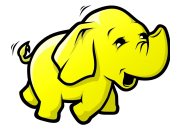




k8s : h4p



RATIS 

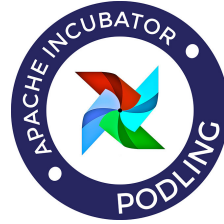


RATIS



RAFT library for Java

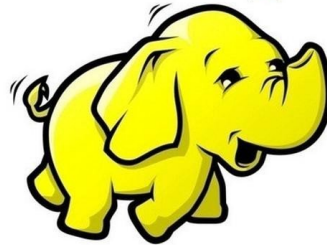
RATIS
RAFT library for Java





RATIS
RAFT library for Java

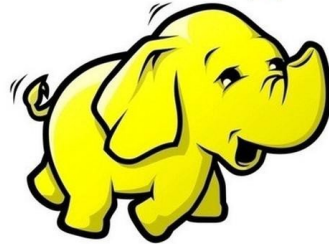


hadoop

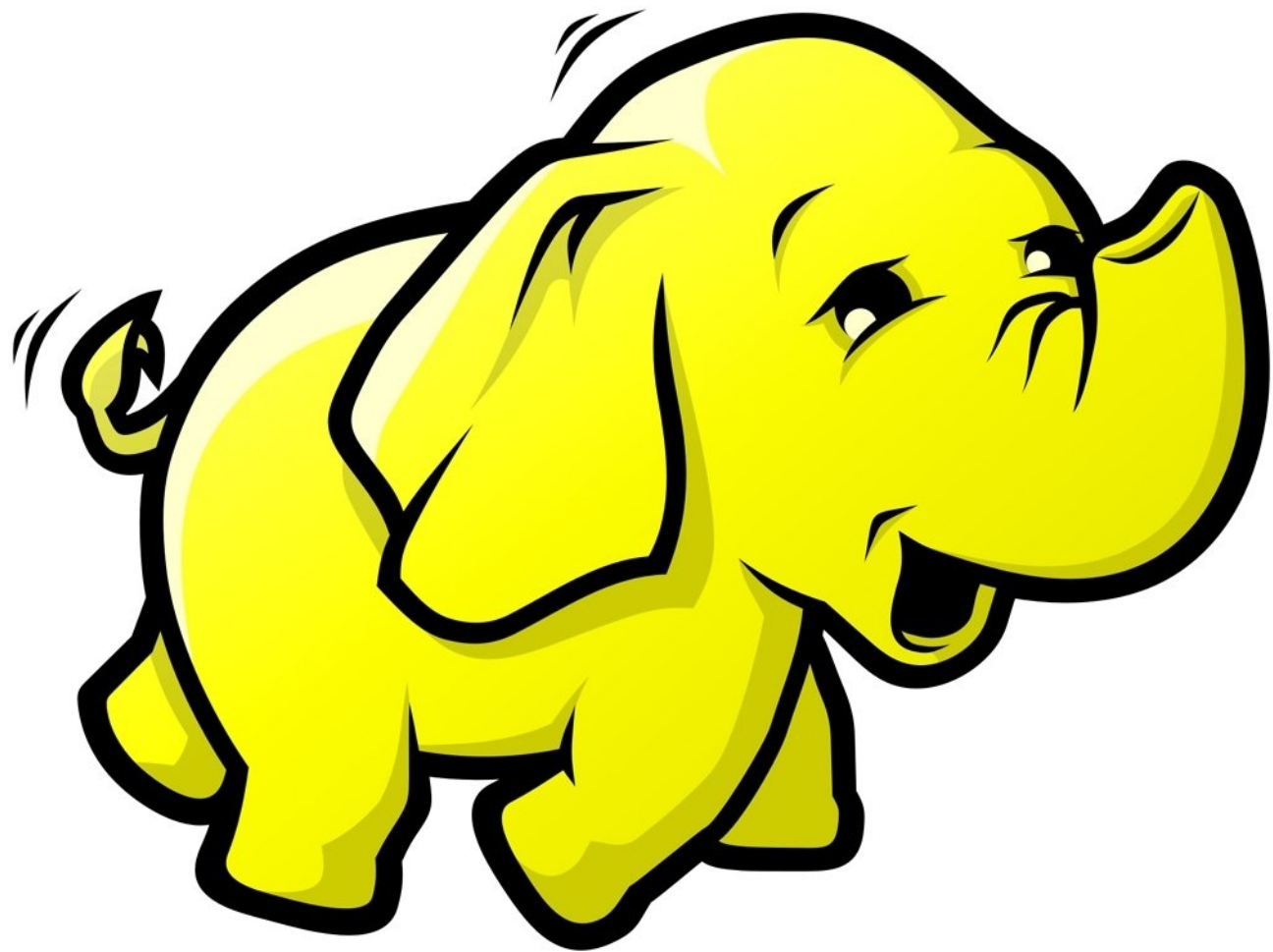


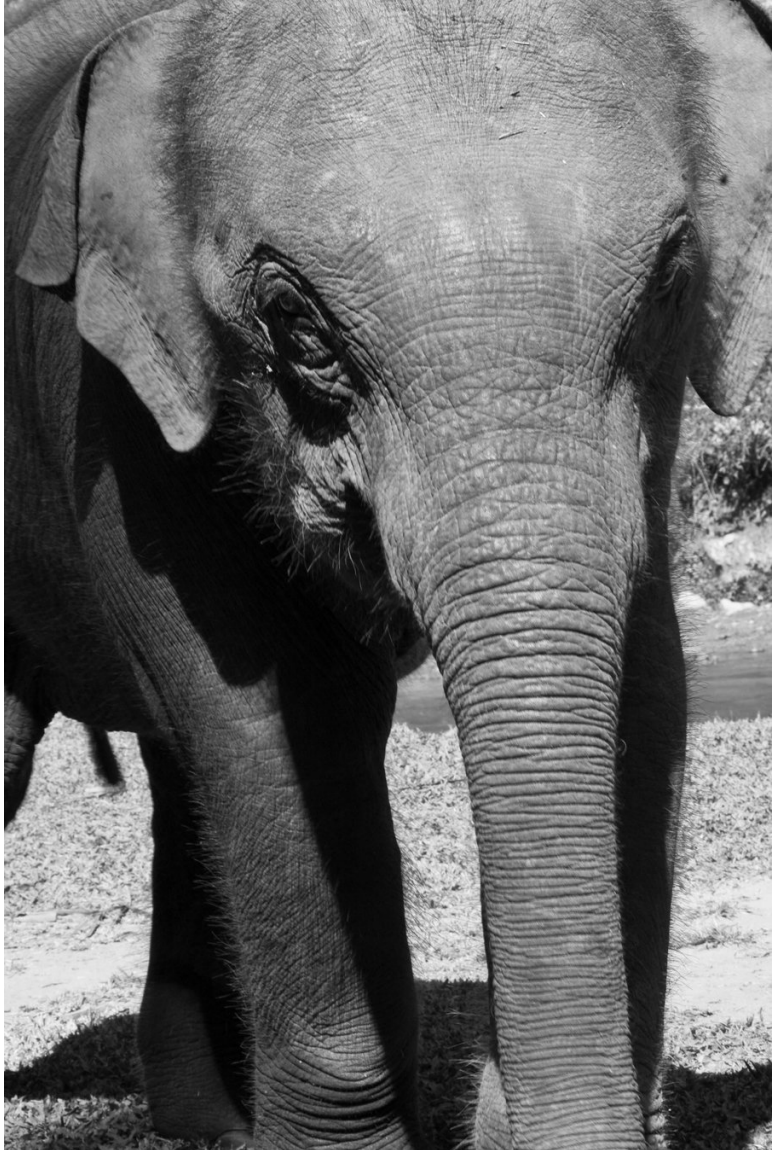
RATIS  
RAFT library for Java

hadoop



<https://flokkr.github.io>  







Energy

Fridge-Freezer

Manufacturer
Model

More efficient



A

Less efficient

Energy consumption kWh/year
(Based on standard test results for 24h)

325

Actual consumption will depend on how the appliance is used and where it is located

Fresh food volume l
Frozen food volume l

190
126

Noise

(dB(A) re 1 pW)

