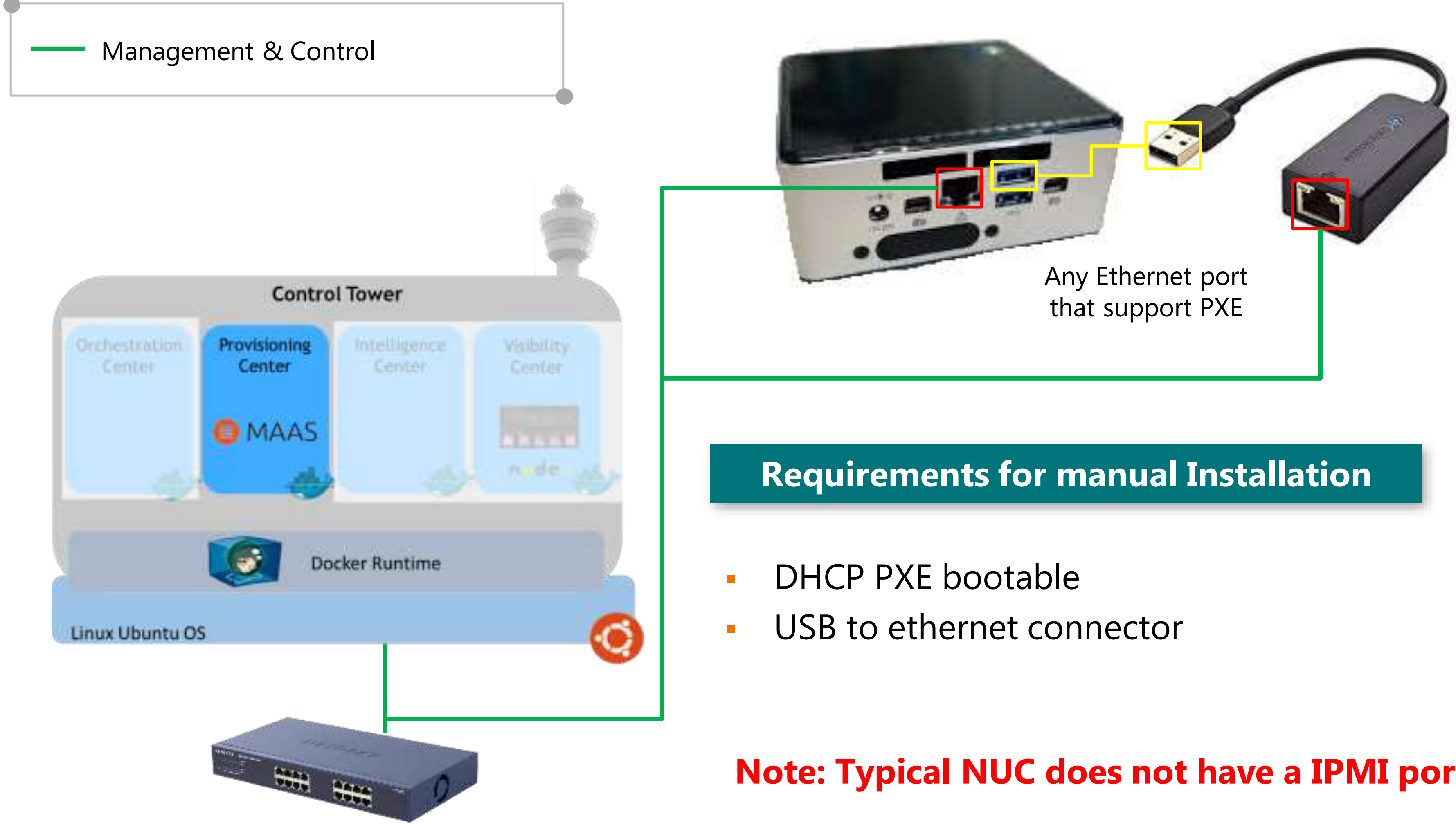


Automated Provisioning & Visibility



#6–1 OS manual Installation: Establish physical interconnect

🟩 Management & Control

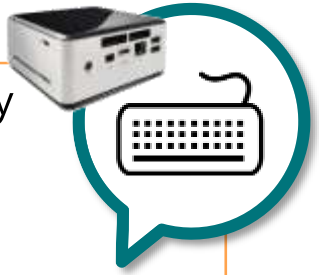


Requirements for manual Installation

- 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
```

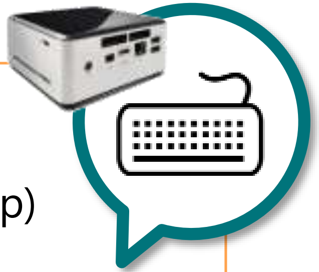
- Login to the MAAS UI at:

  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

