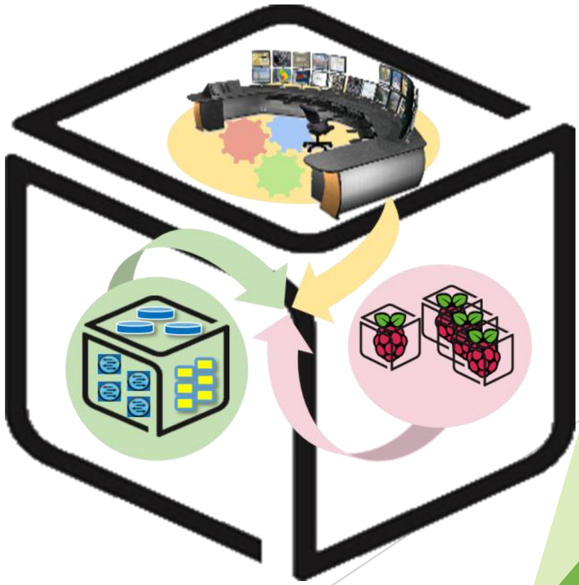
SmartX Labs for Computer Systems

InterConnect Lab

(2018, Spring)



NetCS Lab

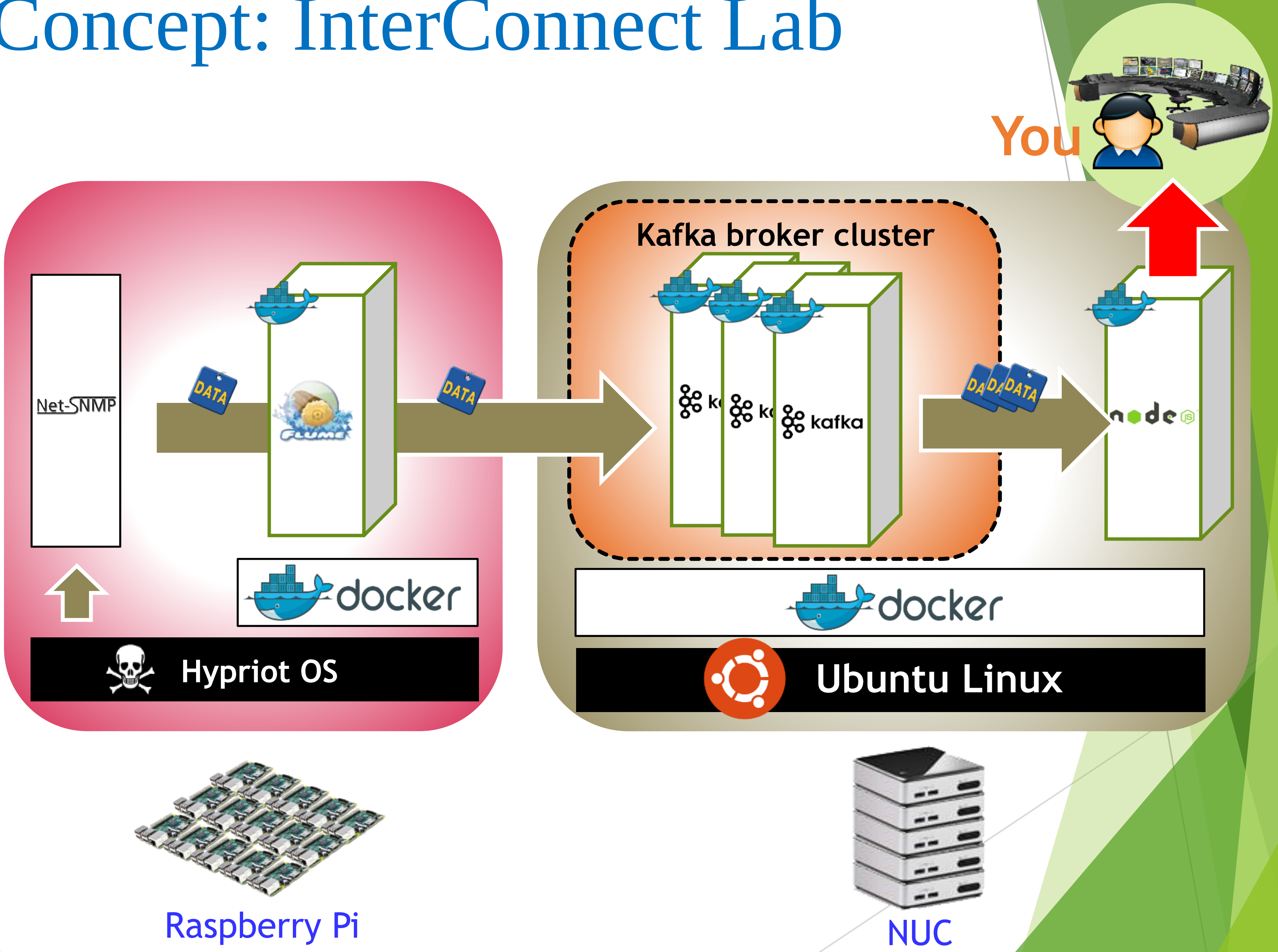
History and Contributor of InterConnect Lab (2018. 04. 30)

Version	Updated Date	Updated Contents	Contributor
v0.1	2017/05	Merged with InterConnect & Tower Lab	Seungryong Kim
v0.2	2018/01	InfluxDB 버전 업데이트 스크립트 변경	Seunghyung Lee
v0.3	2018/04	Tower Lab 부분 분리, 일부 설명 수정	Moonjoong Kang

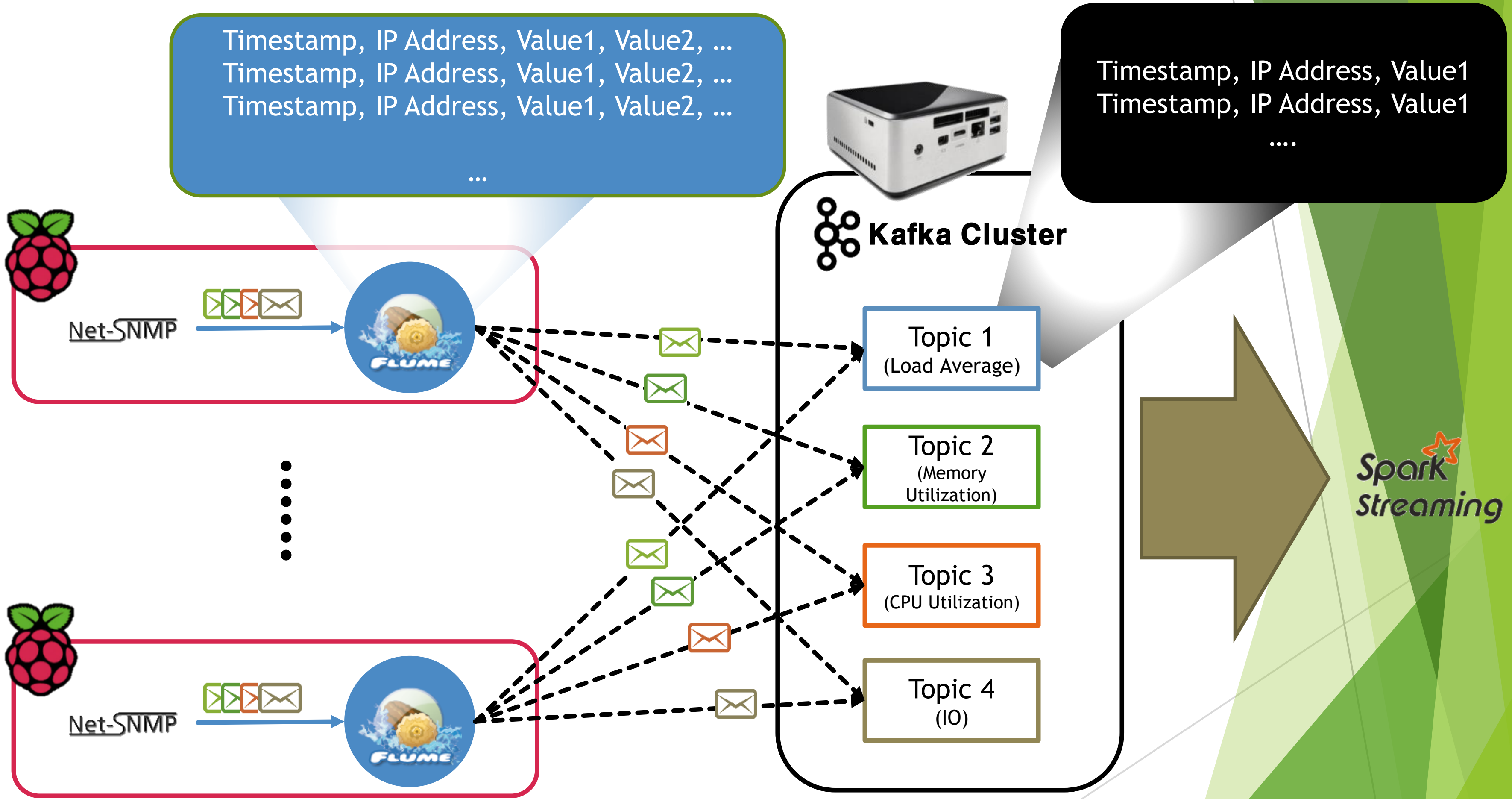
Goals

- Understanding Concepts
  - Net-SNMP, Flume, Kafka, Hypriot OS
- Connecting with each functions
  - With Raspberry Pi and NUC