Further information is contained in product brochures

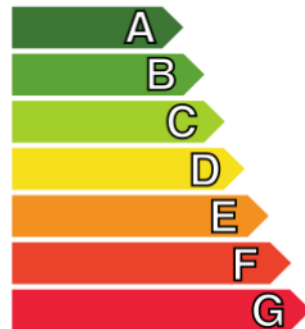


Norm EN 153 May 1990
Refrigerator Label Directive 94/2/EC

Docker

Local compose file

More efficient



Less efficient

Configuration management

Source

Preprocessing

On change

Provisioning, Scheduling

Multihost support

Scheduling

Cluster definition

Scaling

Multi tenancy

Fallover

Network

Intraservice network

DNS

Service discovery

Data locality

Availability of the ports



Less efficient

Configuration management

Source

Preprocessing

On change

Provisioning, Scheduling

Multihost support

Scheduling

Cluster definition

Scaling

Multi tenancy

Failover

Network

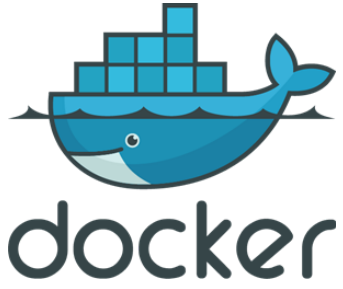
Intraservice network

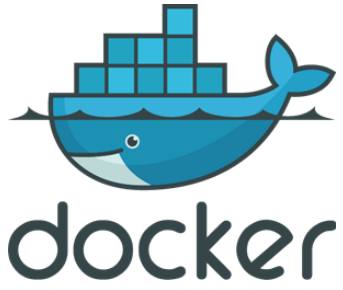
DNS

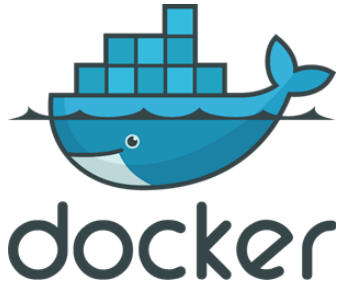
Service discovery

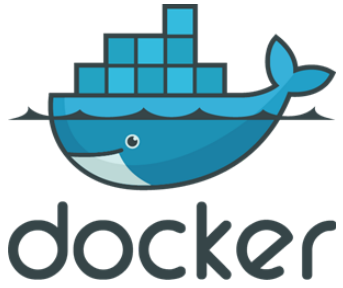
Data locality

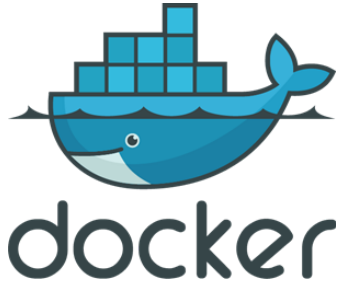
Availability of the ports

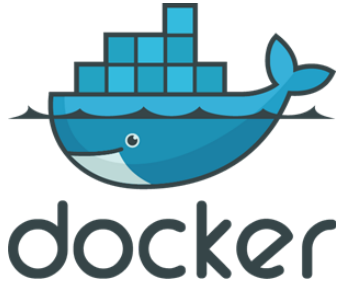


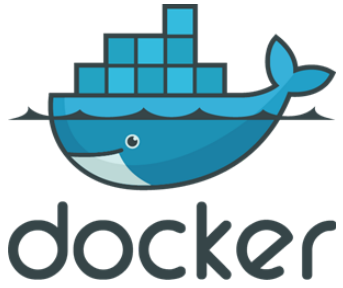












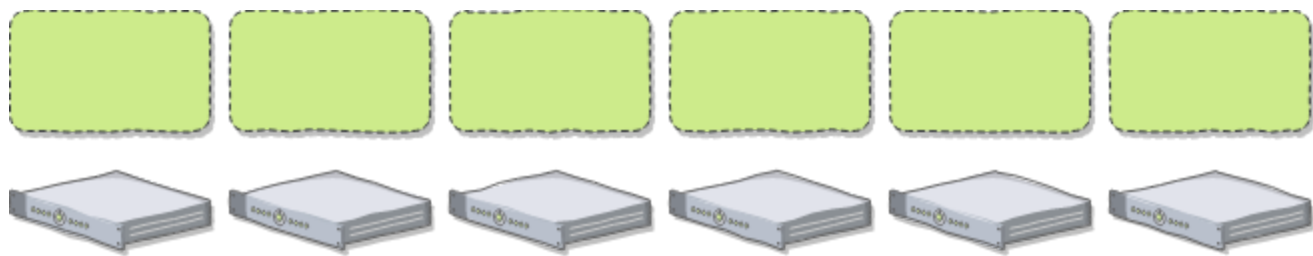
What is Apache Hadoop

What is Apache Hadoop

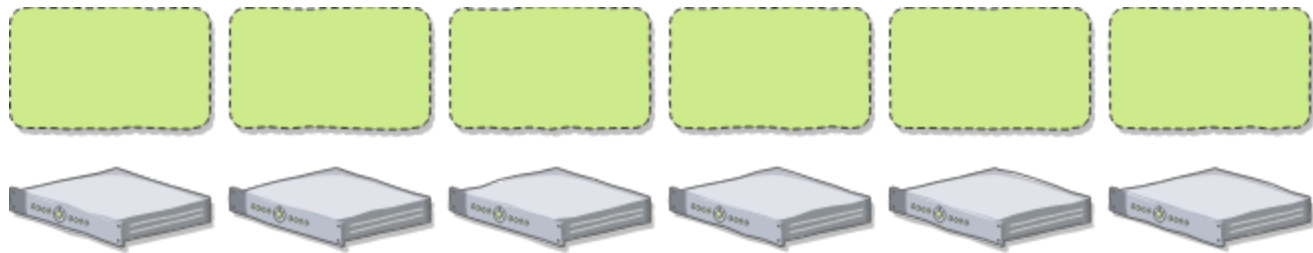
in 60 seconds

Data





HDFS



HDFS



YARN



HDFS



YARN



Mapreduce

HDFS

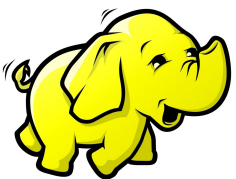
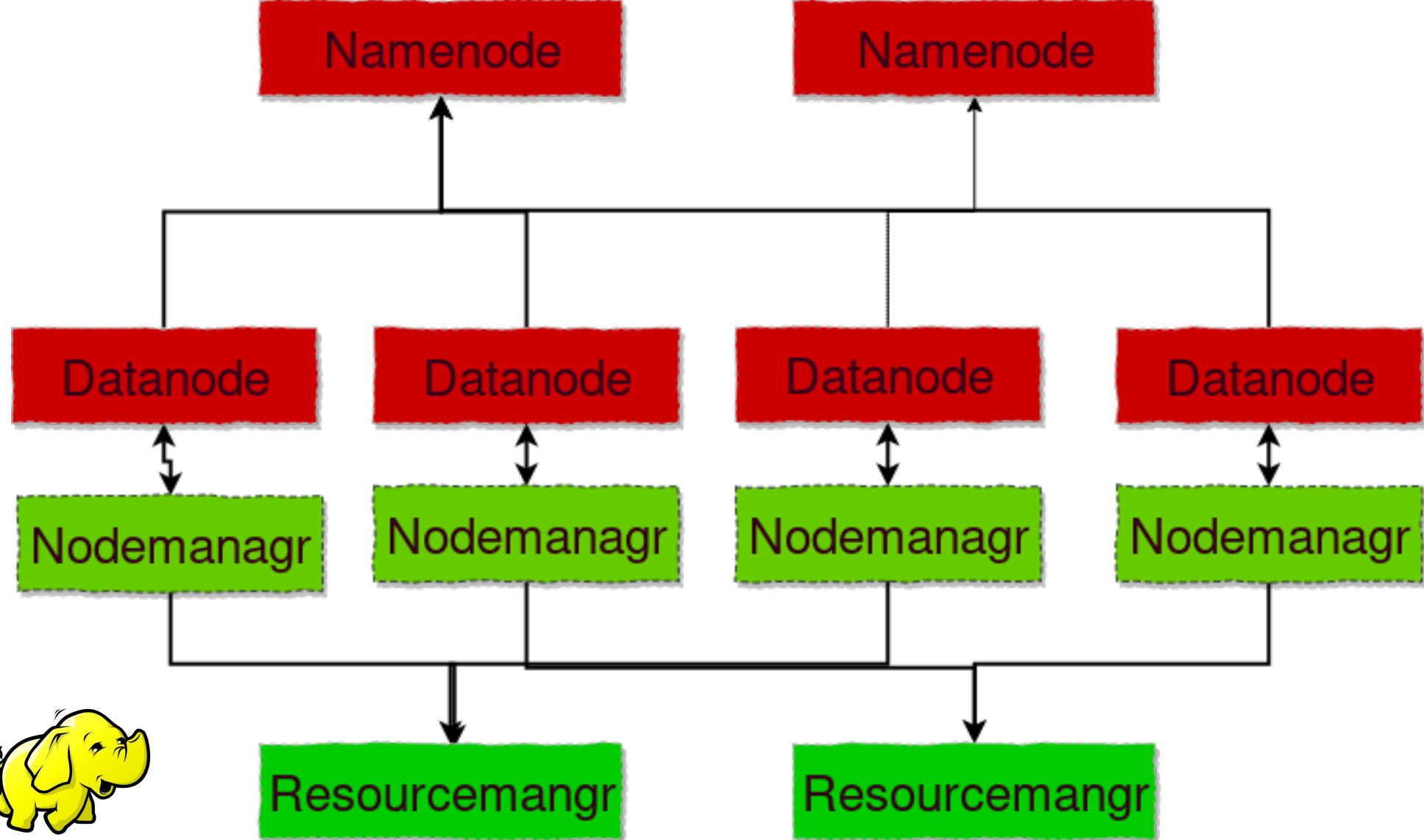


YARN



Mapreduce

Ozone + HDDS



Dockerfile

```
FROM frolvlad/alpine-oraclejdk8  
ADD hadoop-3.2.0.tar.gz /opt  
WORKDIR /opt/hadoop
```



Less efficient

Configuration management

Source

Preprocessing

On change

Provisioning, Scheduling

Multihost support

Scheduling

Cluster definition

Scaling

Multi tenancy

Failover

Network


```
<configuration>
  <property>
    <name>dfs.namenode.rpc-address</name>
    <value>namenode:9000</value>
  </property>
  <property>
    <name>dfs.datanode.plugins</name>
    <value>org.apache.hadoop.ozone.HddsDatanodeService</value>
  </property>
  <property>
    <name>rpc.metrics.percentiles.intervals</name>
    <value>60,300</value>
  </property>
  <property>
    <name>dfs.namenode.name.dir</name>
    <value>/data/namenode</value>
  </property>
  <property>
    <name>rpc.metrics.quantile.enable</name>
    <value>true</value>
  </property>
</configuration>
```

```
version: "3"
services:
  service1:
    image: apache/imagename
    hostname: namenode
    ports:
      - 9870:9870
    environment:
      CONFIGURATION1: value
      DFS_DIR: /dfs
      THREAD_NUMBER: 1
```

How to handle configuration?

Create a simple **launcher** script to

- Create config file from environment variables
- Start the application

```
version: "3"
```

```
services:
```

```
  namenode:
```

```
    image: flokkkr/hadoop
```

```
    hostname: namenode
```

```
    command: ["hdfs", "namenode"]
```

```
    ports:
```

```
      - 9870:9870
```

```
    environment:
```

```
      ENSURE_NAMENODE_DIR: "/tmp/hadoop-root/dfs/name"
```

```
      CORE-SITE.XML_fs.defaultFS: "hdfs://namenode:9000"
```

```
      HDFS-SITE.XML_dfs.namenode.rpc-address: "namenode:9000"
```

```
      HDFS-SITE.XML_dfs.replication: "1"
```

```
  datanode:
```

```
    image: flokkkr/hadoop
```

```
    command: ["hdfs", "datanode"]
```

```
    environment:
```

```
      CORE-SITE.XML_fs.defaultFS: "hdfs://namenode:9000"
```

```
      HDFS-SITE.XML_dfs.namenode.rpc-address: "namenode:9000"
```

```
      HDFS-SITE.XML_dfs.replication: "1"
```

```
      LOG4J.PROPERTIES_log4j.rootLogger: "INFO, stdout"
```

```
      LOG4J.PROPERTIES_log4j.appender.stdout: "org.apache.log4j.ConsoleAppender"
```

```
      LOG4J.PROPERTIES_log4j.appender.stdout.layout: "org.apache.log4j.PatternLayout"
```

```
      LOG4J.PROPERTIES_log4j.appender.stdout.layout.ConversionPattern: "%d{yyyy-MM-dd HH:mm:ss} %-5p
```

**Configuration management is
managing the configuration values**



Less efficient

Configuration management

Source

Preprocessing

On change

ENV (script)

n/a

n/a

Provisioning, Scheduling

Multihost support

Scheduling

Cluster definition

Scaling

Multi tenancy

Failover

Network

Intraservice network

DNS

Service discovery

Data locality

`/usr/bin/hadoop`

`/usr/bin/hive`

`/usr/bin/spark`

`/usr/bin/zeppelin`

`flokkr/all-in-one`

/usr/bin/hadoop

/usr/bin/hive

/usr/bin/spark

/usr/bin/zeppelin

flokkr/all-in-one

/usr/bin/hadoop

flokkr/hadoop

/usr/bin/hive

flokkr/hive

/usr/bin/spark

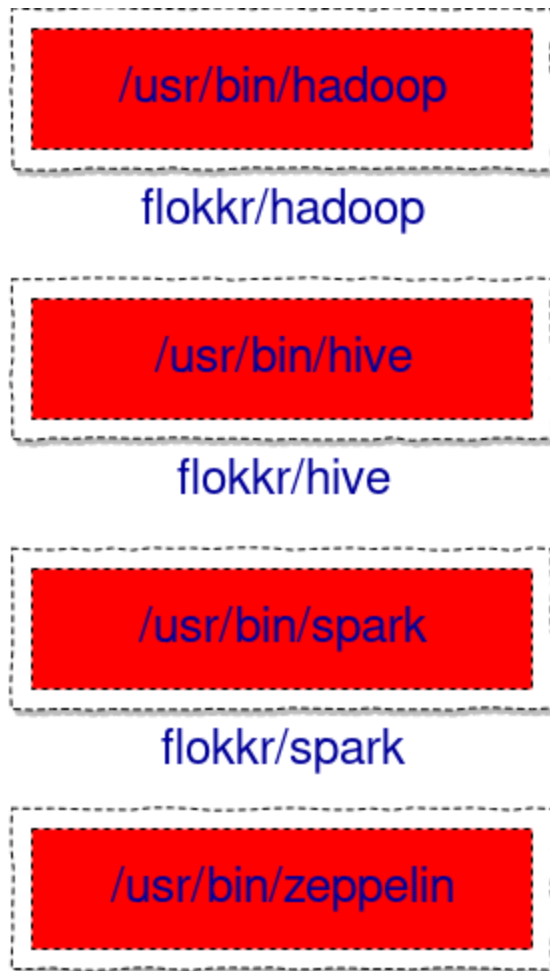
flokkr/spark

/usr/bin/zeppelin

flokkr/zeppelin



flokkr/all-in-one



flokkr/hadoop

/usr/bin/hive

flokkr/hive

/usr/bin/spark

flokkr/spark

/usr/bin/zeppelin

flokkr/zeppelin

/usr/bin/hadoop

/usr/bin/hive

/usr/bin/spark

/usr/bin/zeppelin

flokkr/all-in-one

/usr/bin/hadoop

flokkr/hadoop

/usr/bin/hive

flokkr/hive

/usr/bin/spark

flokkr/spark

/usr/bin/zeppelin

flokkr/zeppelin



Container is the **unit** of packaging.

Launcher script has the power

Launcher script

Launcher script

- Create config files from ENV

Launcher script

- Create config files from ENV
- Wait for the dependency (TCP check)

Launcher script

- Create config files from ENV
- Wait for the dependency (TCP check)
- Download additional optional component

Launcher script

- Create config files from ENV
- Wait for the dependency (TCP check)
- Download additional optional component
- Prepare HDFS (format namenode, ...)

Launcher script

- Create config files from ENV
- Wait for the dependency (TCP check)
- Download additional optional component
- Prepare HDFS (format namenode, ...)
- Retrieve kerberos/SSL secrets

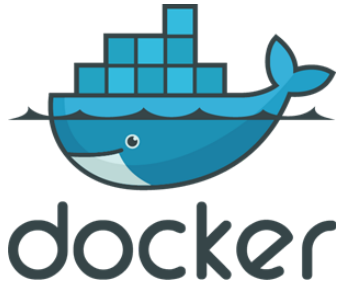
Launcher script

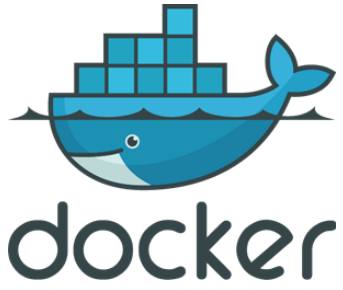
- Create config files from ENV
- Wait for the dependency (TCP check)
- Download additional optional component
- Prepare HDFS (format namenode, ...)
- Retrieve kerberos/SSL secrets
- Enable prometheus monitoring (Java agent)

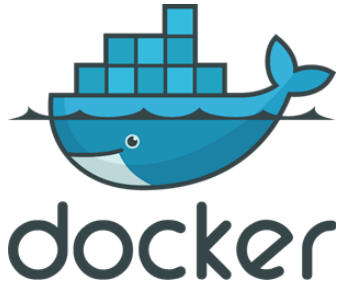
Launcher script

- Create config files from ENV
- Wait for the dependency (TCP check)
- Download additional optional component
- Prepare HDFS (format namenode, ...)
- Retrieve kerberos/SSL secrets
- Enable prometheus monitoring (Java agent)
- Show network traffic (Instrumentation with Java agent)

```
datanode_1 | =====
datanode_1 | *** Launching "hdfs datanode"
namenode_1 | Formatting using clusterid: CID-51b1d36b-356a-4c2a-9570-57b6300ccd5d
namenode_1 | ===== Plugin is activated BYTEMAN =====
namenode_1 | Connecting to kv.anzix.net (176.9.127.13:443)
namenode_1 | byteman.jar          14% |****                               |   112k  0:00:06 ETA
namenode_1 | byteman.jar          100% |*****                             |   791k  0:00:00 ETA
namenode_1 |
namenode_1 | Connecting to gist.githubusercontent.com (151.101.36.133:443)
namenode_1 | byteman.btm          100% |*****                             |    710  0:00:00 ETA
namenode_1 |
namenode_1 | Process is instrumented with setting JAVA_OPTS to -javaagent:/opt/byteman/byteman.jar=script:/tmp/byteman.btm
namenode_1 | Standard output is replaced with btrace output
namenode_1 | =====
namenode_1 | *** Launching "hdfs namenode"
namenode_1 | --> RPC message request: VersionRequestProto from 172.23.0.3:52480
namenode_1 |
namenode_1 | --> RPC message response: VersionRequestProto to 172.23.0.3:52480
namenode_1 | info {
namenode_1 |   buildVersion: "16b70619a24cdcf5d3b0fcf4b58ca77238ccbe6d"
namenode_1 |   unused: 0
namenode_1 |   blockPoolID: "BP-1294221783-172.23.0.2-1528791611084"
namenode_1 |   storageInfo {
namenode_1 |     layoutVersion: 4294967232
namenode_1 |     namespaceID: 2129242657
namenode_1 |     clusterID: "CID-51b1d36b-356a-4c2a-9570-57b6300ccd5d"
namenode_1 |     cTime: 1528791611084
namenode_1 |   }
namenode_1 |   softwareVersion: "3.1.0"
namenode_1 |   capabilities: 1
namenode_1 |   state: ACTIVE
namenode_1 | }
namenode_1 |
namenode_1 | --> RPC message request: RegisterDatanodeRequestProto from 172.23.0.3:52480
namenode_1 | registration {
namenode_1 |   datanodeID {
namenode_1 |     ipAddr: "0.0.0.0"
namenode_1 |     hostName: "8d6011399538"
namenode_1 |     datanodeUuid: "92926bb4-04b5-4e9c-8f85-694a2d7c61ec"
namenode_1 |     xferPort: 9866
namenode_1 |     infoPort: 9864
namenode_1 |     ipcPort: 9867
namenode_1 |     infoSecurePort: 0
```





Hashicorp stack

"do it yourself"