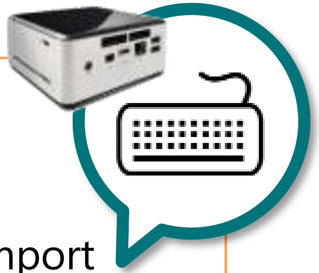


- Enlist the NUC
  - Set all the servers to PXE boot
  - NUC reboot (When hardware is initialized, all software operations stop)
  - Check the NUC appear in MAAS
  - If the NUC does not support IPMI based BMC, edit them and enter their BMC details

- 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

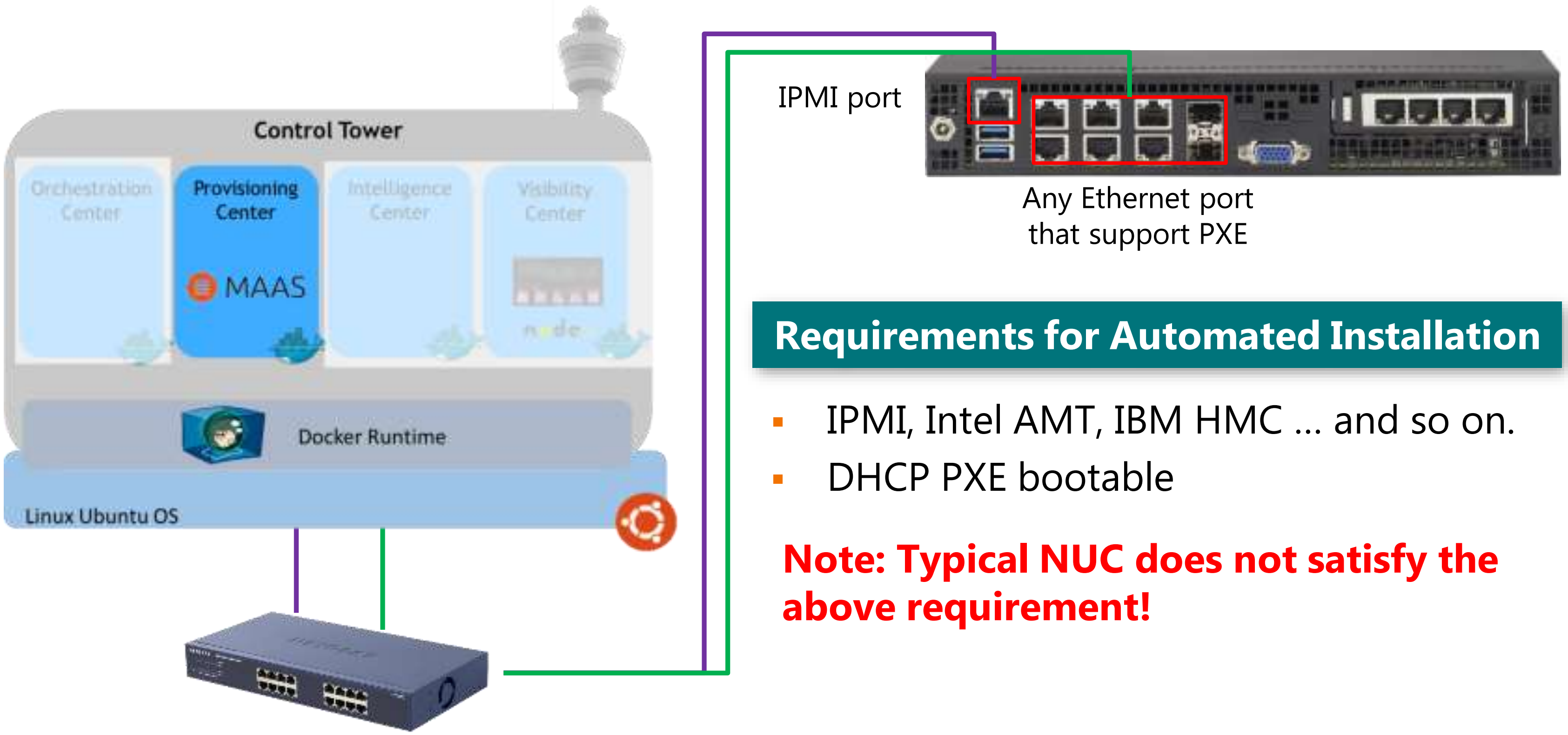
- From “take action”, choose “deploy”
- OS remote Install complete