Concept: InterConnect Lab



Cont’d

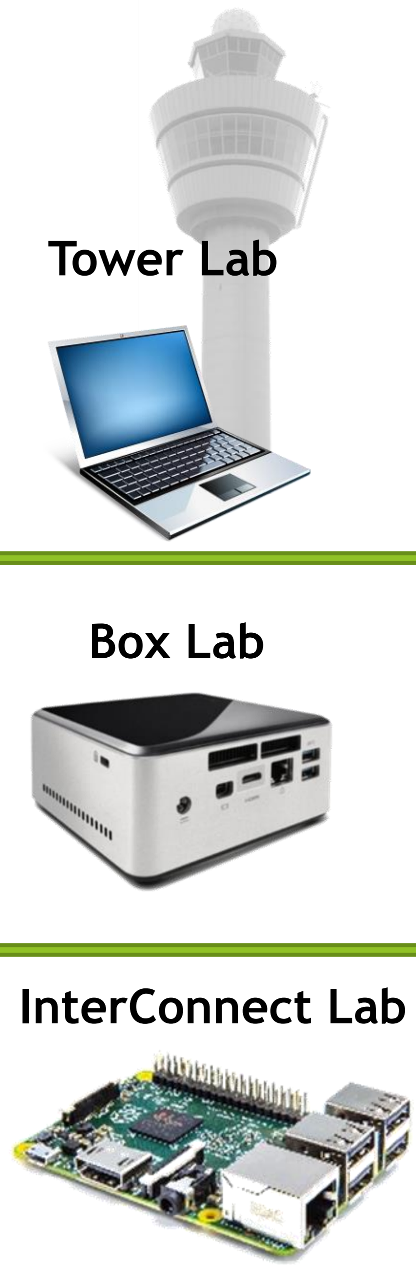

Timestamp, IP Address, Value1, Value2, …
Timestamp, IP Address, Value1, Value2, …
Timestamp, IP Address, Value1, Value2, …
…
Timestamp, IP Address, Value1
Timestamp, IP Address, Value1
….
Kafka Cluster
Net-SNMP
FLUME
Topic 1 (Load Average)
Topic 2 (Memory Utilization)
Topic 3 (CPU Utilization)
Topic 4 (IO)
Net-SNMP
FLUME
Spark Streaming

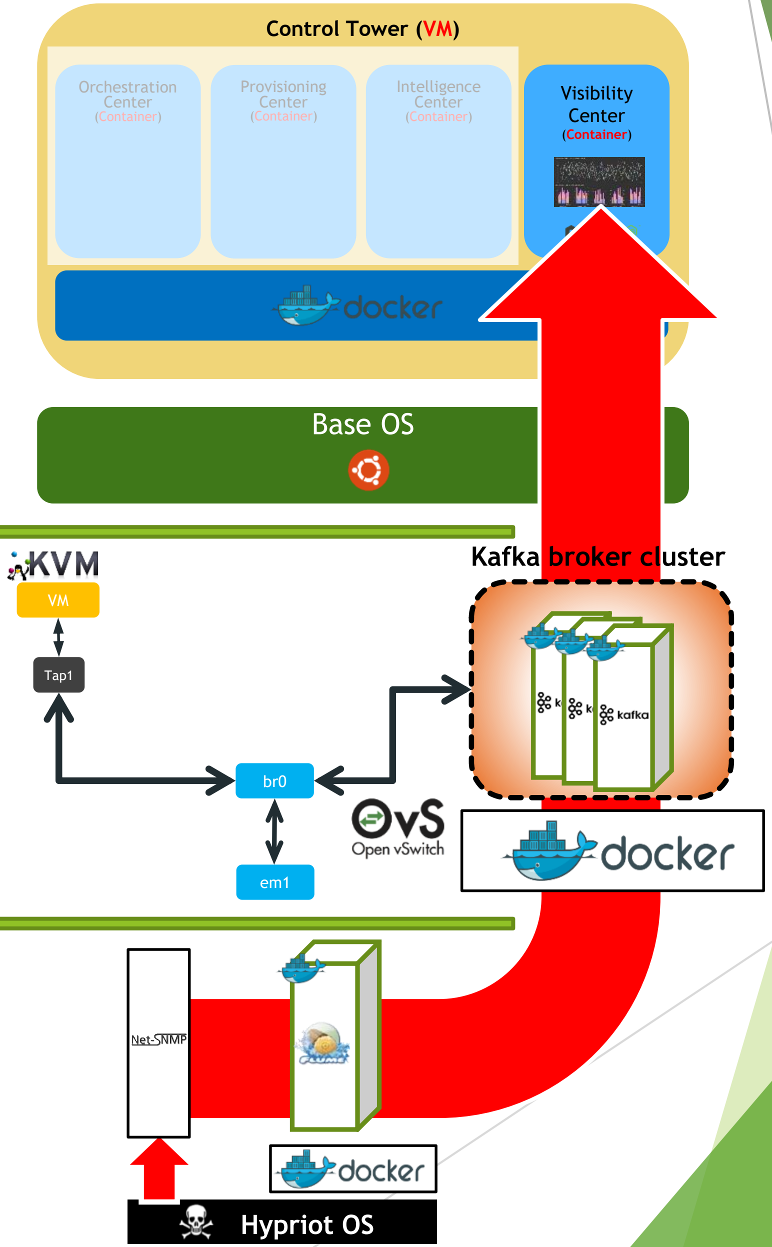
Relation of SmartX Lab

Tower Lab

Box Lab

InterConnect Lab



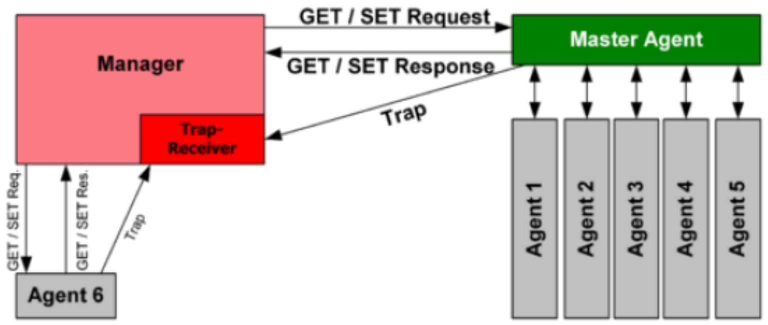


Control Tower (VM)
Orchestration Center (Container)
Provisioning Center (Container)
Intelligence Center (Container)
Visibility Center (Container)
docker
Base OS
KVM
VM
Tap1
br0
em1
Open vSwitch
Kafka broker cluster
kafka
docker
Net-SNMP
docker
Hypriot OS

Backgrounds

SNMP

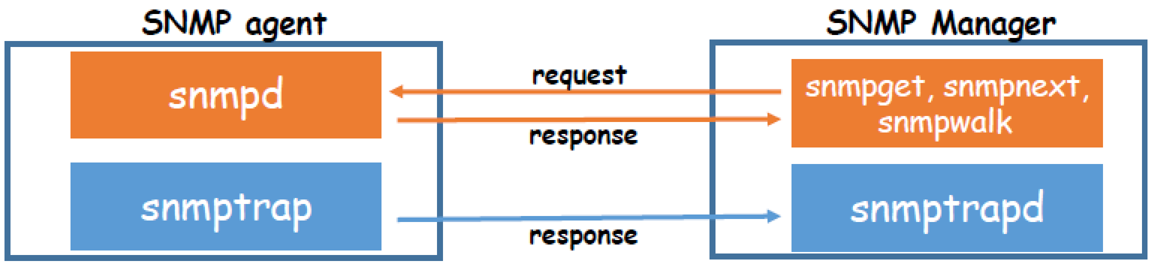
- SNMP: **Simple Network Management Protocol**
- Used in network management systems to monitor network-attached devices
- Include routers, switches, servers, workstations, printers, modem racks and more.



Backgrounds

Net-SNMP Net-SNMP

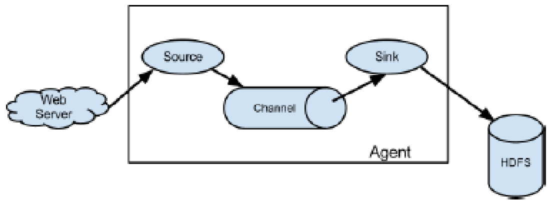
- A Suite of **software** for using and deploying the SNMP protocol



Backgrounds

Flume

- Log aggregator
  - Many customizable data sources
  - Flume can be used to handle them.
  - Run asynchronously



- Flume Agent
  - **Source**
    - Consuming events having a specific format.
    - Delivering it to the channel
  - **Channel**
    - Holding the event until that consumed
  - **Sink**
    - Removing an event from the channel.
    - Putting it into an external repository or another source.