Service Discovery and Configuration Made Easy



Service Discovery and Configuration Made Easy



HashiCorp
Vault

A Tool for Managing Secrets



Service Discovery and Configuration Made Easy



HashiCorp
Vault

A Tool for Managing Secrets

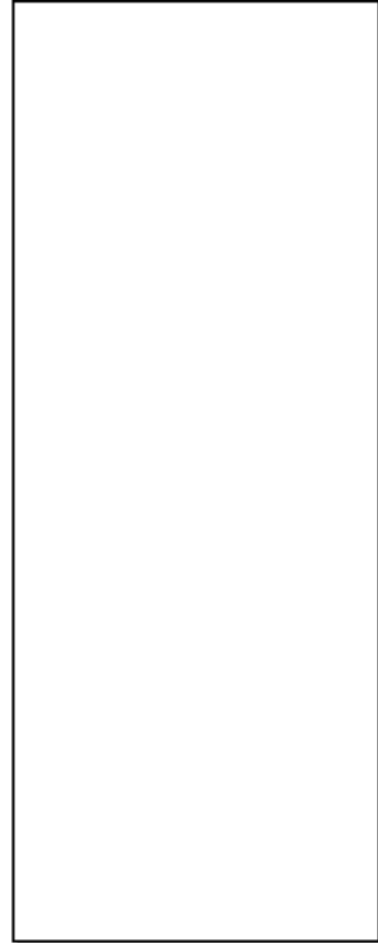


HashiCorp
Nomad

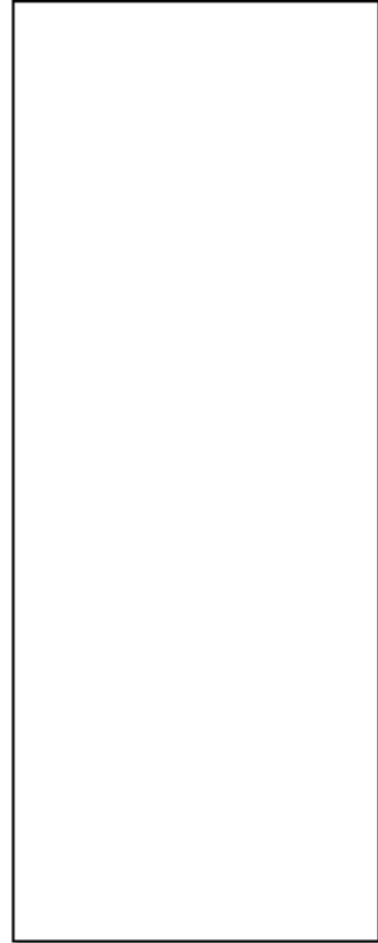
Easily Deploy Applications at Any Scale



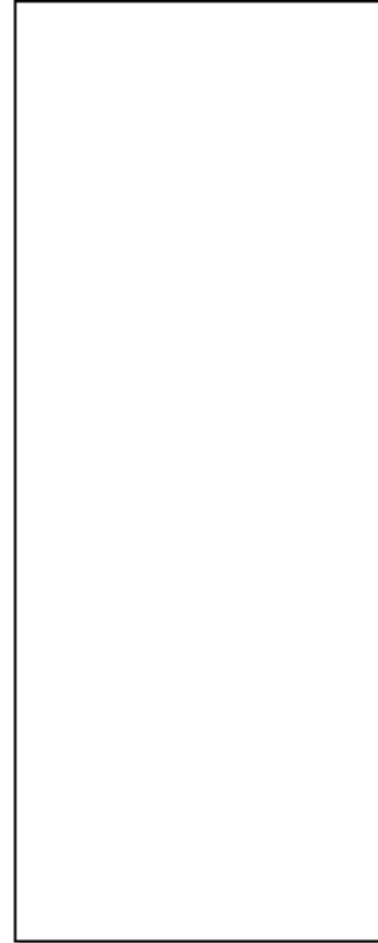
node1



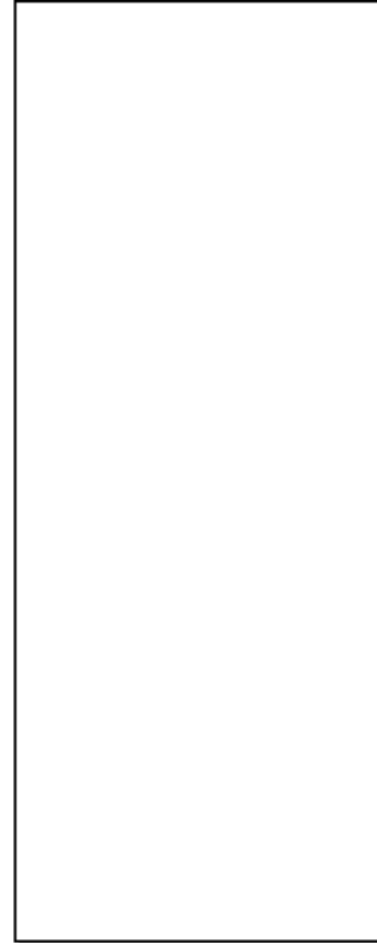
node2



node3



node4



node5

The diagram consists of five vertical rectangular boxes arranged horizontally. Each box is divided into two sections by a horizontal dashed line. The bottom section of each box contains the word 'nomad' in red text. Below each box is a label: 'node1', 'node2', 'node3', 'node4', and 'node5' respectively. The top section of each box is empty.

nomad

node1

nomad

node2

nomad

node3

nomad

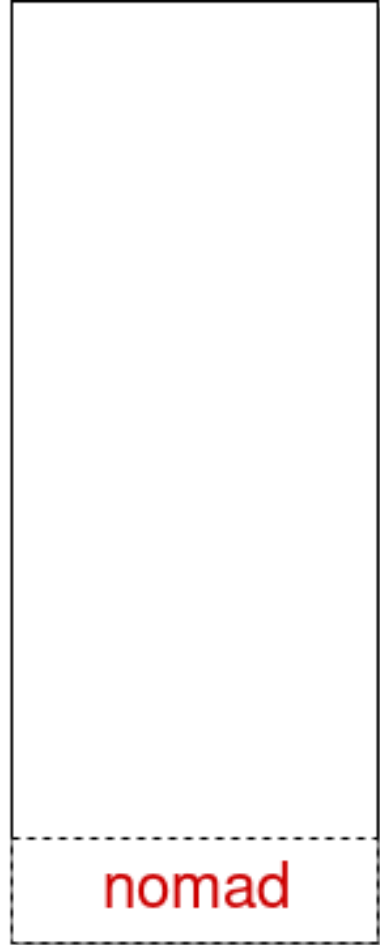
node4

nomad

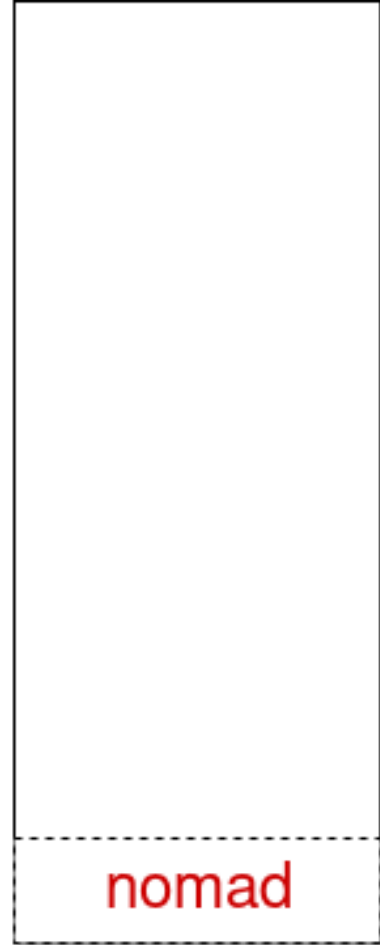
node5



node1



node2



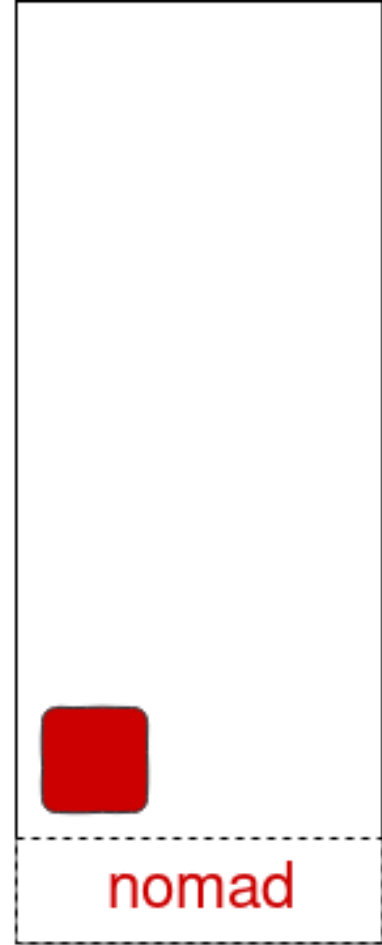
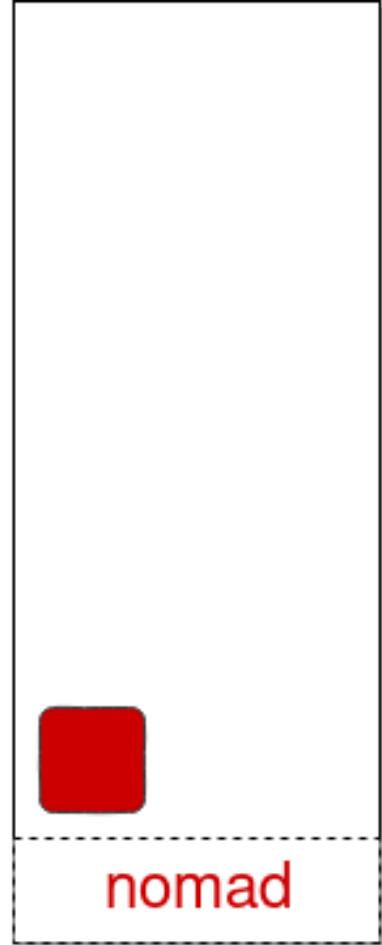
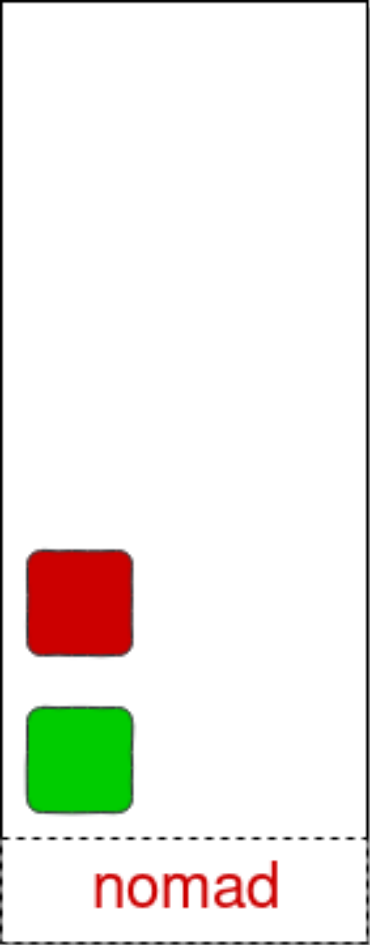
node3

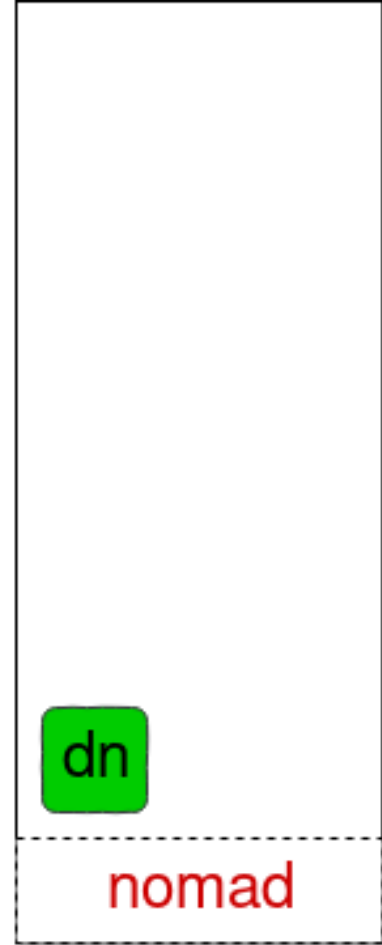
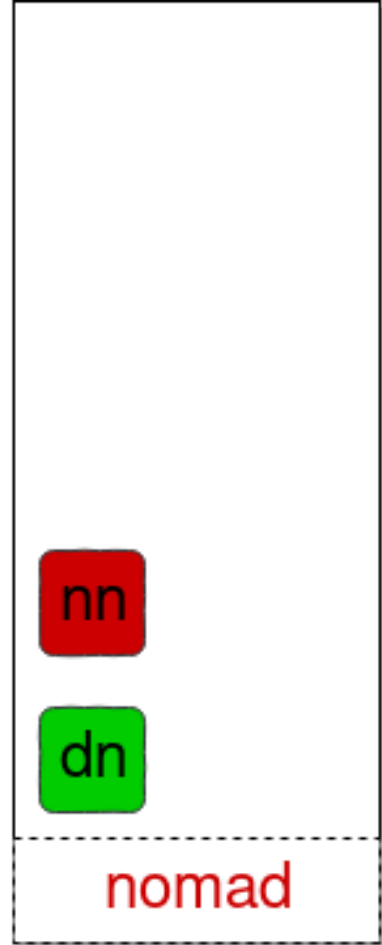
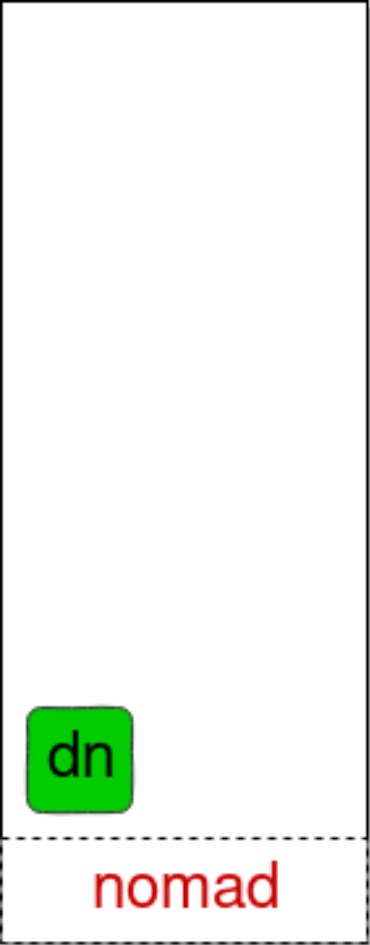