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

━ Power
━ Management & Control

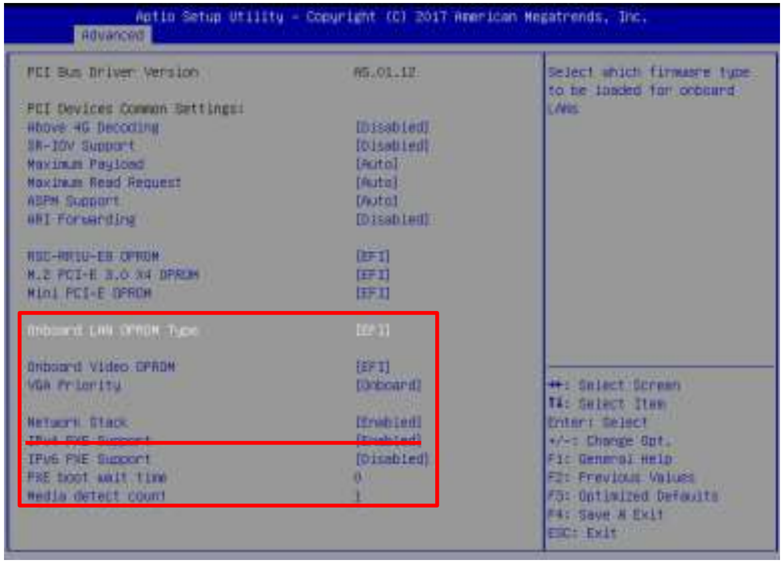


Requirements for Automated Installation

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

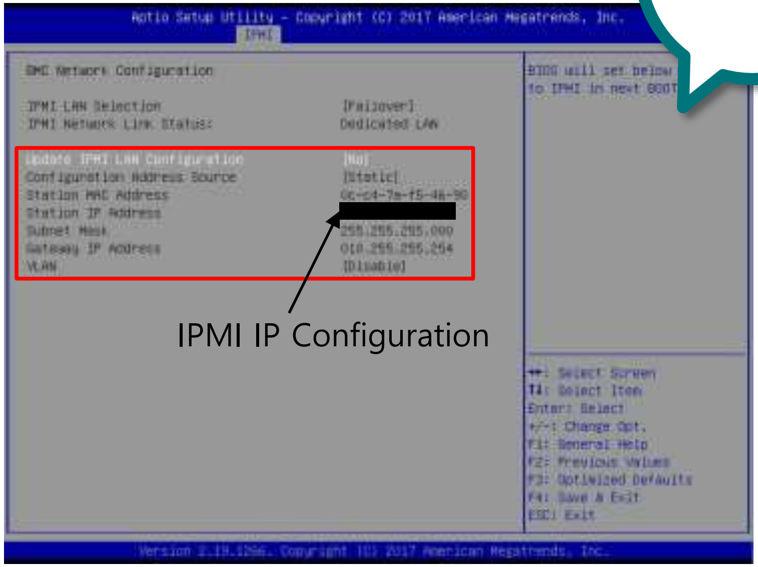
Note: Typical NUC does not satisfy the above requirement!

#7–2 OS Automatic Installation: BIOS Configuration



```
          Aptio Setup Utility - Copyright (C) 2017 American Megatrends, Inc.