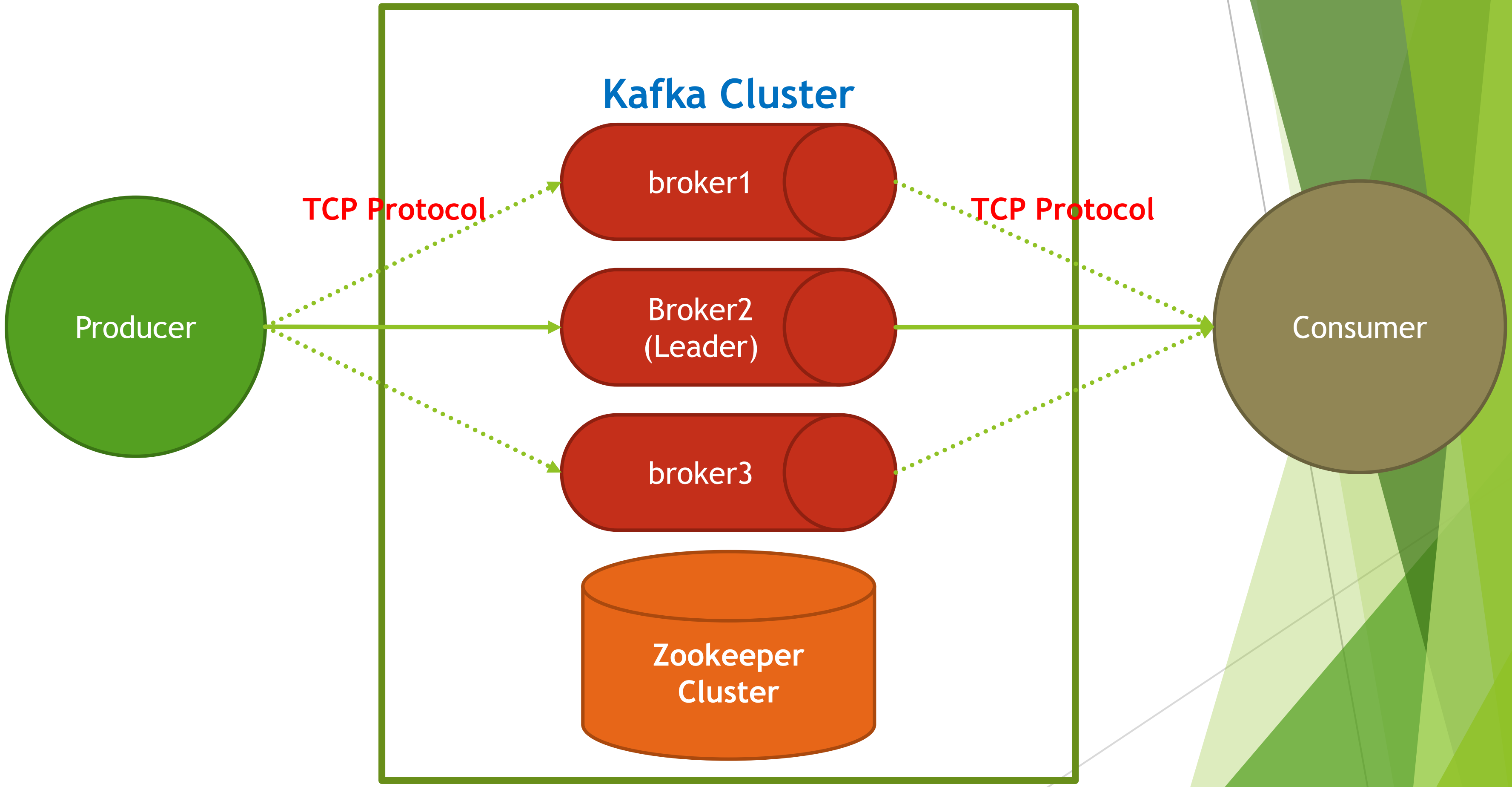
Backgrounds

Kafka

- Kafka
  - is a **distributed, partitioned, replicated** commit log service.
  - It provides **the functionality of a messaging system**, but with a unique design
- Basic messaging terminology
  - ***Topics:***
    - maintains feeds of **messages in categories**
  - ***Producers:***
    - processes that **publish messages** to a Kafka topic
  - ***Consumers:***
    - processes that subscribe to topics and process **the feed of published messages**
  - ***Broker:***
    - run as a cluster comprised of one or more servers