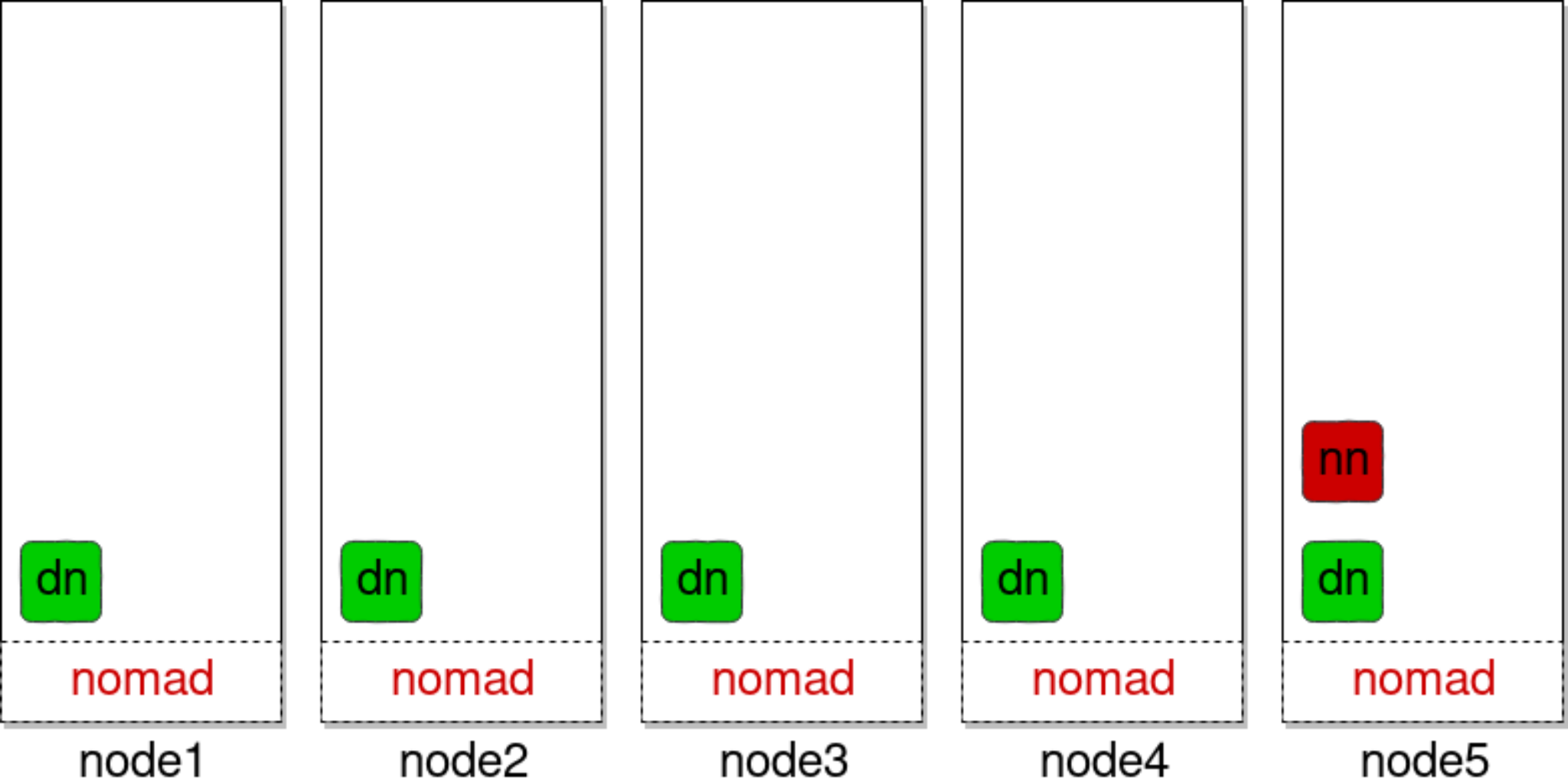
node4

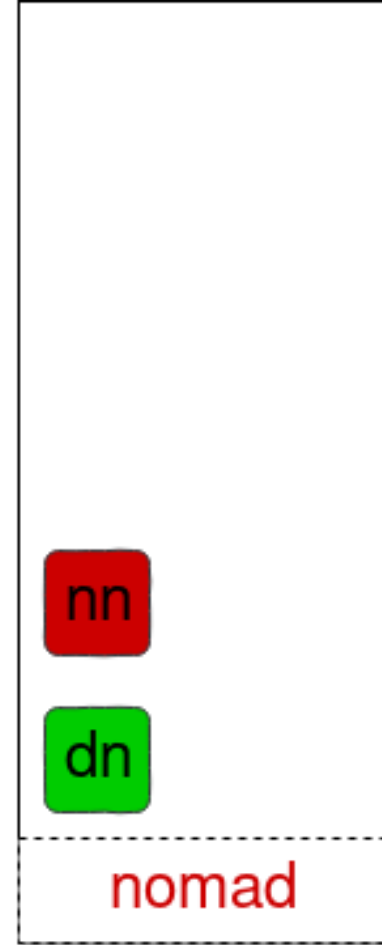
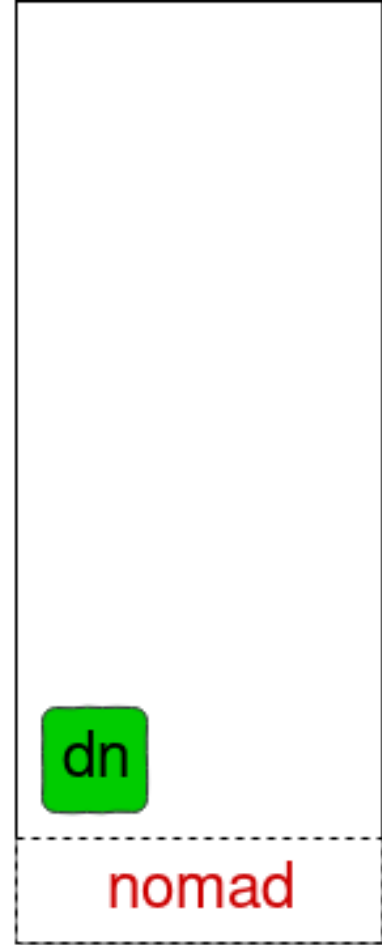
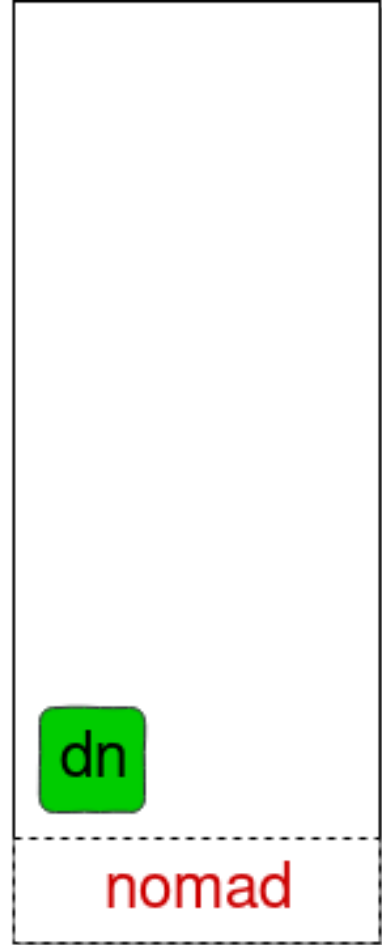
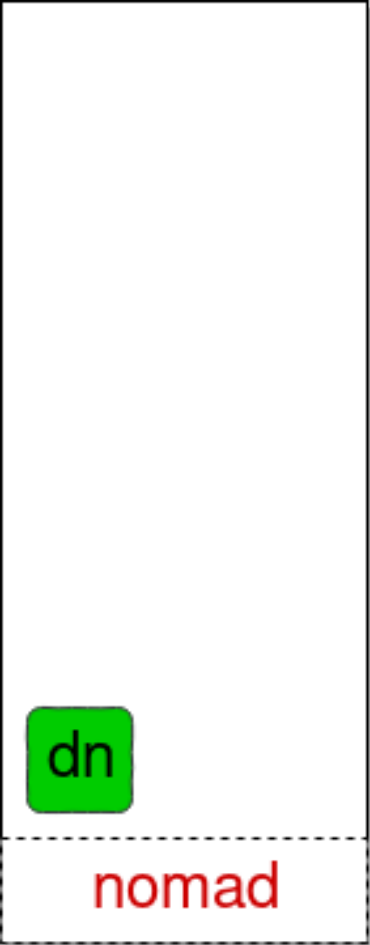