      Advanced

 PCI Bus Driver Version                     A5.01.12        | Select which firmware type
                                                            | to be loaded for onboard
 PCI Devices Common Settings:                               | LANs
 Above 4G Decoding                          [Disabled]      |
 SR-IOV Support                             [Disabled]      |
 Maximum Payload                            [Auto]          |
 Maximum Read Request                       [Auto]          |
 ASPM Support                               [Auto]          |
 ARI Forwarding                             [Disabled]      |
                                                            |
 RSC-RR1U-E8 OPROM                          [EFI]           |
 M.2 PCI-E 3.0 X4 OPROM                     [EFI]           |
 Mini PCI-E OPROM                           [EFI]           |
                                                            |
 Onboard LAN OPROM Type                     [EFI]           |
                                                            |
 Onboard Video OPROM                        [EFI]           | ++: Select Screen
 VGA Priority                               [Onboard]       | ↑↓: Select Item
                                                            | Enter: Select
 Network Stack                              [Enabled]       | +/-: Change Opt.
 ~~IPv4 PXE Support                          [Enabled]~~     | F1: General Help
 IPv6 PXE Support                           [Disabled]      | F2: Previous Values
 PXE boot wait time                         0               | F3: Optimized Defaults
 Media detect count                         1               | F4: Save & Exit
                                                            | ESC: Exit
```

<BIOS PXE Configuration>



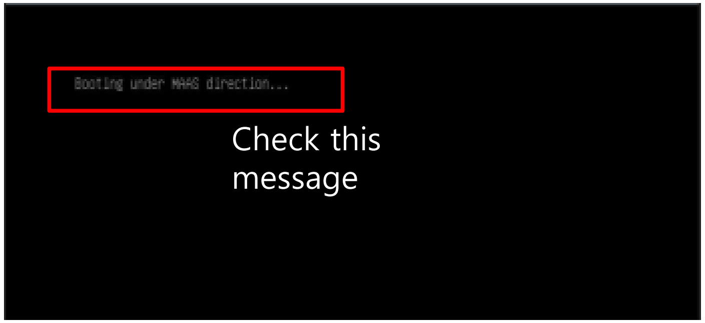
```
          Aptio Setup Utility - Copyright (C) 2017 American Megatrends, Inc.
                                  IPMI

 BMC Network Configuration                                  | BIOS will set below
                                                            | to IPMI in next BOOT
 IPMI LAN Selection                         [Failover]      |
 IPMI Network Link Status:                  Dedicated LAN   |
                                                            |
 Update IPMI LAN Configuration              [No]            |
 Configuration Address Source               [Static]        |
 Station MAC Address                        0c-c4-7a-f5-46-90
 Station IP Address                         [redacted]      |
 Subnet Mask                                255.255.255.000 |
 Gateway IP Address                         010.255.255.254 |
 VLAN                                       [Disable]       |
                                                            | ++: Select Screen
                                                            | ↑↓: Select Item
                                                            | Enter: Select
                                                            | +/-: Change Opt.
                                                            | F1: General Help
                                                            | F2: Previous Values
                                                            | F3: Optimized Defaults
                                                            | F4: Save & Exit
                                                            | ESC: Exit
          Version 2.19.1266. Copyright (C) 2017 American Megatrends, Inc.
```

IPMI IP Configuration

<BIOS IPMI Configuration>

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

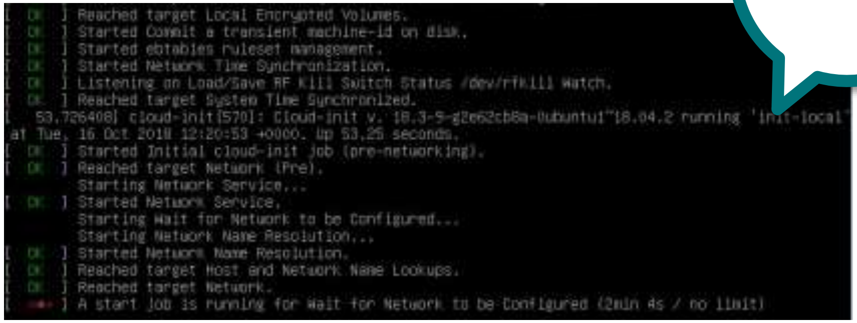
#8–1 OS Automatic Installation:Enlistment



```
Booting under MAAS direction...