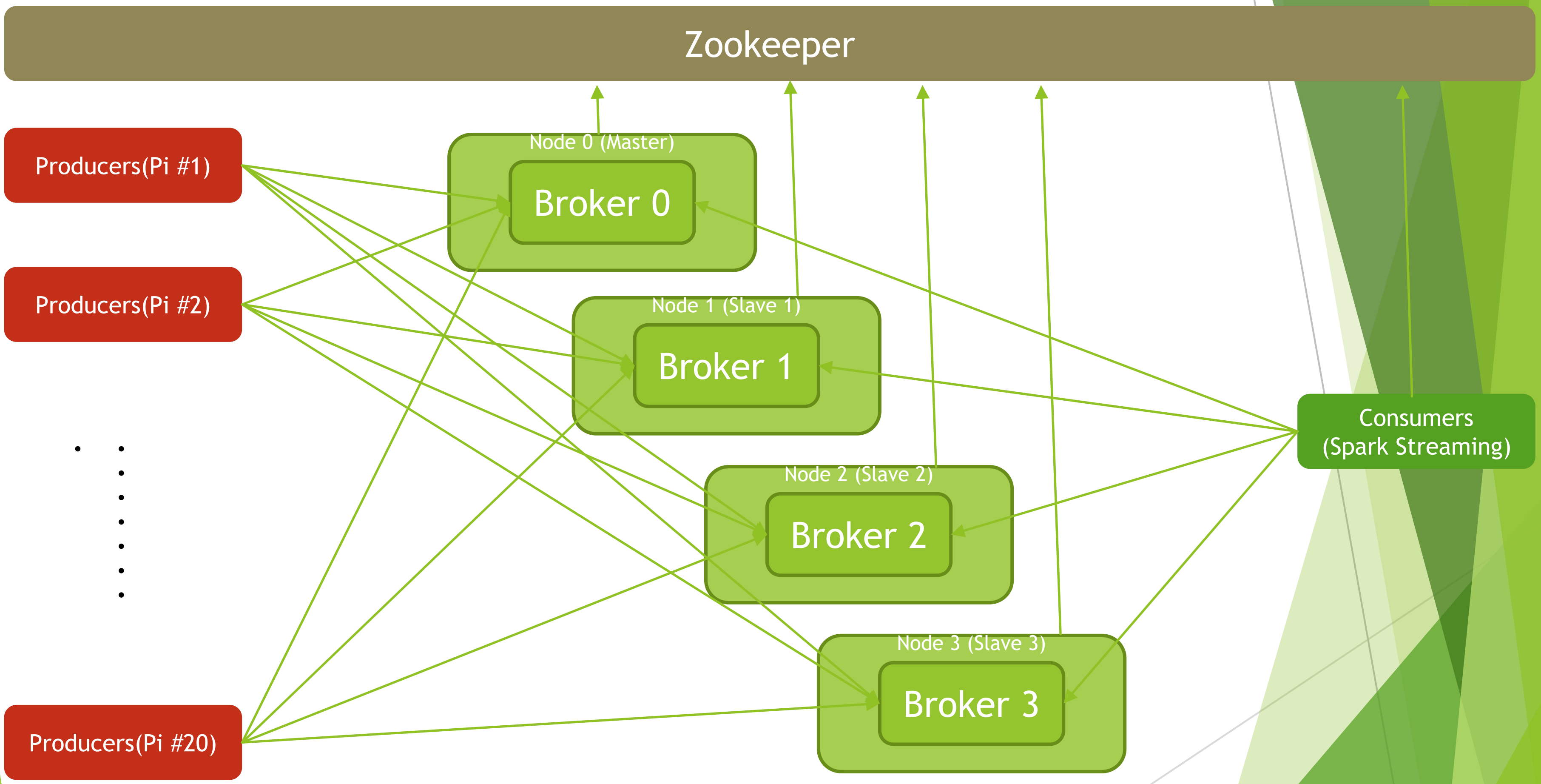
Backgrounds

Kafka: Architecture



Backgrounds

Cont'd

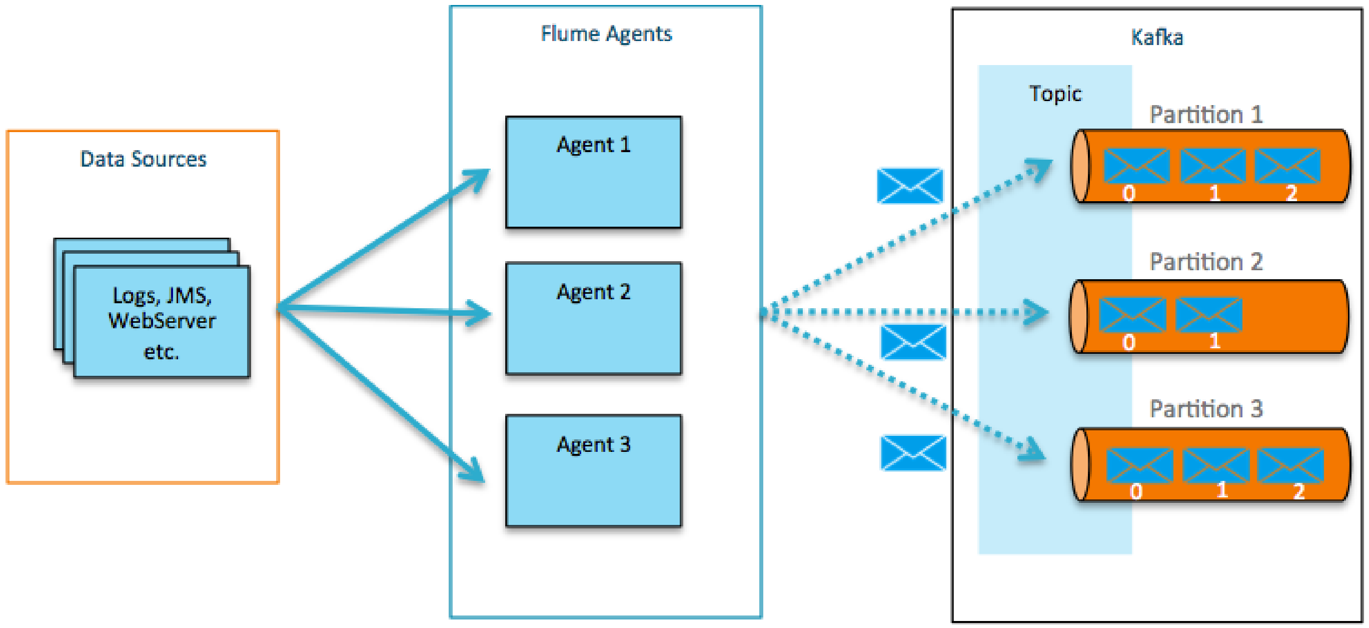


Backgrounds

Kafka: with Flume

- **Flafka**

  : Apache Flume Meets Apache Kafka for Event Processing

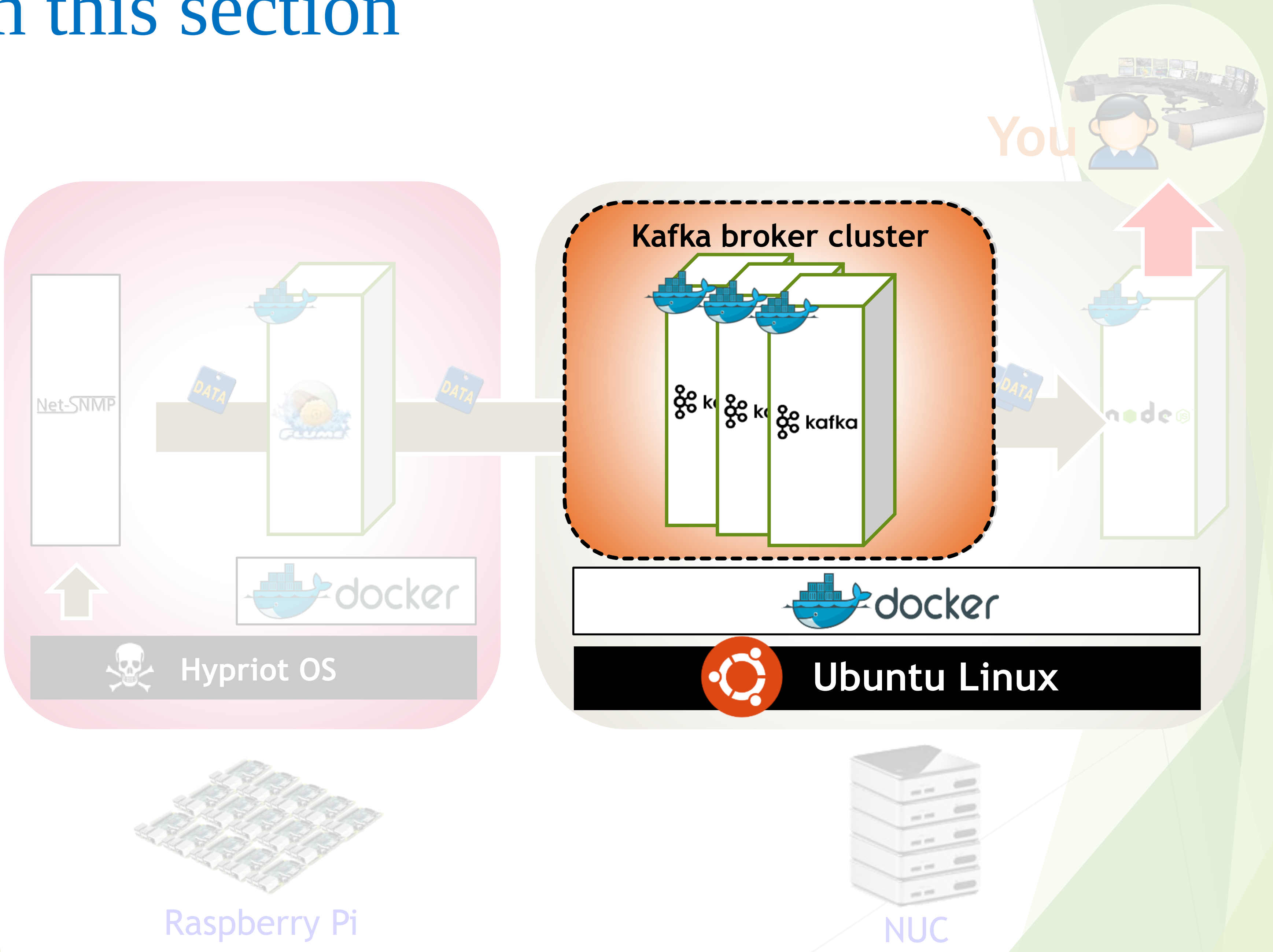


Backgrounds

- Docker is an open platform for building, shipping and running distributed applications. It gives programmers, development teams and operations engineers the common toolbox they need to take advantage of the distributed and networked nature of modern applications.

Connecting Configuration on NUC

In this section



0. Install HypriotOS on Raspberry Pi

- Before we start, your Raspberry Pi must be ready with proper OS.
- In this lab, we will use "HypriotOS" Linux for it.

1. Eject a MicroSD card from your Raspberry Pi, and insert it into your SD card reader and attach the reader to your NUC.
2. Issue the commands below to get "flash" script for the OS setup.
   - `$ sudo apt update && sudo apt install -y pv curl python-pip unzip hdparm`
   - `$ sudo pip install awscli`
   - `$ curl -O https://raw.githubusercontent.com/hypriot/flash/master/flash`
   - `$ chmod +x flash`
   - `$ sudo mv flash /usr/local/bin/flash`
3. Issue "flash" command to see if it's installed correctly.

0. Install HypriotOS on Raspberry Pi

4. Download & edit HypriotOS configuration file for your Raspberry Pi.

```
$ wget -O hypriot-init.yaml https://mirror.nm.gist.ac.kr/getHypriotConf
```

Let's open the "hypriot.yaml" file and edit its network section.

```
$ sudo vi hypriot.yaml

…
 # static IP configuration:
      interface eth0
      static ip_address=172.29.0.250/24 # Write your Raspberry Pi address
      static routers=172.29.0.254
      static domain_name_servers=8.8.8.8 8.8.4.4
…
```

- The assigned IP address will be automatically applied, when you're initially booting your Raspberry Pi.

0. Install HypriotOS on Raspberry Pi

4. Download & write Hypriot OS image to your MicroSD card.
   - `$ flash -u hypriot-init.yaml -d /dev/sdb -f https://mirror.nm.gist.ac.kr/getHypriot`

Be careful! Wrong –d option may lead to the loss of NUC's OS!

NOTE: This may take some time, even after its gauge reached 100%.

- And, that's it! Now HypriotOS is flashed to your MicroSD card.
- Insert the card back to your Raspberry Pi and boot it up.