node5

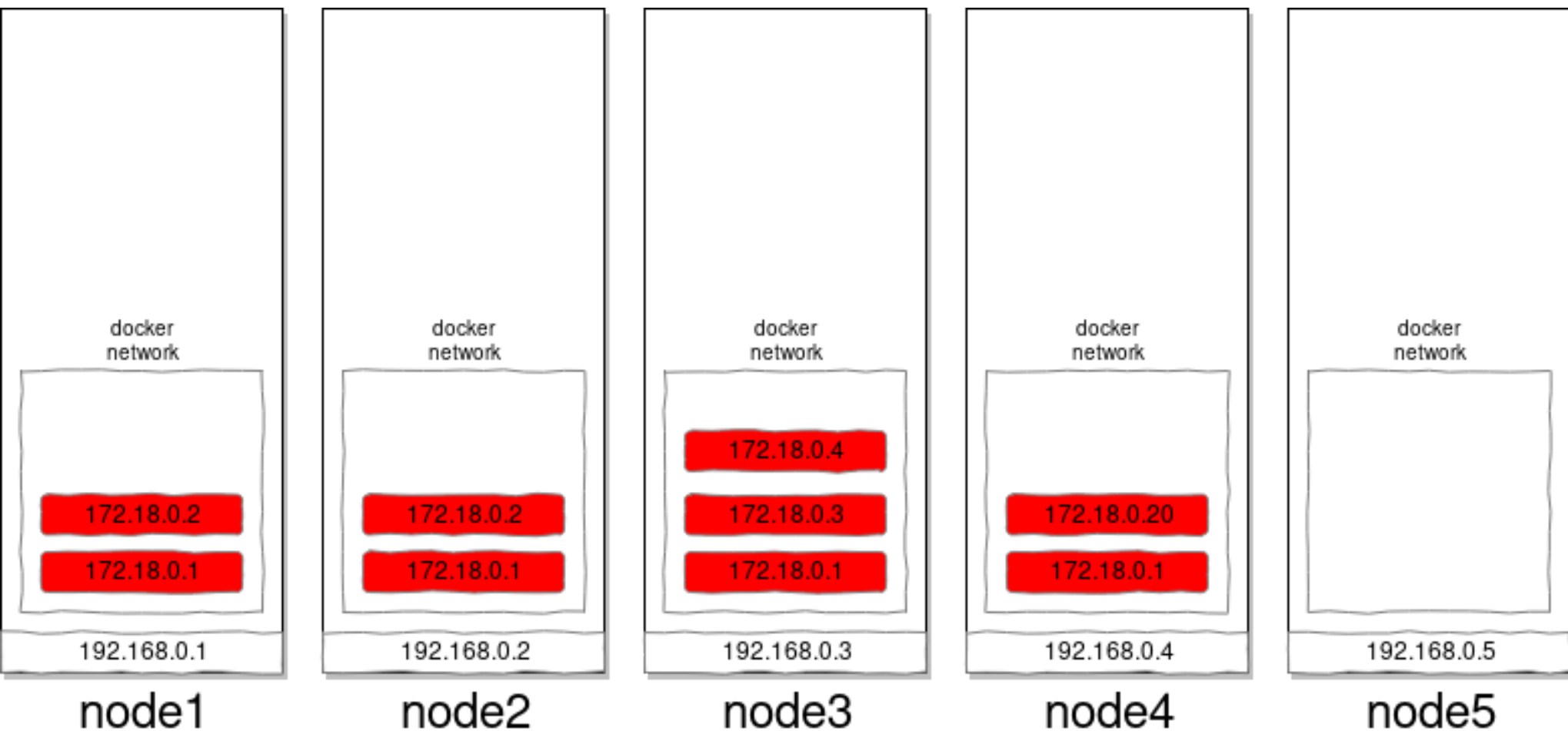




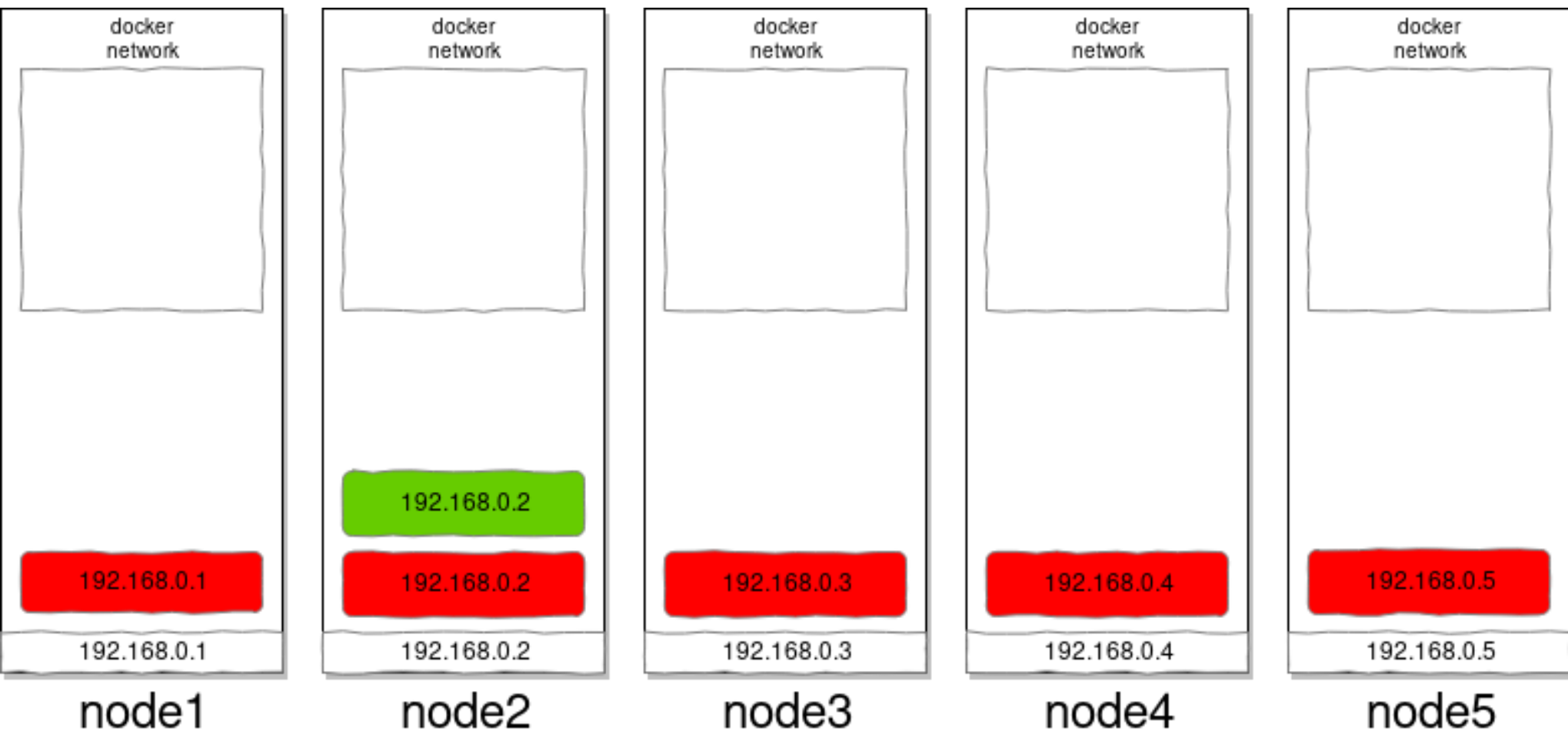




Docker network



Host network



Preprocessing

On change

Provisioning, Scheduling

Multihost support

Scheduling

Cluster definition

Scaling

Multi tenancy

Failover

Network

Intraservice network

DNS

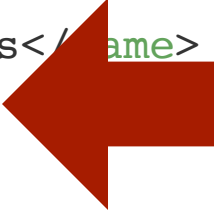
Service discovery

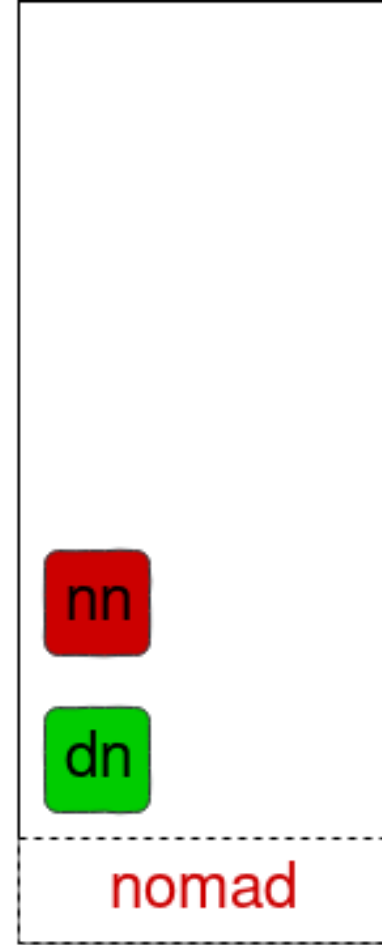
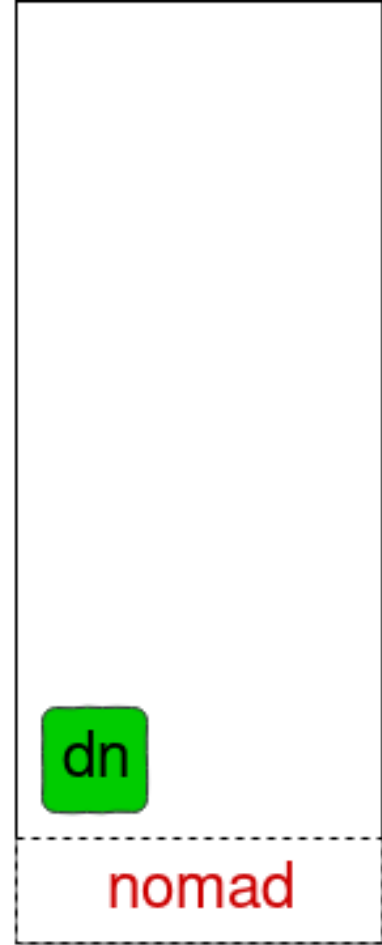
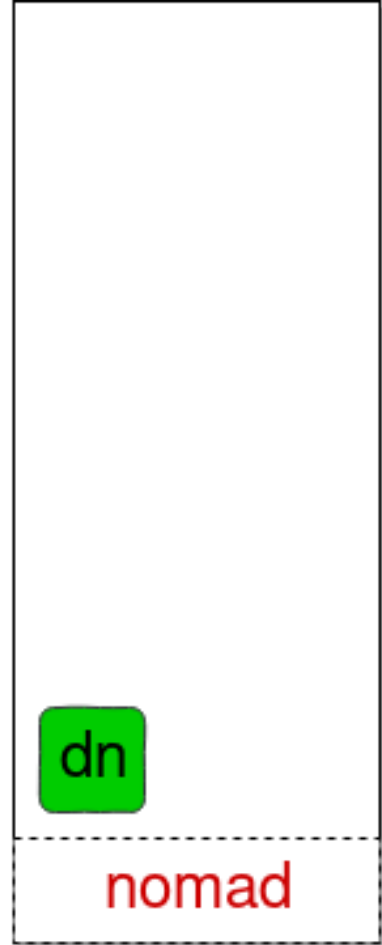
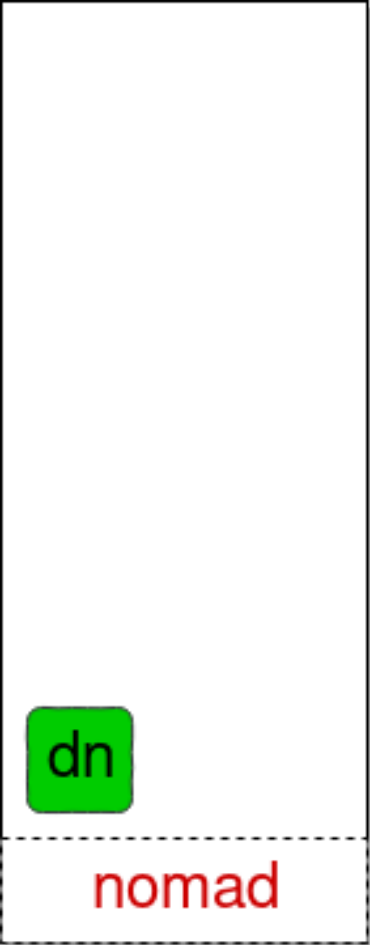
Data locality

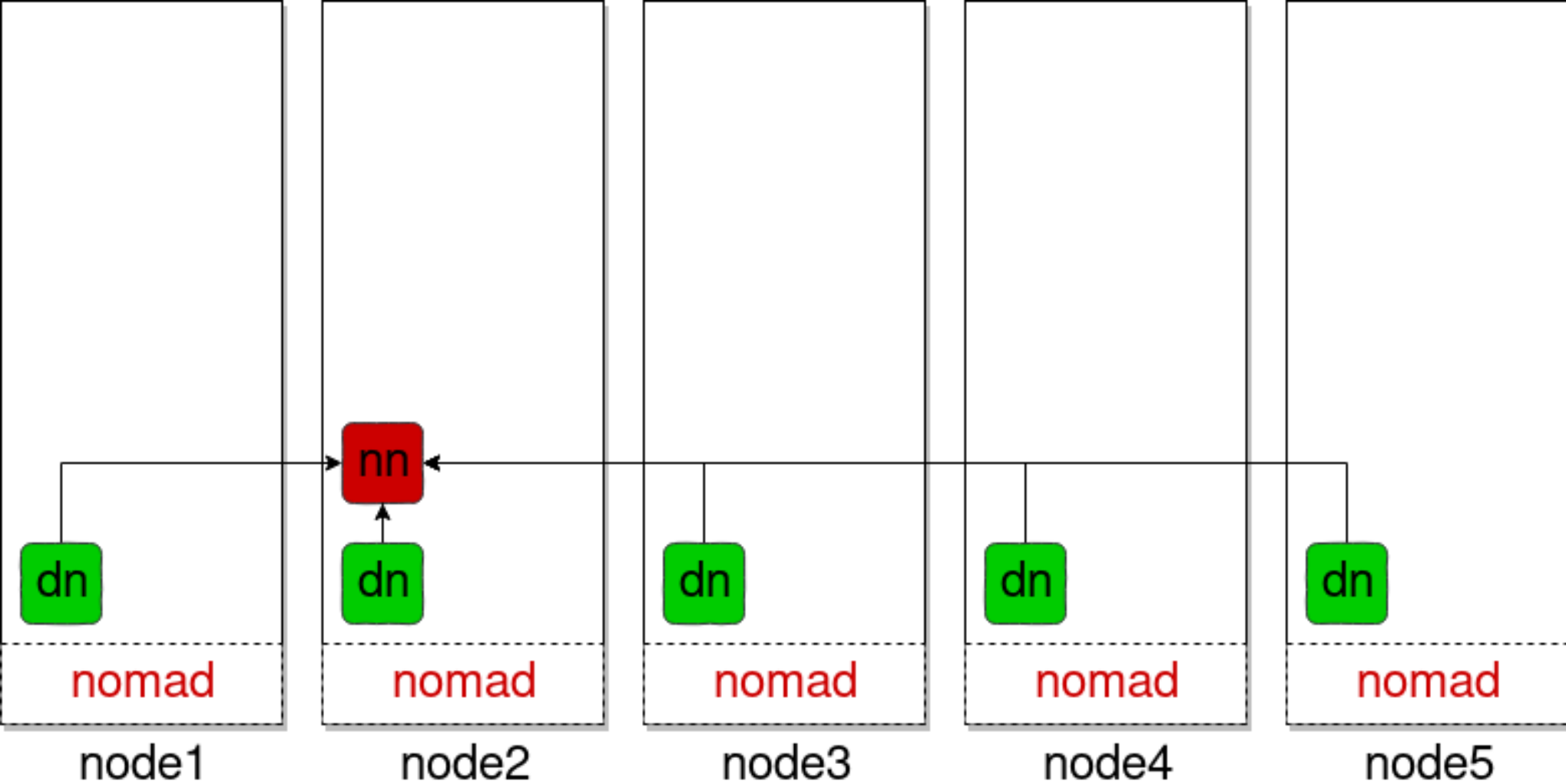
Availability of the ports

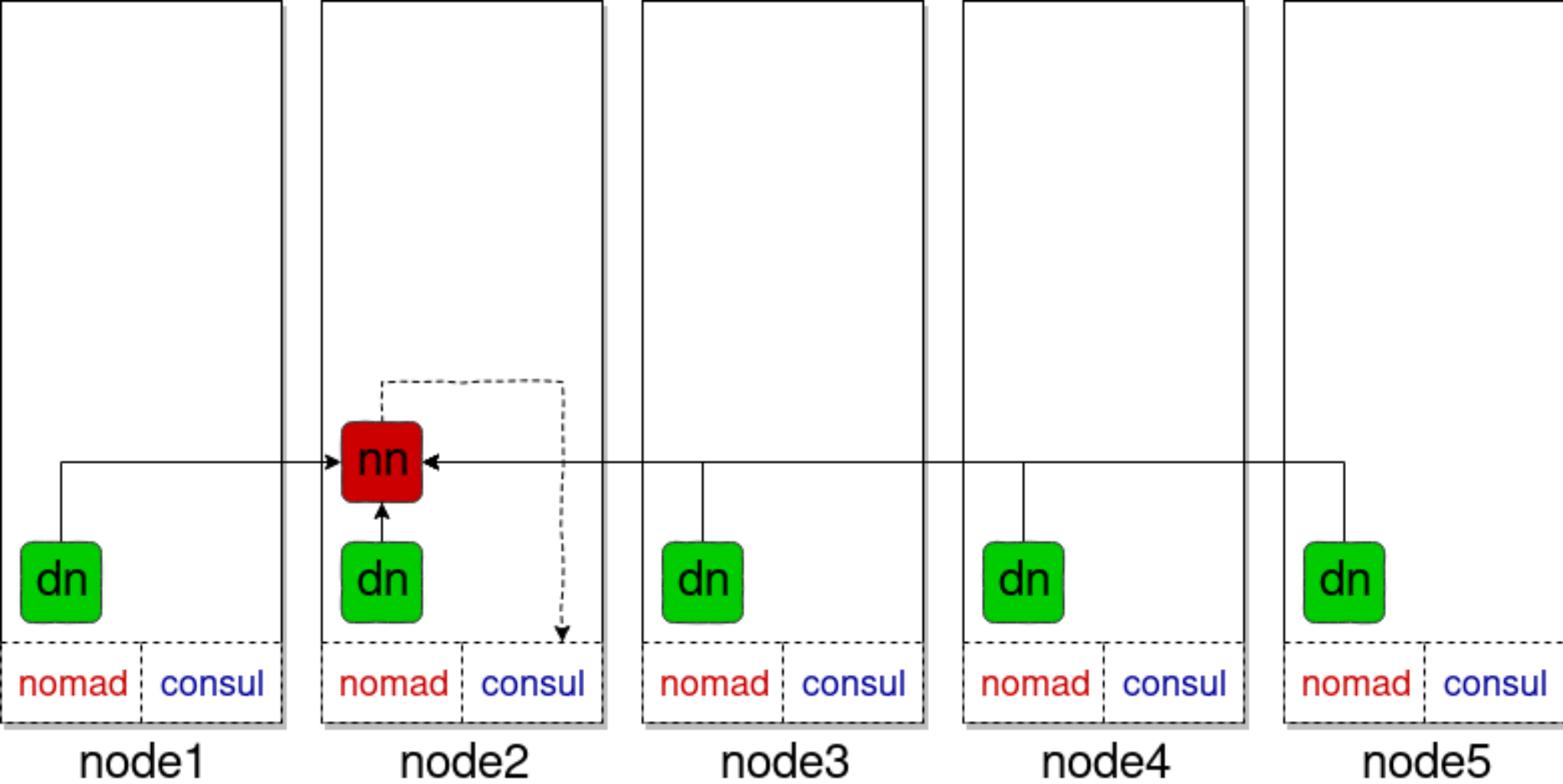
docker host
host dns

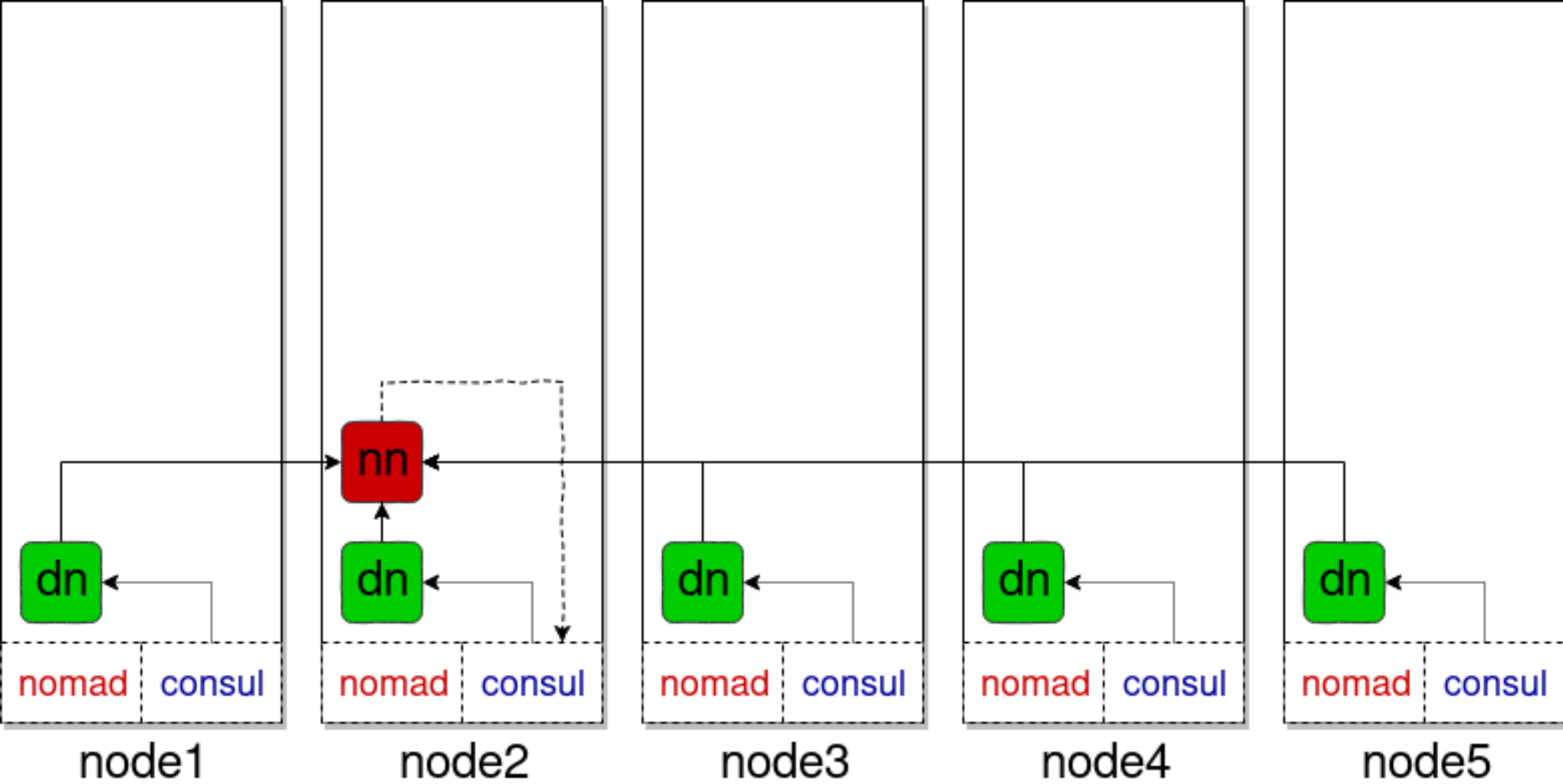

```
<configuration>
  <property>
    <name>dfs.namenode.rpc-address</name>
    <value>namenode:9000</value>
  </property>
  <property>
    <name>dfs.datanode.plugins</name>
    <value>org.apache.hadoop.ozone.HddsDatanodeService</value>
  </property>
  <property>
    <name>rpc.metrics.percentiles.intervals</name>
    <value>60,300</value>
  </property>
  <property>
    <name>dfs.namenode.name.dir</name>
    <value>/data/namenode</value>
  </property>
  <property>
    <name>rpc.metrics.quantile.enable</name>
    <value>true</value>
  </property>
</configuration>
```