```

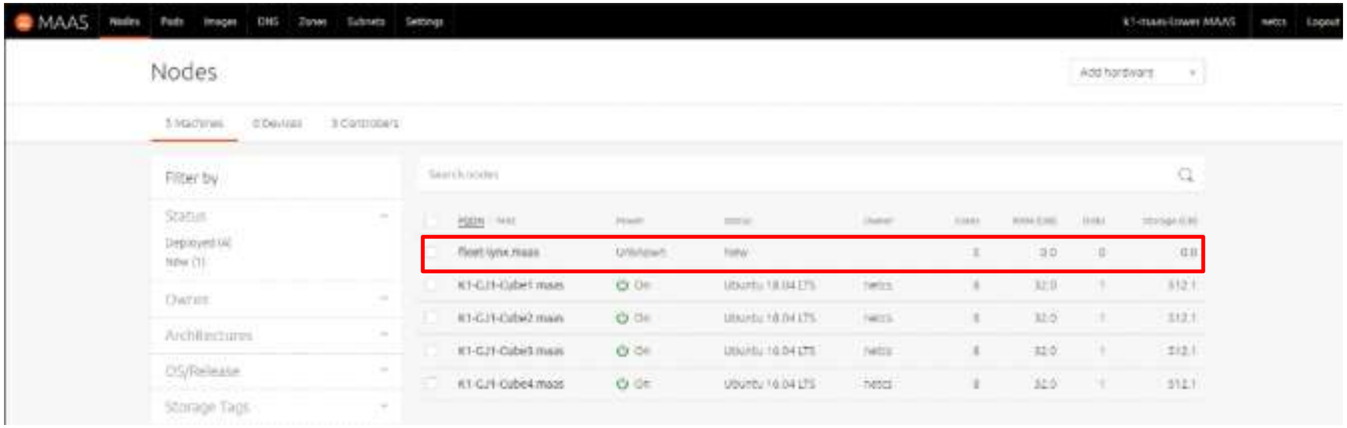
Check this message

<Succeed PXE booting>



```
[  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)
```

<Enlistment after PXE booting>



Nodes

5 Machines | 0 Devices | 3 Controllers

Filter by

Status
Deployed (4)
New (1)

Owner

Architectures

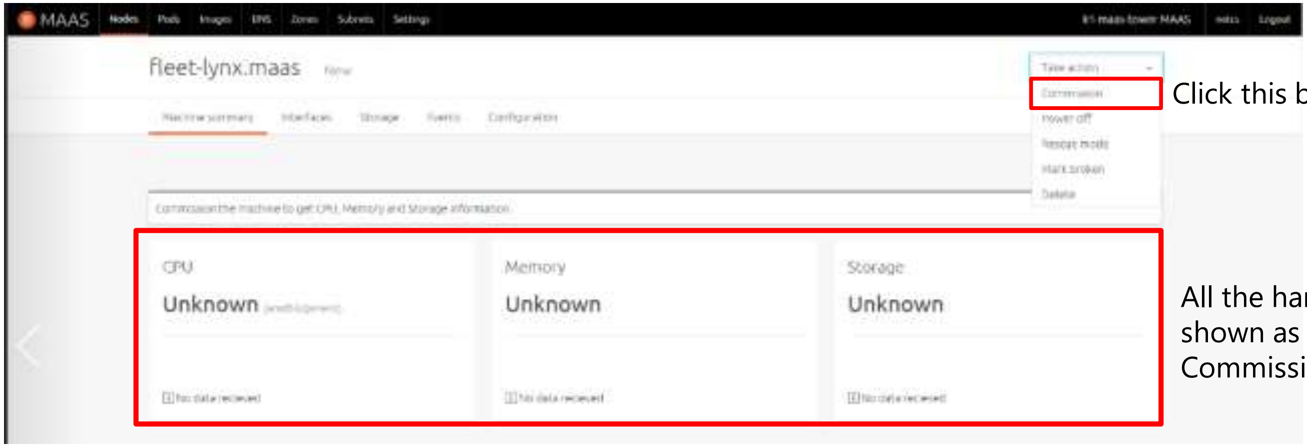
OS/Release

Storage Tags

FQDN | MAC	Power	Status	Owner	Cores	RAM (GiB)	Disks	Storage (GB)
fleet-lynx.maas	Unknown	New		1	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



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

fleet-lynx.maas New

Machine summary | Interfaces | Storage | Events | Configuration

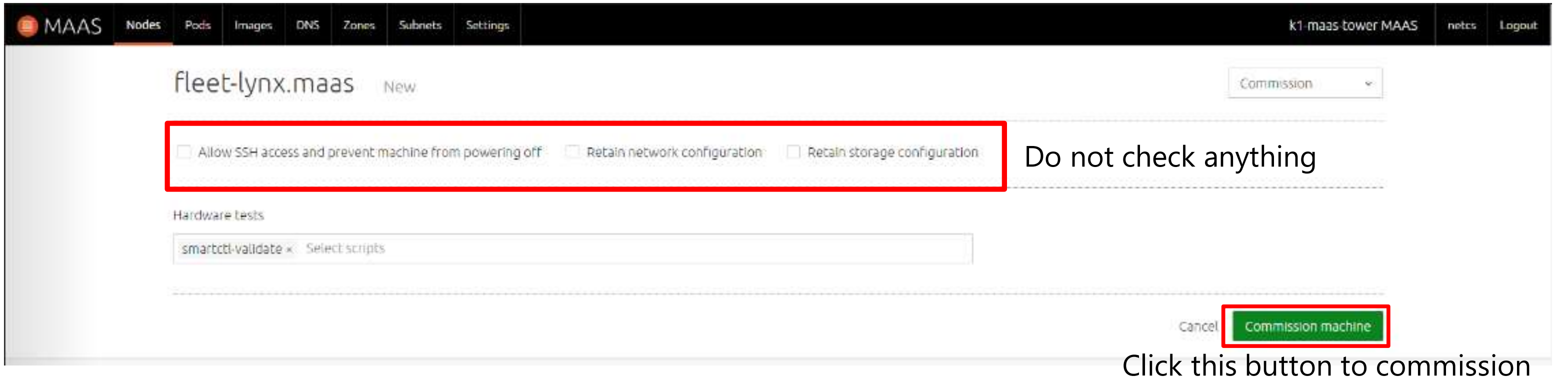
Take action:
- Commission
- Power off