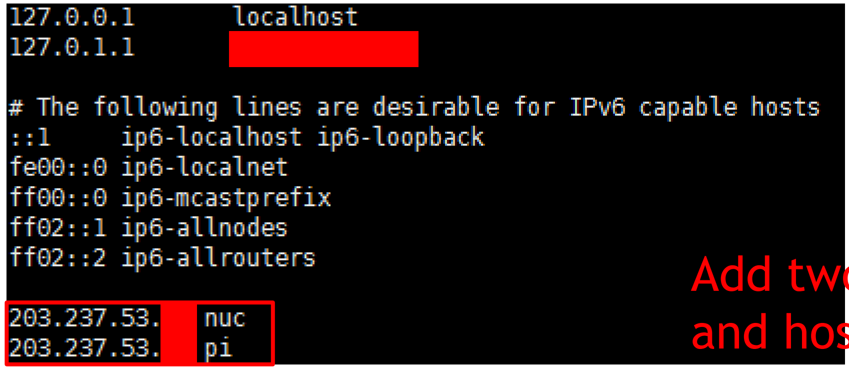
1. Edit /etc/hosts (for NUC)

- Every machine which communicate with themselves must know their own address.

1. Edit /etc/hosts

   $ sudo vi /etc/hosts

(For Example)



```
127.0.0.1       localhost
127.0.1.1       [redacted]

# The following lines are desirable for IPv6 capable hosts
::1     ip6-localhost ip6-loopback
fe00::0 ip6-localnet
ff00::0 ip6-mcastprefix
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters

203.237.53.[redacted]   nuc
203.237.53.[redacted]   pi
```

Add two lines which describe the IP address and hostname of devices

1. Edit /etc/hosts (for Pi)

- Every machine which communicate with themselves must know their own address.

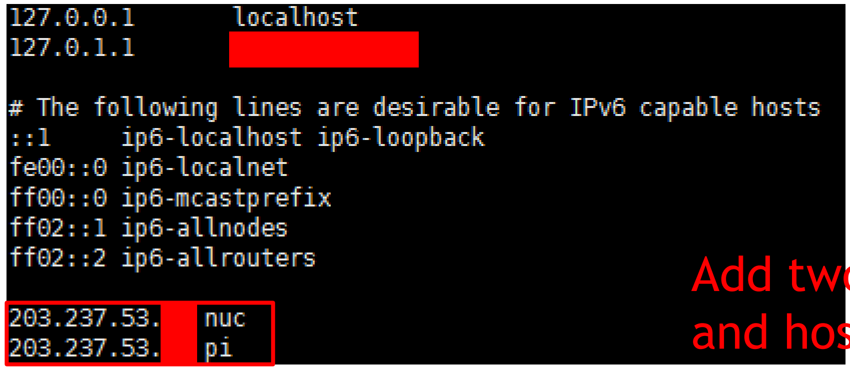
1. SSH into your Pi (ID: pirate, PW: hypriot)

```
$ ssh pirate@<Your Raspberry Pi IP address>
```

2. Edit /etc/hosts

```
$ sudo vi /etc/hosts
```

(For Example)



```
127.0.0.1       localhost
127.0.1.1       [redacted]

# The following lines are desirable for IPv6 capable hosts
::1     ip6-localhost ip6-loopback
fe00::0 ip6-localnet
ff00::0 ip6-mcastprefix
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters

203.237.53.[redacted]     nuc
203.237.53.[redacted]     pi
```

Add two lines which describe the IP address and hostname of devices

1. Edit /etc/hosts

- After editing /etc/hosts, check the edit is done correctly
- For NUC

```
$ ping <Your NUC hostname>
$ ping <Your Raspberry Pi hostname>
```

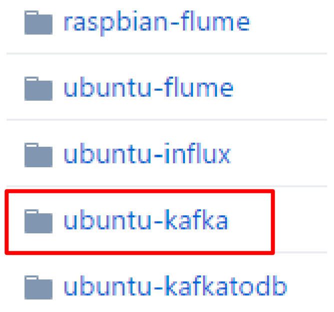
- For Raspberry Pi

```
$ sudo ping <Your NUC hostname>
$ sudo ping <Your Raspberry Pi hostname>
```

2. Download Source from Github

- Download all files from Github
  (http://github.com/SmartXBox/SmartX-mini)
  - $ git clone https://github.com/SmartXBox/SmartX-mini.git

- Folder List



raspbian-flume
ubuntu-flume
ubuntu-influx
ubuntu-kafka
ubuntu-kafkatodb

In this section, we use this

3. Allocate Broker IDs and Ports

1. We'll use a one zookeeper, 3 brokers and one consumer containers which share host's public IP address
2. Zookeeper container doesn't have broker id.
3. Each Broker has a unique id and port to interact each other.
4. Consumer container just used to manage topic and check the data from brokers.

Container Name	IP address	Broker id	Listening port
zookeeper	Host's public IP address	-	2181
broker0		0	9090
broker1		1	9091
broker2		2	9092
consumer		-	-

4. Build Docker Image

- Build Docker Image

1. $cd ~/SmartX-mini/ubuntu-kafka
2. Build Dockerfile ※ It takes long time.

```
$ docker build --tag ubuntu-kafka .
```

- If you want to check Docker instruction words

```
$ docker --help
```

ex) docker ps : List containers

docker start : Start one or more stopped containers

docker rm : Remove one or more containers

5. Run Docker Container

(recommend making new terminal window)

- Run Docker Container

```
$ docker run -it --net=host --name [container name] ubuntu-kafka
```

  - We need to run 5 containers (zookeeper 1, broker 3, consumer 1)
  - Let's assume the name of each containers,

    zookeeper, broker0, broker1, broker2, consumer
  - Repeatedly type the above command with changing container name
  - If you want to look for more details about Docker command, see https://docs.docker.com/reference/commandline/