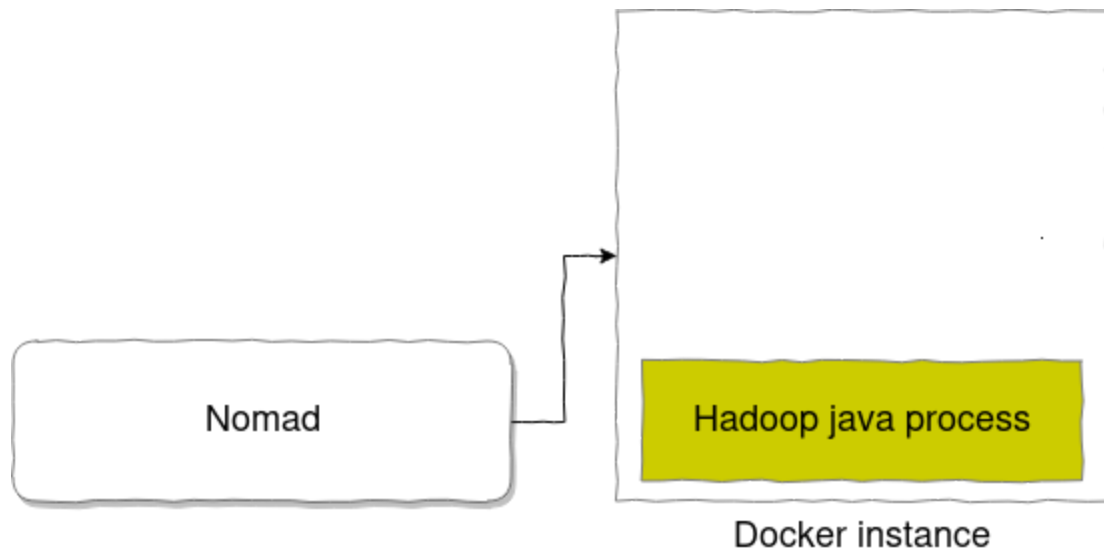


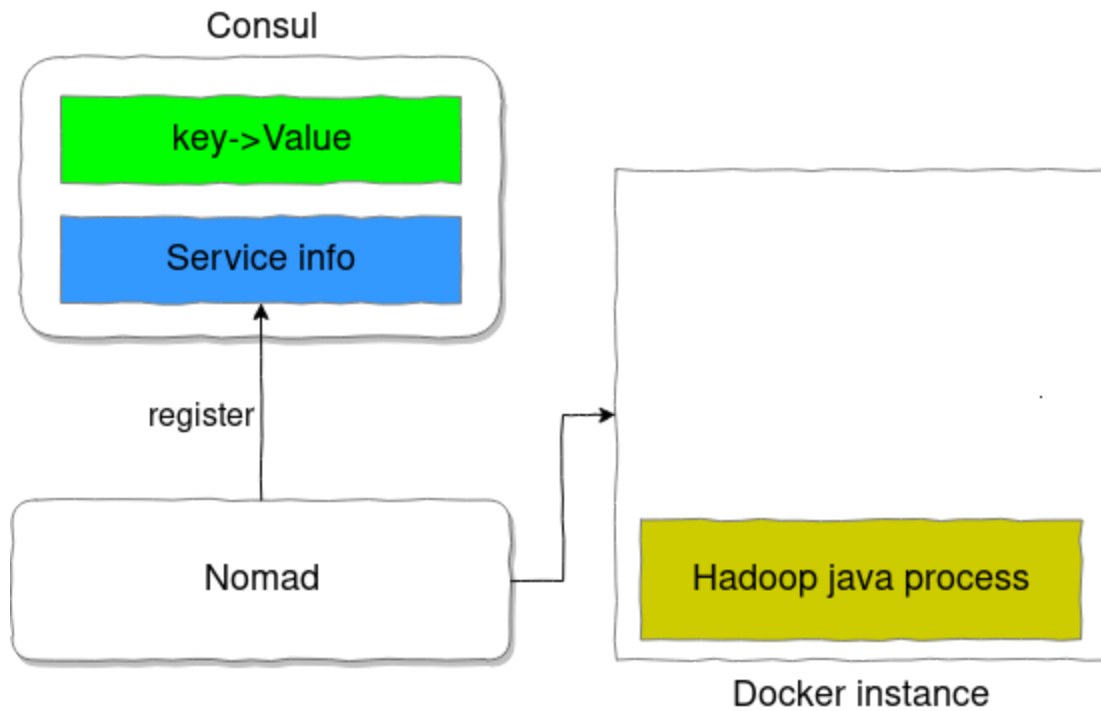


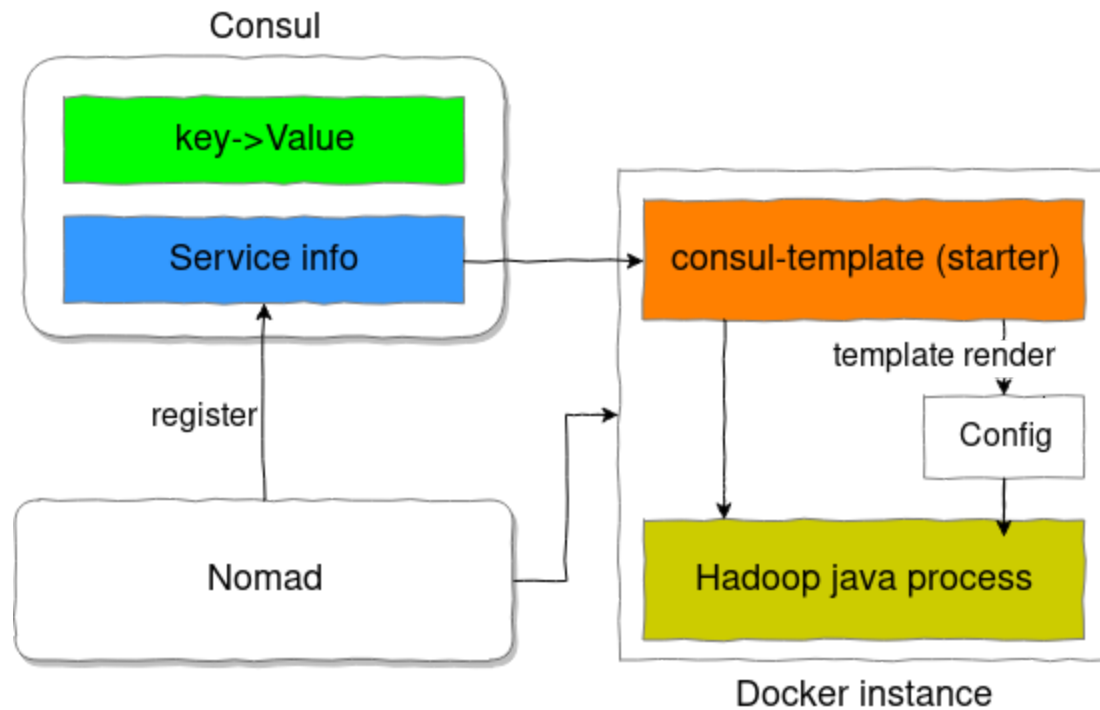


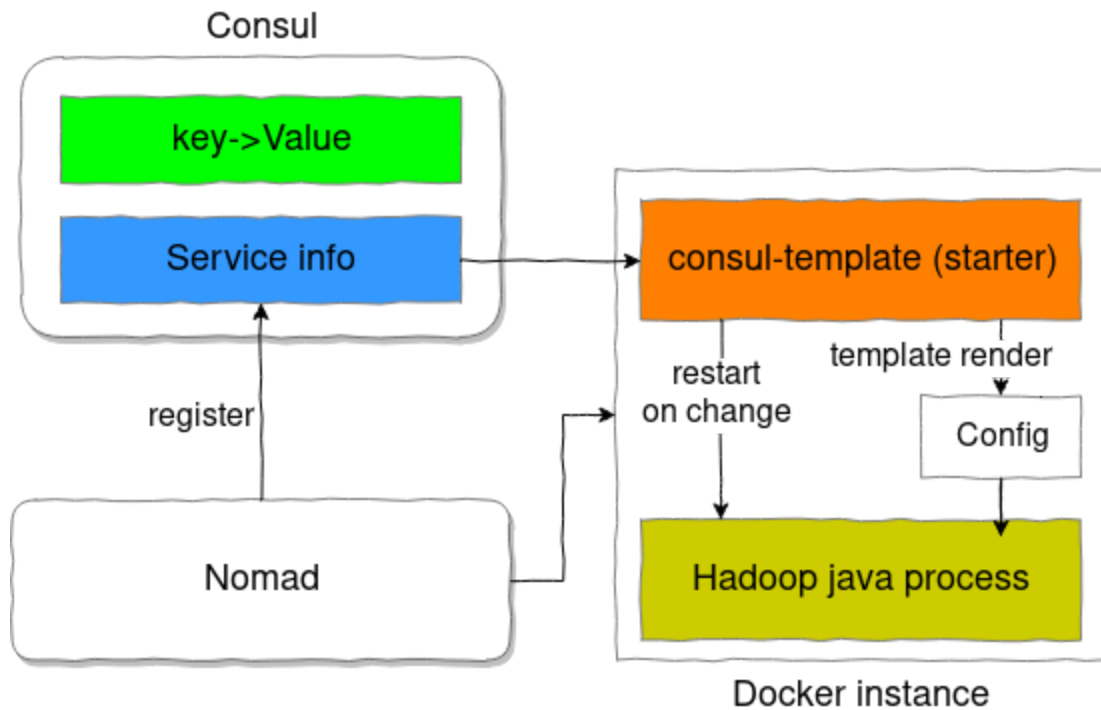


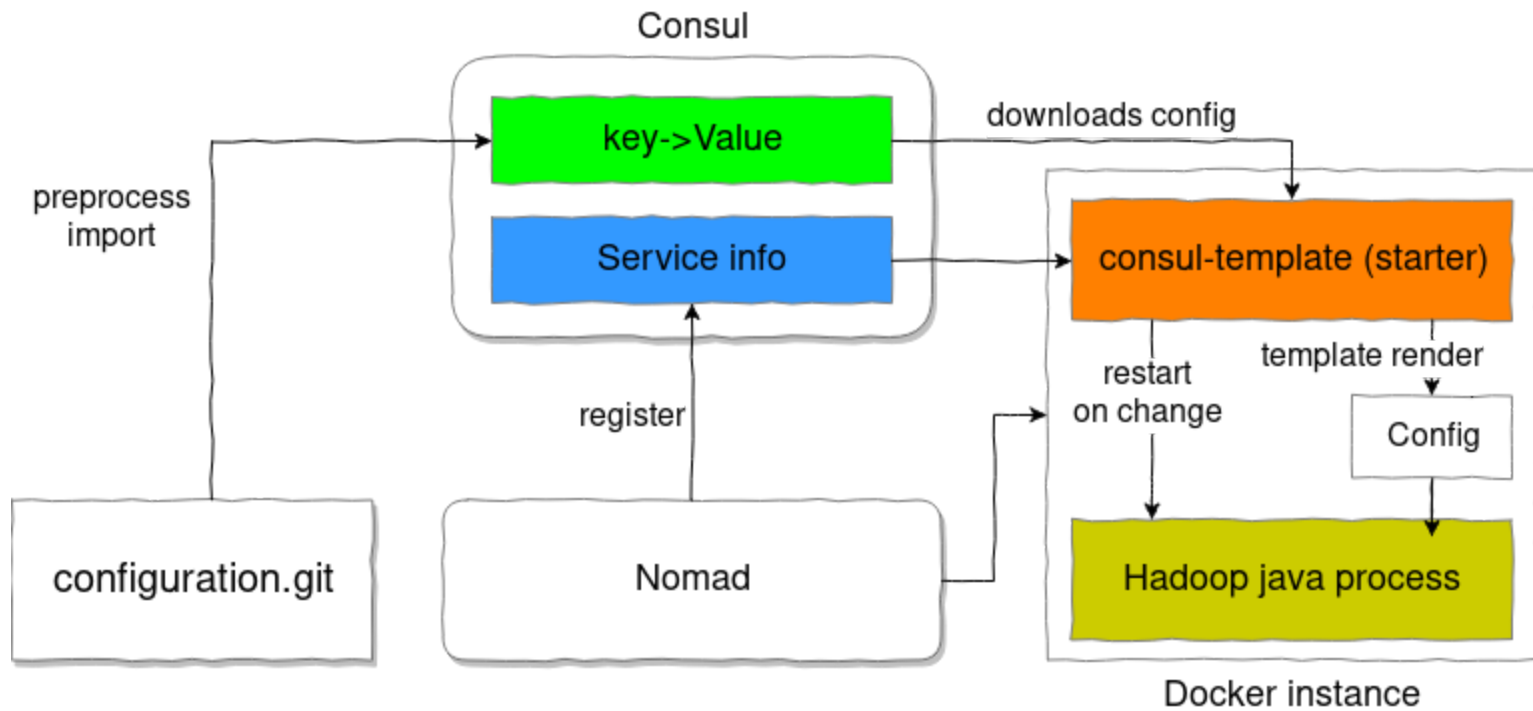














DIY



Nodes

ALL (10) Pending (0) Warning (0) Critical (0)

Healthy Nodes

namenode1 10.14.10.117	namenode2 10.14.10.118	namenode3 10.14.10.119
----------------------------------	----------------------------------	----------------------------------



```
2018-06-04 21:24:04 INFO DataNode:422 - Successfully sent block report 0x763c45c65afe8d18, containing 1 storage report(s), of which we sent 1. The reports had 0 total blocks and used 1 RPC(s). This took 12 msec to generate and 75 msec for RPC and NN processing. Got back one command: FinalizeCommand/5.
```

```
2018-06-04 21:24:04 INFO DataNode:759 - Got finalize command for block pool BP-643505683-127.0.0.1-1528147013324
```

```
2018-06-04 21:24:01 INFO DataNode:422 - Successfully sent block report 0x900e6d4b9e81dba1, containing 1 storage report(s), of which we sent 1. The reports had 8 total blocks and used 1 RPC(s). This took 7 msec to generate and 49 msec for RPC and NN processing. Got back one command: FinalizeCommand/5.
```

```
2018-06-04 21:24:01 INFO DataNode:759 - Got finalize command for block pool BP-643505683-127.0.0.1-1528147013324
```

Less efficient

Configuration management

Source

Preprocessing

On change

Consul

Yes (script)