- Rescue mode
- Mark broken
- Delete

Commission the machine to get CPU, Memory and Storage information

CPU	Memory	Storage
Unknown (architecture)	Unknown	Unknown
No data recieved	No data recieved	No data recieved

Click this button to commission

All the hardware information shown as ‘Unknown’ before Commissioning



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

fleet-lynx.maas New

Commission

☐ Allow SSH access and prevent machine from powering off ☐ Retain network configuration ☐ Retain storage configuration

Hardware tests

smartctl-validate × Select scripts

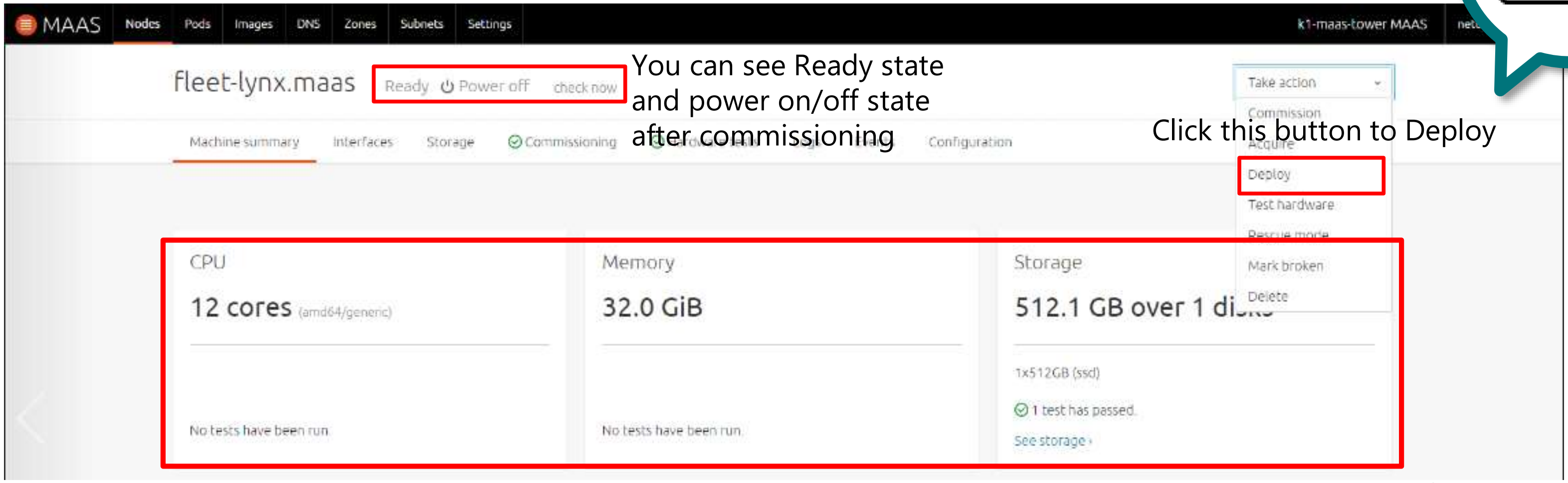
Cancel | Commission machine

Do not check anything

Click this button to commission

- It takes about 10 minutes