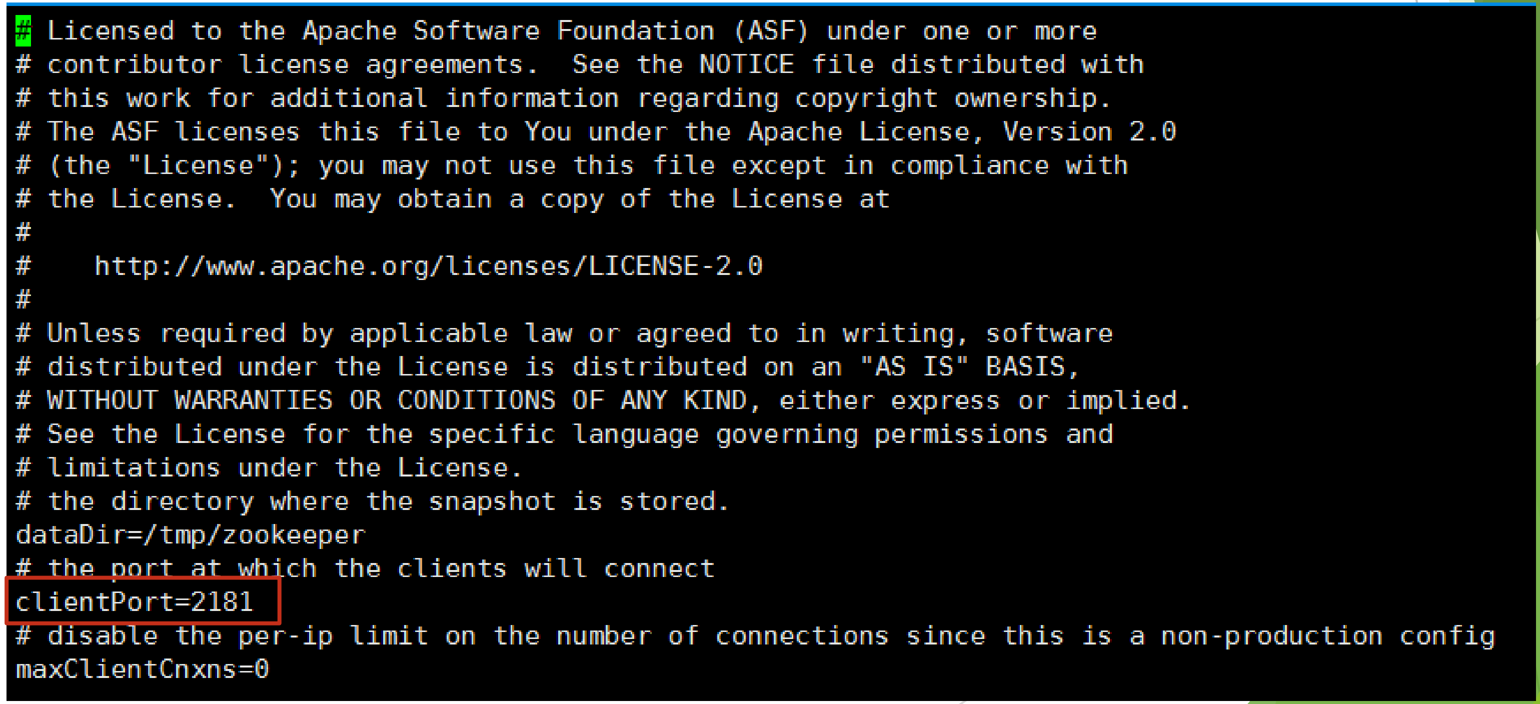
6-1. Configure Zookeeper properties

- Actually we use default configurations

1. Open zookeeper properties file

   ```
   $ vi config/zookeeper.properties
   ```

2. Check the client port



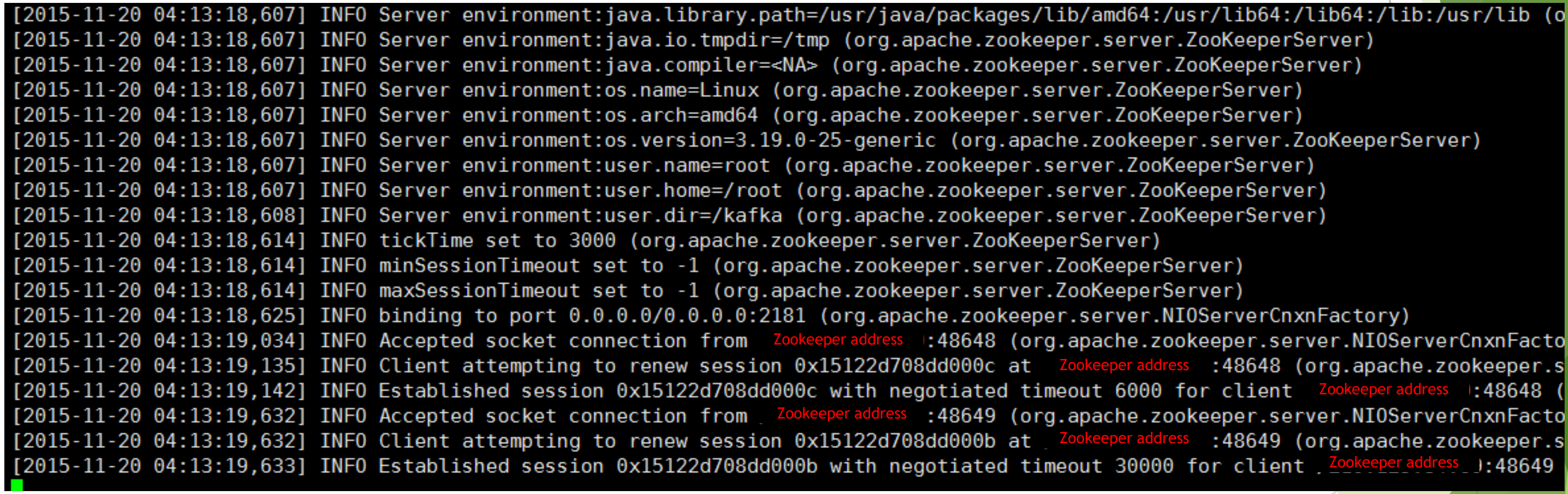
```
# Licensed to the Apache Software Foundation (ASF) under one or more
# contributor license agreements.  See the NOTICE file distributed with
# this work for additional information regarding copyright ownership.
# The ASF licenses this file to You under the Apache License, Version 2.0
# (the "License"); you may not use this file except in compliance with
# the License.  You may obtain a copy of the License at
#
#    http://www.apache.org/licenses/LICENSE-2.0
#
# Unless required by applicable law or agreed to in writing, software
# distributed under the License is distributed on an "AS IS" BASIS,
# WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
# See the License for the specific language governing permissions and
# limitations under the License.
# the directory where the snapshot is stored.
dataDir=/tmp/zookeeper
# the port at which the clients will connect
clientPort=2181
# disable the per-ip limit on the number of connections since this is a non-production config
maxClientCnxns=0
```

6-2. Launching Zookeeper

- ✓ zookeeper must launch first

  `$ bin/zookeeper-server-start.sh config/zookeeper.properties`

  Leave Zookeeper running and open a new terminal for next tasks



```
[2015-11-20 04:13:18,607] INFO Server environment:java.library.path=/usr/java/packages/lib/amd64:/usr/lib64:/lib64:/lib:/usr/lib (o
[2015-11-20 04:13:18,607] INFO Server environment:java.io.tmpdir=/tmp (org.apache.zookeeper.server.ZooKeeperServer)
[2015-11-20 04:13:18,607] INFO Server environment:java.compiler=<NA> (org.apache.zookeeper.server.ZooKeeperServer)
[2015-11-20 04:13:18,607] INFO Server environment:os.name=Linux (org.apache.zookeeper.server.ZooKeeperServer)
[2015-11-20 04:13:18,607] INFO Server environment:os.arch=amd64 (org.apache.zookeeper.server.ZooKeeperServer)
[2015-11-20 04:13:18,607] INFO Server environment:os.version=3.19.0-25-generic (org.apache.zookeeper.server.ZooKeeperServer)
[2015-11-20 04:13:18,607] INFO Server environment:user.name=root (org.apache.zookeeper.server.ZooKeeperServer)
[2015-11-20 04:13:18,607] INFO Server environment:user.home=/root (org.apache.zookeeper.server.ZooKeeperServer)
[2015-11-20 04:13:18,608] INFO Server environment:user.dir=/kafka (org.apache.zookeeper.server.ZooKeeperServer)
[2015-11-20 04:13:18,614] INFO tickTime set to 3000 (org.apache.zookeeper.server.ZooKeeperServer)
[2015-11-20 04:13:18,614] INFO minSessionTimeout set to -1 (org.apache.zookeeper.server.ZooKeeperServer)
[2015-11-20 04:13:18,614] INFO maxSessionTimeout set to -1 (org.apache.zookeeper.server.ZooKeeperServer)
[2015-11-20 04:13:18,625] INFO binding to port 0.0.0.0/0.0.0.0:2181 (org.apache.zookeeper.server.NIOServerCnxnFactory)
[2015-11-20 04:13:19,034] INFO Accepted socket connection from  Zookeeper address :48648 (org.apache.zookeeper.server.NIOServerCnxnFacto
[2015-11-20 04:13:19,135] INFO Client attempting to renew session 0x15122d708dd000c at  Zookeeper address :48648 (org.apache.zookeeper.s
[2015-11-20 04:13:19,142] INFO Established session 0x15122d708dd000c with negotiated timeout 6000 for client  Zookeeper address :48648 (
[2015-11-20 04:13:19,632] INFO Accepted socket connection from  Zookeeper address :48649 (org.apache.zookeeper.server.NIOServerCnxnFacto
[2015-11-20 04:13:19,632] INFO Client attempting to renew session 0x15122d708dd000b at  Zookeeper address :48649 (org.apache.zookeeper.s
[2015-11-20 04:13:19,633] INFO Established session 0x15122d708dd000b with negotiated timeout 30000 for client  Zookeeper address :48649
```

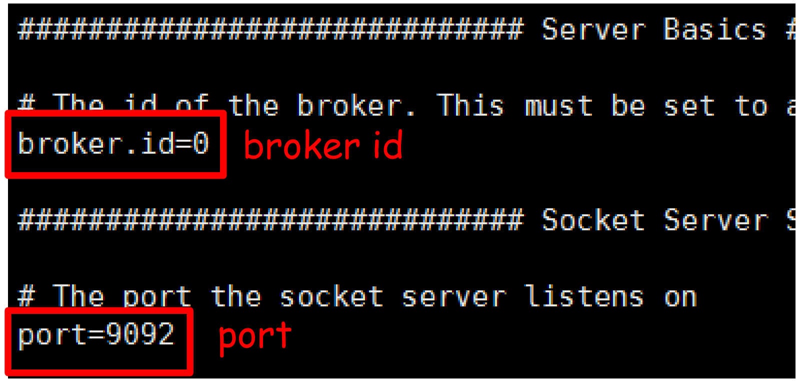
7-1. Configure Kafka properties

1. Create a Kafka container with the docker command before

   ```
   docker run -it --net=host --name [container name] ubuntu-kafka
   ```

2. Open server properties file and change proper broker id and port (they must be unique to each other) (Only for broker0,1,2)

   ```
   $ vi config/server.properties
   ```



```
############################# Server Basics #

# The id of the broker. This must be set to a
broker.id=0

############################# Socket Server S

# The port the socket server listens on
port=9092
```

Container Name	Broker id	Listening port
broker0	0	9090
broker1	1	9091
broker2	2	9092
consumer	-	-