Restart

Provisioning, Scheduling

Multihost support

Scheduling

Cluster definition

Scaling

Multi tenancy

Failover

host netw

Nomad

.nomad

redploy

no

yes

Network

Intraservice network

DNS

Service discovery

Data locality

Availability of the ports

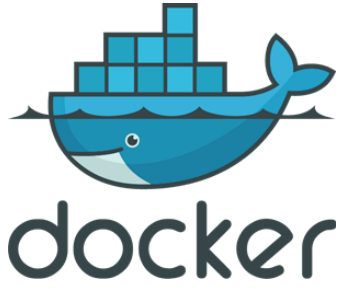
host netw

yes

consul

yes

host

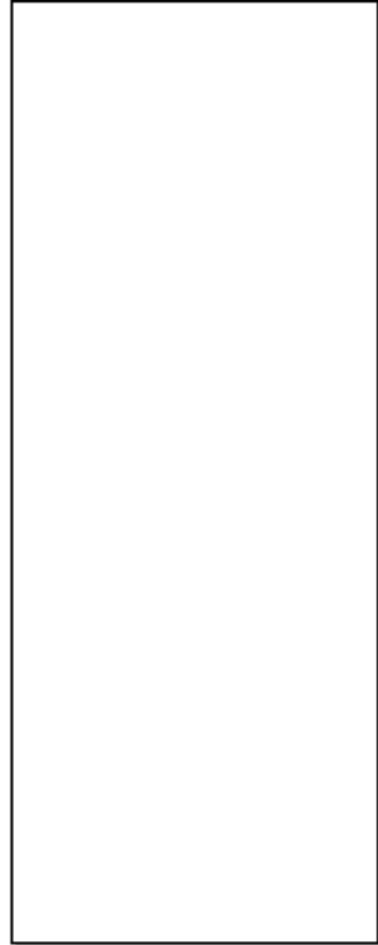


Kubernetes

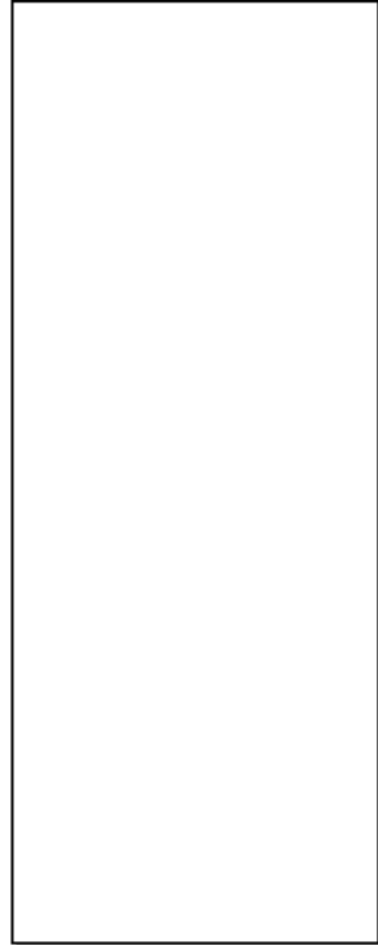
"out of the box"