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



CPU
12 cores (amd64/generic)
No tests have been run

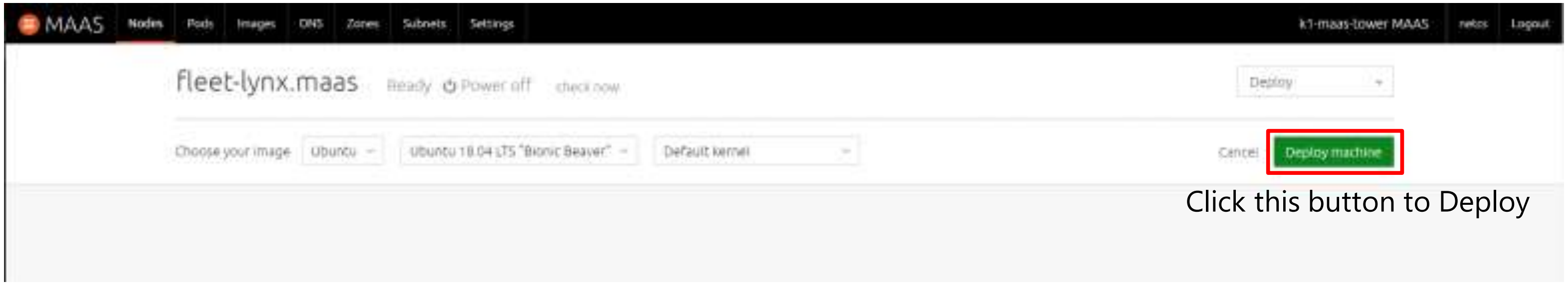
Memory
32.0 GiB
No tests have been run

Storage
512.1 GB over 1 disks
1x512GB (ssd)
1 test has passed.
See storage ›

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

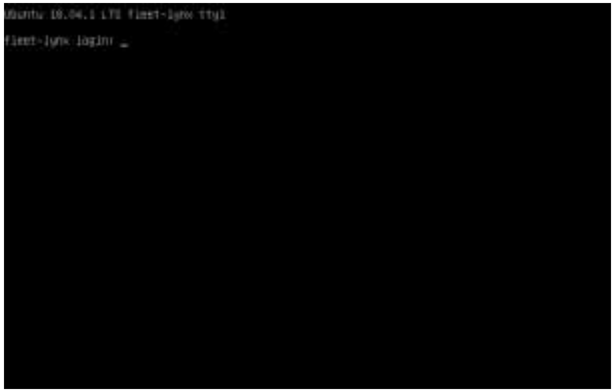


Choose your image: Ubuntu — Ubuntu 18.04 LTS "Bionic Beaver" — Default kernel

Click this button to Deploy

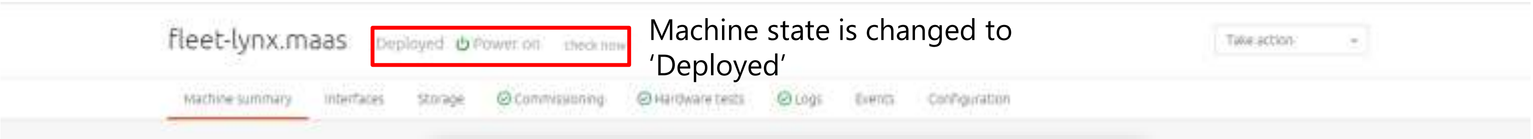
- It is a OS Installation procedure
- It takes about 15 minutes

#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’

<Complete Deploying (On Web GUI)>

Lab Review

Lab Summary

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