Consumer container will not run any brokers

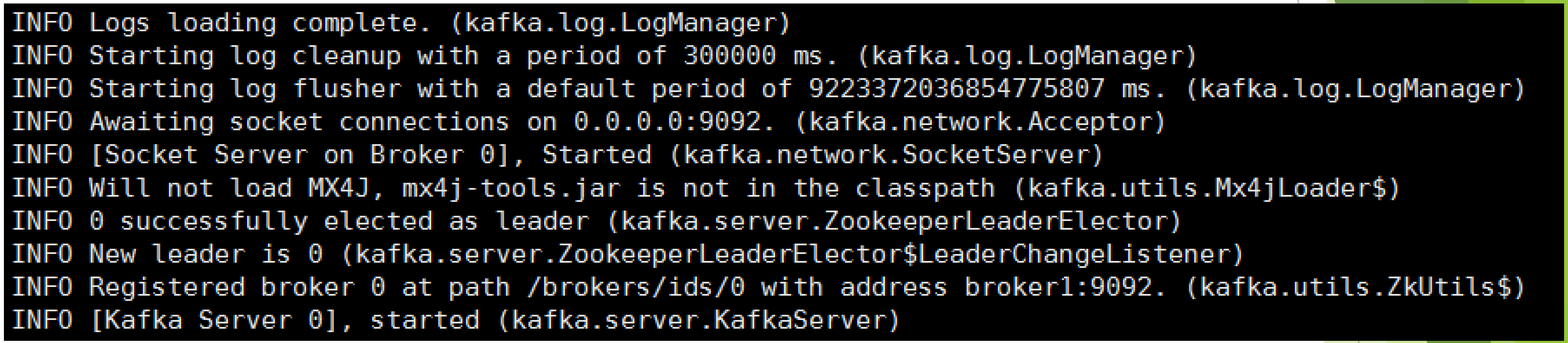
3. Launch Kafka brokers (Only for broker0,1,2)

   ```
   $ bin/kafka-server-start.sh config/server.properties
   ```

4. Repeat previous steps for broker0, broker1, broker2,consumer

7-2. Launching Kafka brokers

✓ When it successfully works, each broker containers will show messages like the below



```
INFO Logs loading complete. (kafka.log.LogManager)
INFO Starting log cleanup with a period of 300000 ms. (kafka.log.LogManager)
INFO Starting log flusher with a default period of 9223372036854775807 ms. (kafka.log.LogManager)
INFO Awaiting socket connections on 0.0.0.0:9092. (kafka.network.Acceptor)
INFO [Socket Server on Broker 0], Started (kafka.network.SocketServer)
INFO Will not load MX4J, mx4j-tools.jar is not in the classpath (kafka.utils.Mx4jLoader$)
INFO 0 successfully elected as leader (kafka.server.ZookeeperLeaderElector)
INFO New leader is 0 (kafka.server.ZookeeperLeaderElector$LeaderChangeListener)
INFO Registered broker 0 at path /brokers/ids/0 with address broker1:9092. (kafka.utils.ZkUtils$)
INFO [Kafka Server 0], started (kafka.server.KafkaServer)
```

8. Make a topic

- Create topic
  - `$ bin/kafka-topics.sh --create --zookeeper localhost:2181 --replication-factor 1 --partitions 3 --topic resource`

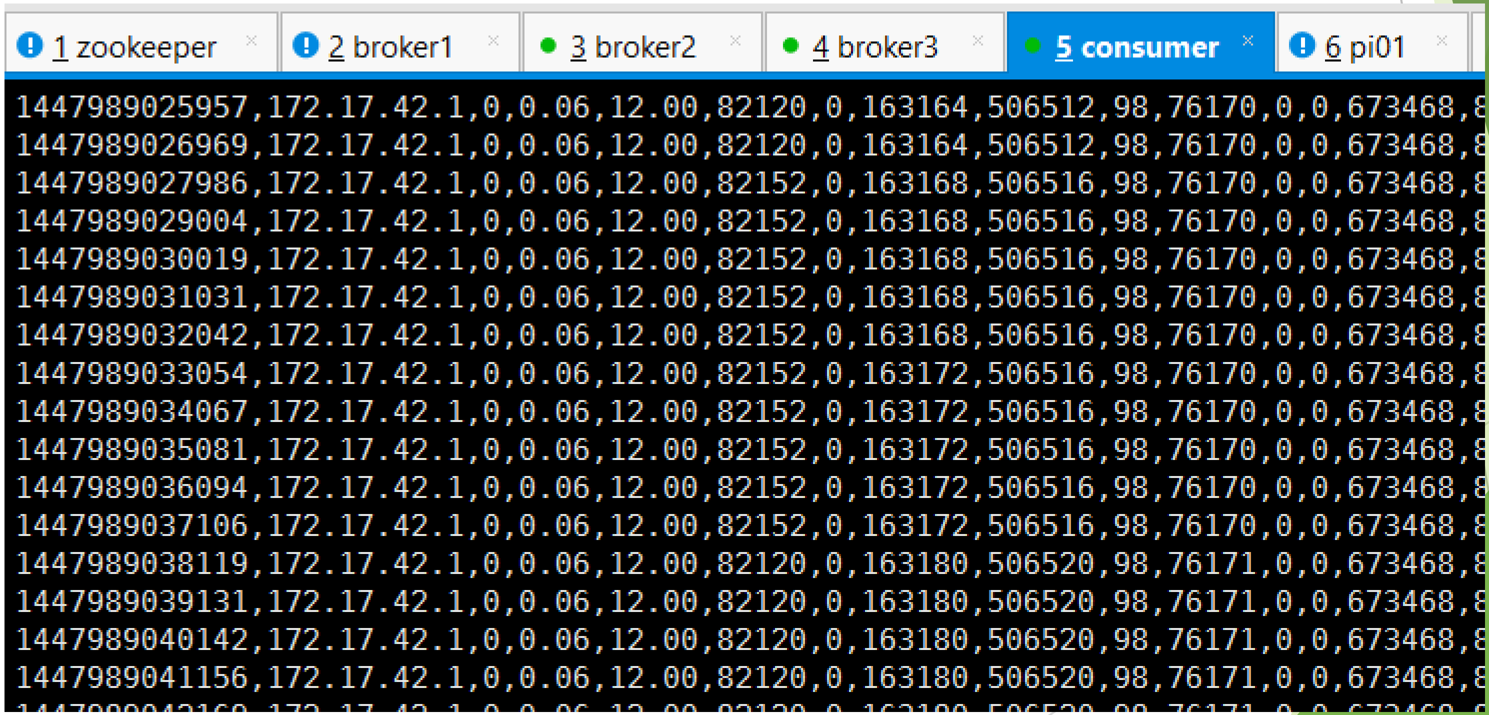
- We can check topics.

  Topic List
  - `$ bin/kafka-topics.sh --list --zookeeper localhost:2181`

  Topic specification
  - `$ bin/kafka-topics.sh --describe --zookeeper localhost:2181 --topic resource`

9. Consume message from brokers

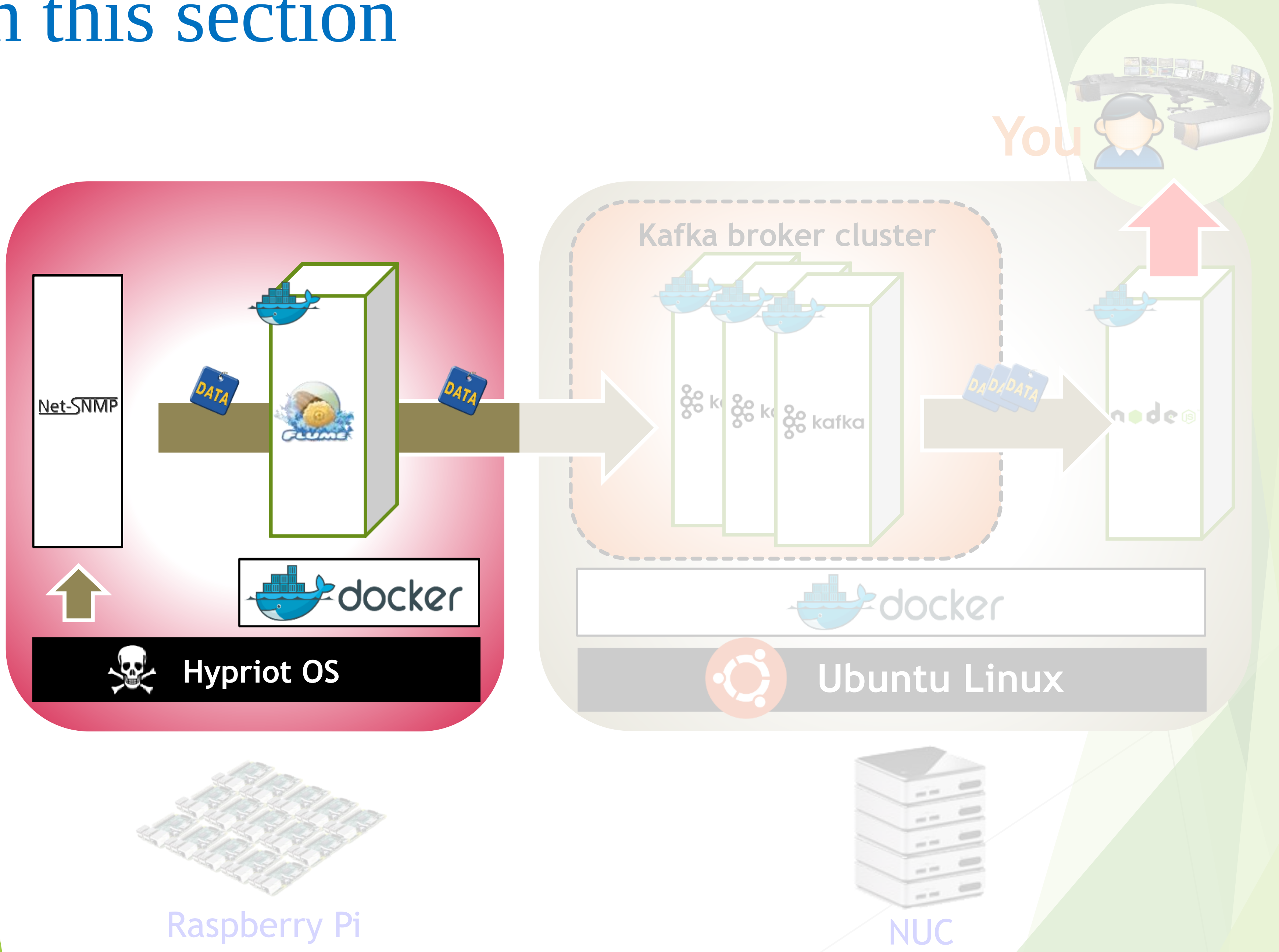
1. Launch consumer script on the **Consumer container**
   - `$ bin/kafka-console-consumer.sh --zookeeper localhost:2181 --topic resource --from-beginning`



```
1447989025957,172.17.42.1,0,0.06,12.00,82120,0,163164,506512,98,76170,0,0,673468,
1447989026969,172.17.42.1,0,0.06,12.00,82120,0,163164,506512,98,76170,0,0,673468,
1447989027986,172.17.42.1,0,0.06,12.00,82152,0,163168,506516,98,76170,0,0,673468,
1447989029004,172.17.42.1,0,0.06,12.00,82152,0,163168,506516,98,76170,0,0,673468,
1447989030019,172.17.42.1,0,0.06,12.00,82152,0,163168,506516,98,76170,0,0,673468,
1447989031031,172.17.42.1,0,0.06,12.00,82152,0,163168,506516,98,76170,0,0,673468,
1447989032042,172.17.42.1,0,0.06,12.00,82152,0,163168,506516,98,76170,0,0,673468,
1447989033054,172.17.42.1,0,0.06,12.00,82152,0,163172,506516,98,76170,0,0,673468,
1447989034067,172.17.42.1,0,0.06,12.00,82152,0,163172,506516,98,76170,0,0,673468,
1447989035081,172.17.42.1,0,0.06,12.00,82152,0,163172,506516,98,76170,0,0,673468,
1447989036094,172.17.42.1,0,0.06,12.00,82152,0,163172,506516,98,76170,0,0,673468,
1447989037106,172.17.42.1,0,0.06,12.00,82152,0,163172,506516,98,76170,0,0,673468,
1447989038119,172.17.42.1,0,0.06,12.00,82120,0,163180,506520,98,76171,0,0,673468,
1447989039131,172.17.42.1,0,0.06,12.00,82120,0,163180,506520,98,76171,0,0,673468,
1447989040142,172.17.42.1,0,0.06,12.00,82120,0,163180,506520,98,76171,0,0,673468,
1447989041156,172.17.42.1,0,0.06,12.00,82120,0,163180,506520,98,76171,0,0,673468,
```

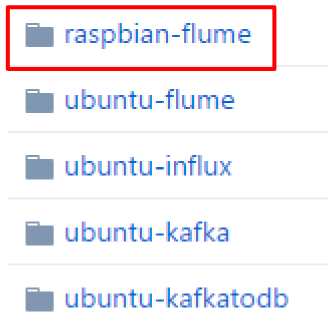
Connecting Configuration on Raspberry Pi

In this section



1. Download Source from Github

- Git package is already installed in Hypriot OS
- Download all files from Github
  (http://github.com/SmartXBox/SmartX-mini)
  - $ git clone https://github.com/SmartXBox/SmartX-mini.git
- Folder List



raspbian-flume
ubuntu-flume
ubuntu-influx
ubuntu-kafka
ubuntu-kafkatodb

In this section, we use this

2. Install Net-SNMP

- Update packages

```
$ sudo apt-get update
```

- Download Net-SNMP

```
$ apt-get install  - y snmp snmpd snmp-mibs-downloader
```

- Download MIBs

```
$ sudo download-mibs
```

- Modify configuration file

```
$ sudo vi /etc/snmp/snmpd.conf
    #rocommunity public localhost -> Delete #
$ sudo systemctl restart snmpd.service
```

3. Install Flume on RPi

1) Build Dockerfile ※ It takes long time.

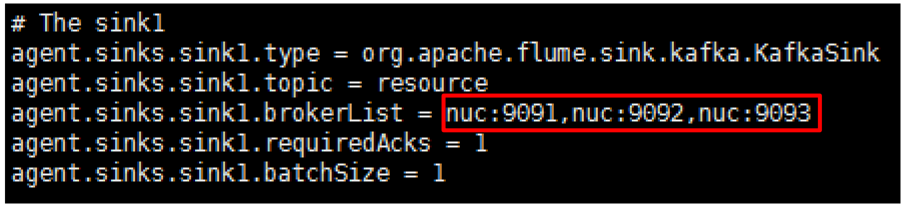
```
$ cd SmartX-mini/raspbian-flume
$ docker build --tag raspbian-flume .
$ docker run -it --net=host --name flume raspbian-flume
```

2) Check the configuration file

```
$ vi conf/flume-conf.properties
```

3) Modifying broker list

- Default value sets "nuc"
- Edit them into your own nuc's hostname

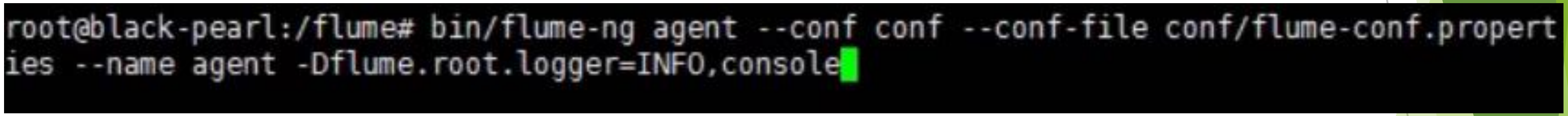


```
# The sink1
agent.sinks.sink1.type = org.apache.flume.sink.kafka.KafkaSink
agent.sinks.sink1.topic = resource
agent.sinks.sink1.brokerList = nuc:9091,nuc:9092,nuc:9093
agent.sinks.sink1.requiredAcks = 1
agent.sinks.sink1.batchSize = 1
```

4. Run Flume Agent

- Run Flume on RPi

```
$ bin/flume-ng agent --conf conf --conf-file conf/flume-
conf.properties --name agent -Dflume.root.logger=INFO,console
```

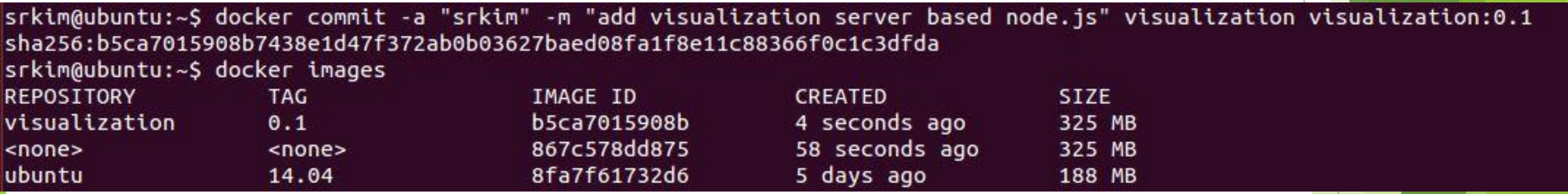


```
root@black-pearl:/flume# bin/flume-ng agent --conf conf --conf-file conf/flume-conf.propert
ies --name agent -Dflume.root.logger=INFO,console
```

Thank You for Your Attention
Any Questions?

(참고) Container 변경사항 저장 및 재시작

- Commit Container
  - 컨테이너 내의 변경사항을 반영하여 새로운 컨테이너 이미지 작성
  - Ctrl+P+Q
  - docker commit -a “[username]” -m "add visualization server based node.js" visualization visualization:0.1



```
srkim@ubuntu:~$ docker commit -a "srkim" -m "add visualization server based node.js" visualization visualization:0.1
sha256:b5ca7015908b7438e1d47f372ab0b03627baed08fa1f8e11c88366f0c1c3dfda
srkim@ubuntu:~$ docker images
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
visualization       0.1                 b5ca7015908b        4 seconds ago       325 MB
<none>              <none>              867c578dd875        58 seconds ago      325 MB
ubuntu              14.04               8fa7f61732d6        5 days ago          188 MB
```

- Restart Container
  - Stop했던 컨테이너를 Restart하면 이전 작업 내용을 유지한 채로 다시 컨테이너를 시작할 수 있다.
  - docker stop visualization
  - docker restart visualization