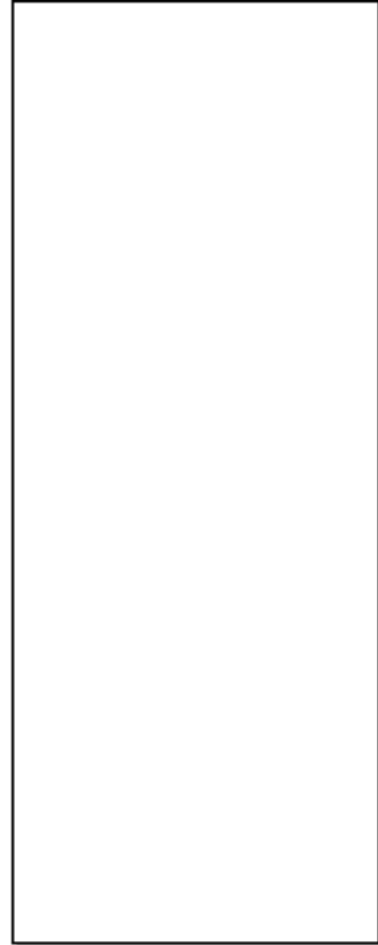
node1



node2



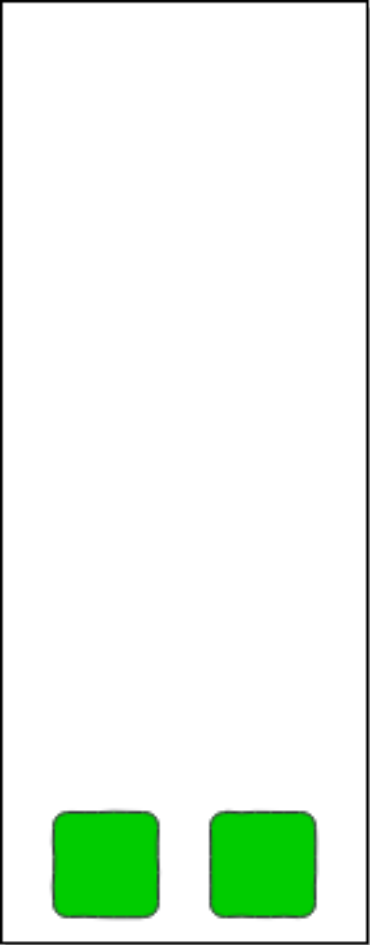
node3



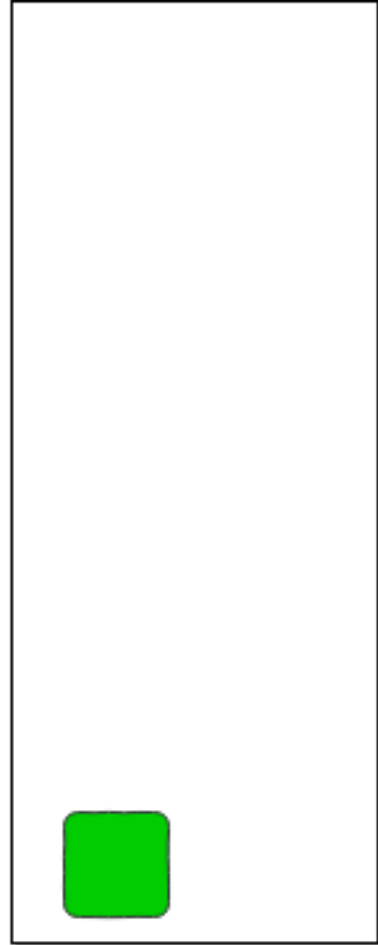
node4



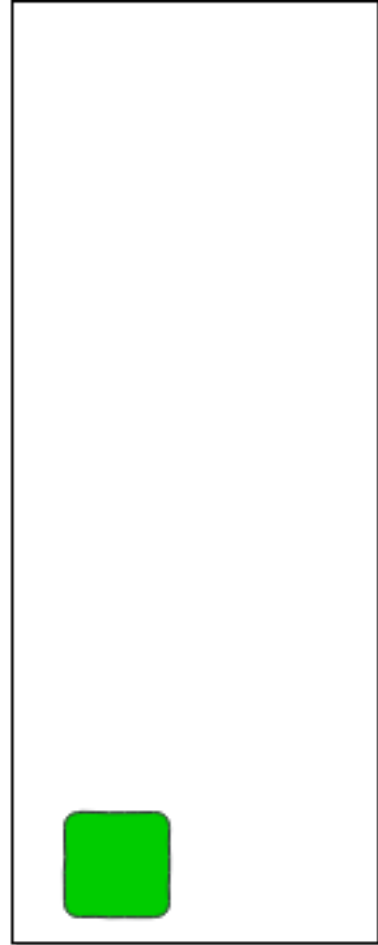
node5



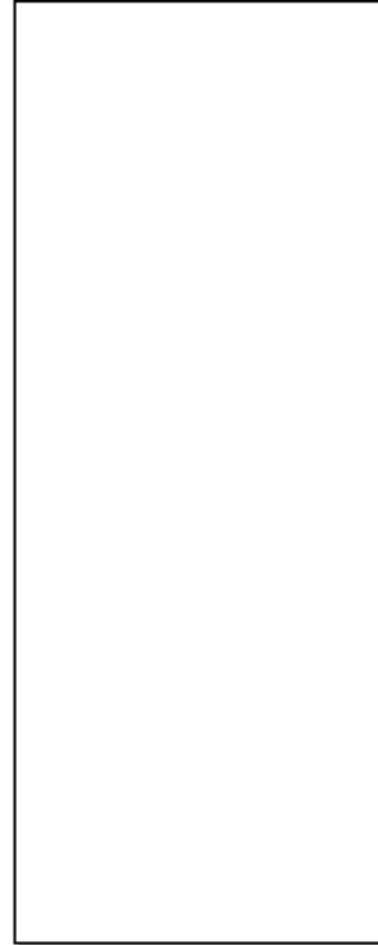
node1



node2



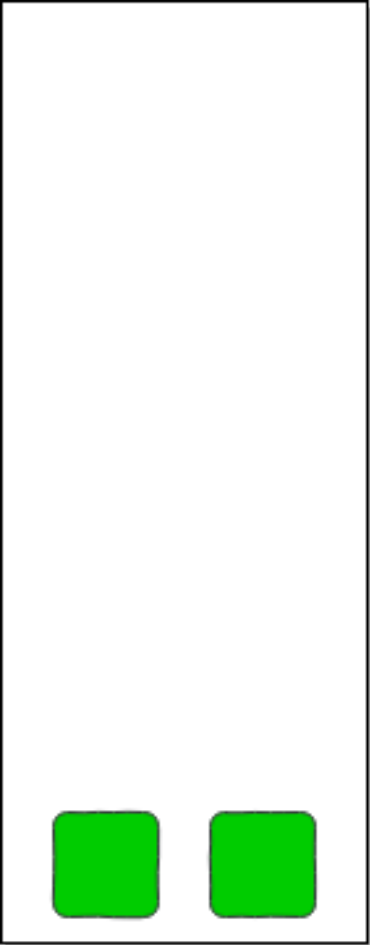
node3



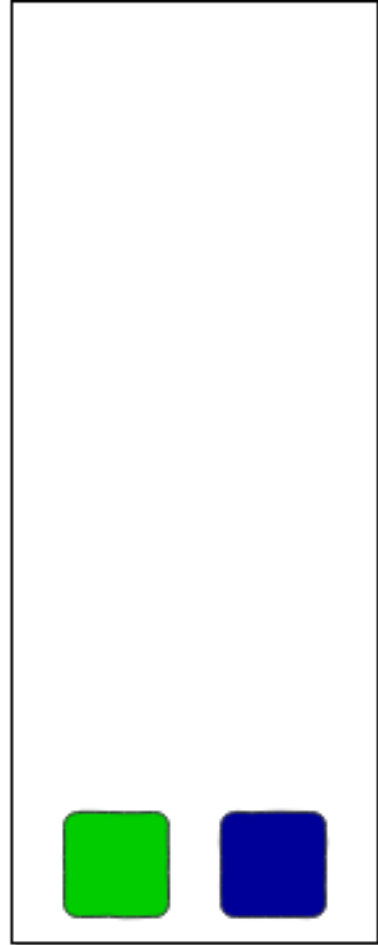
node4



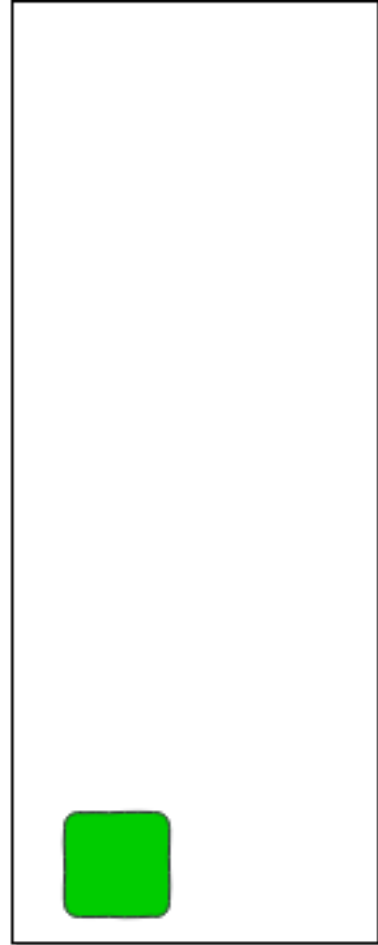
node5



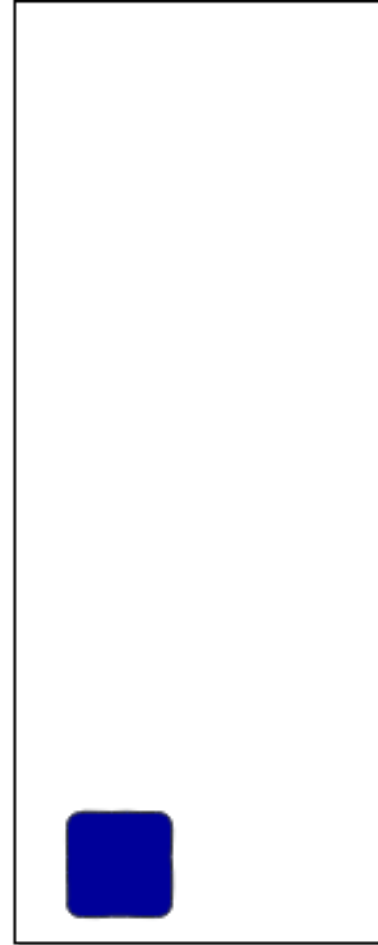
node1



node2



node3



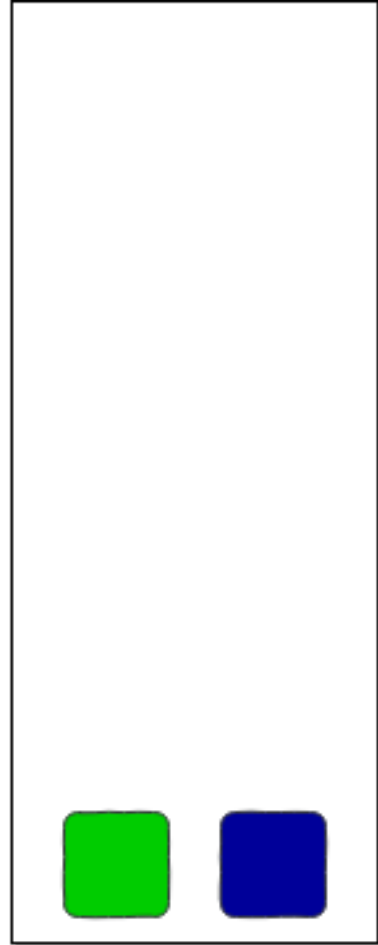
node4



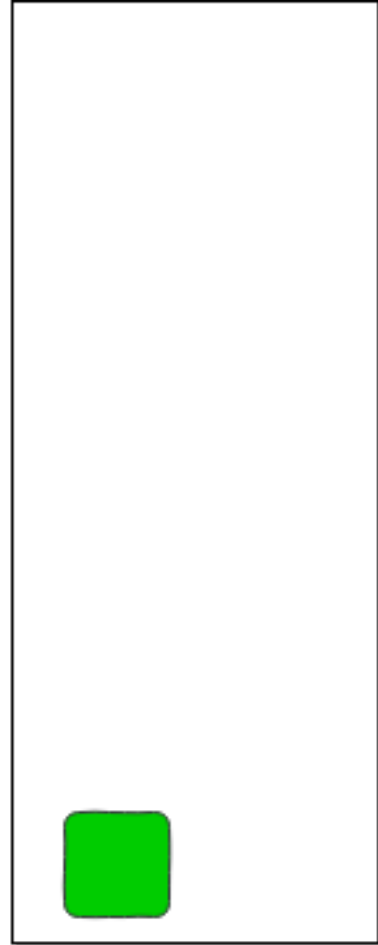
node5



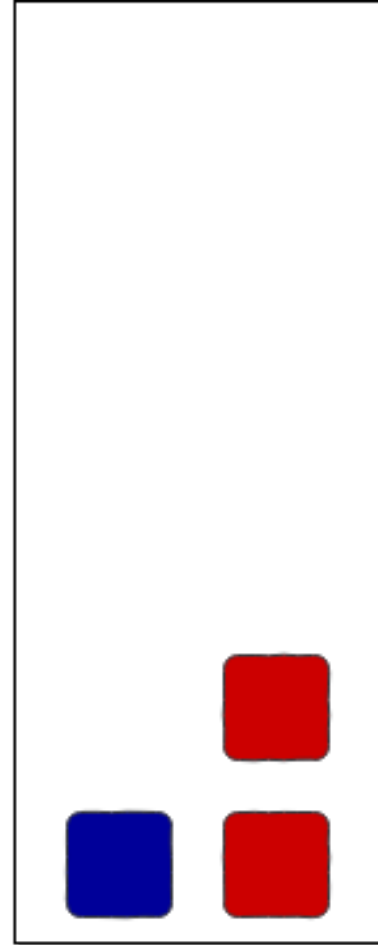
node1



node2



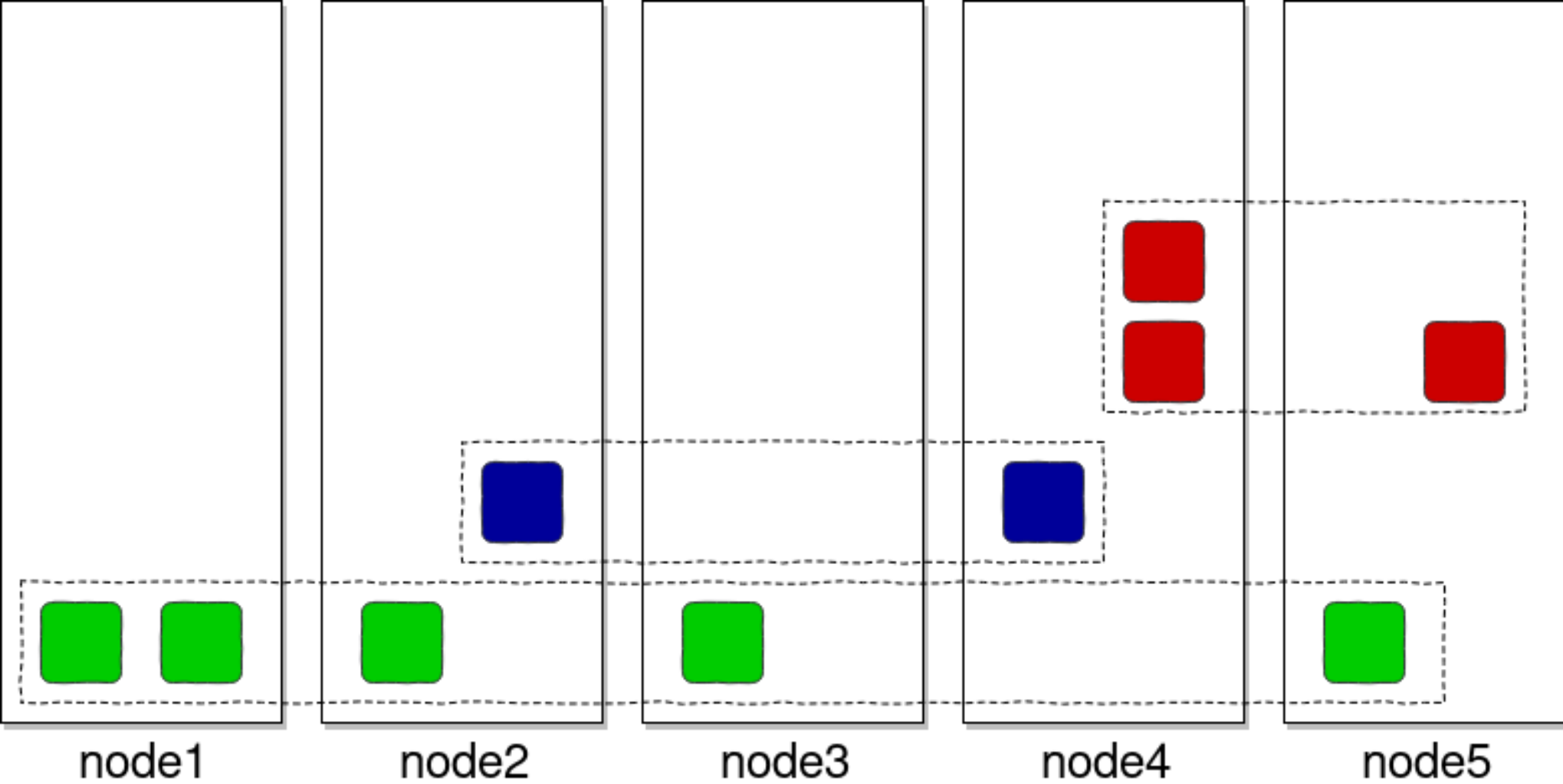
node3



node4



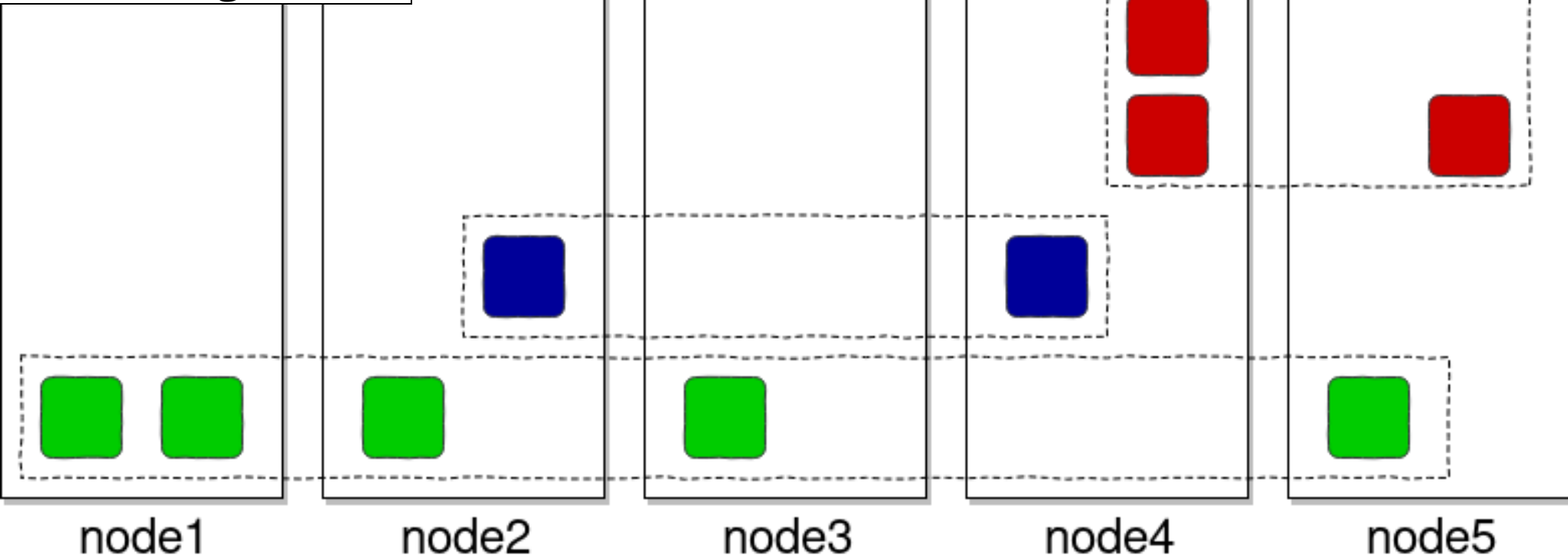
node5

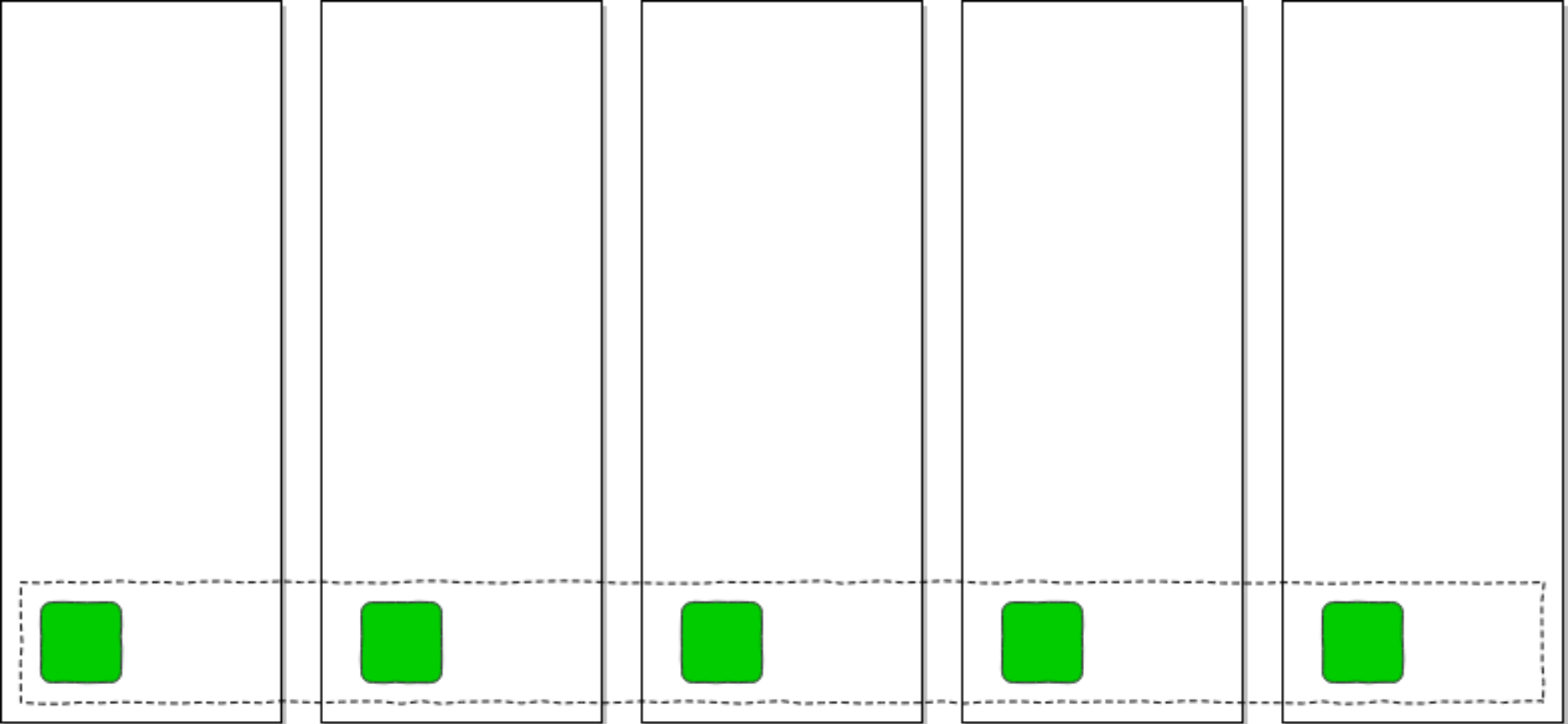


+Network!

+Storage!!

(volume, secrets,
configs)





node1

node2

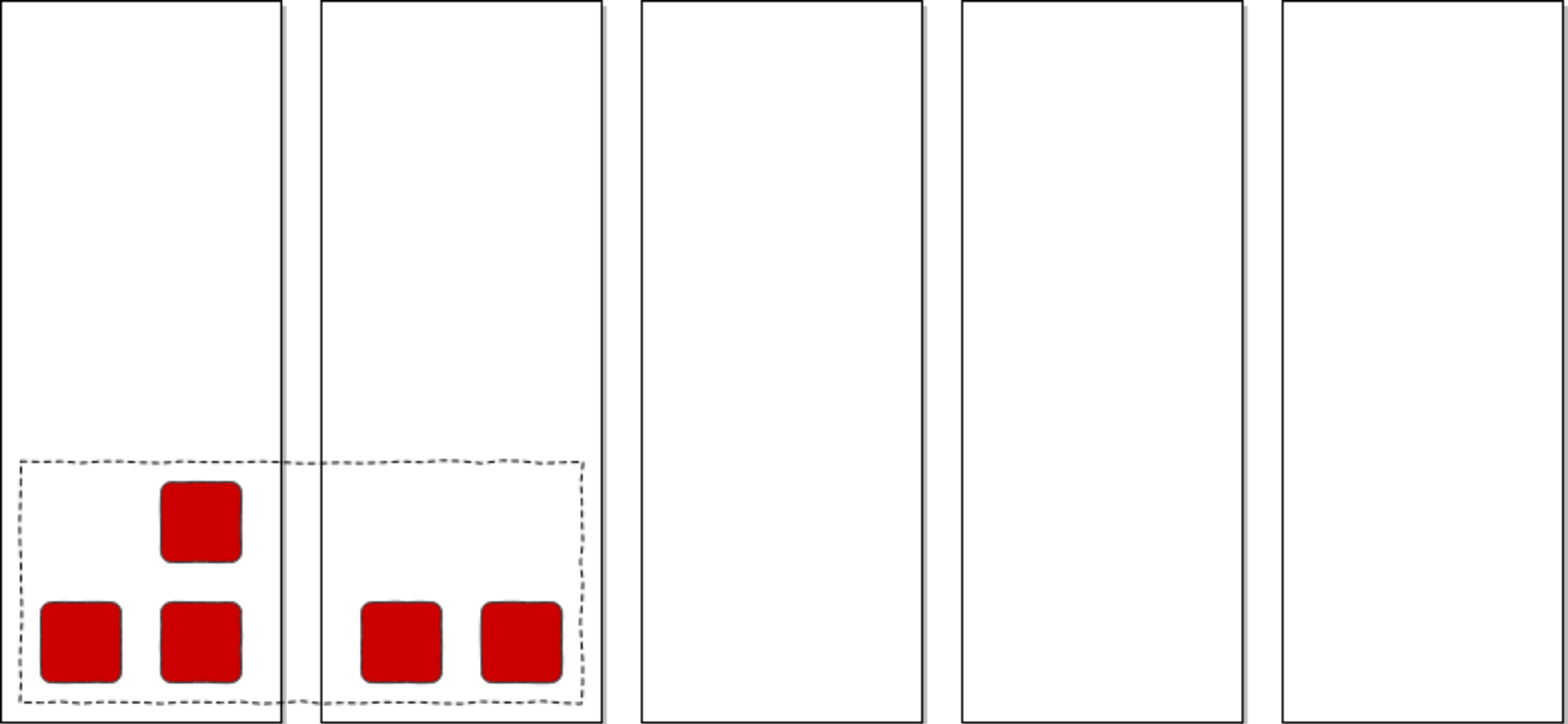
node3

node4

node5



DaemonSet



node1

node2

node3

node4

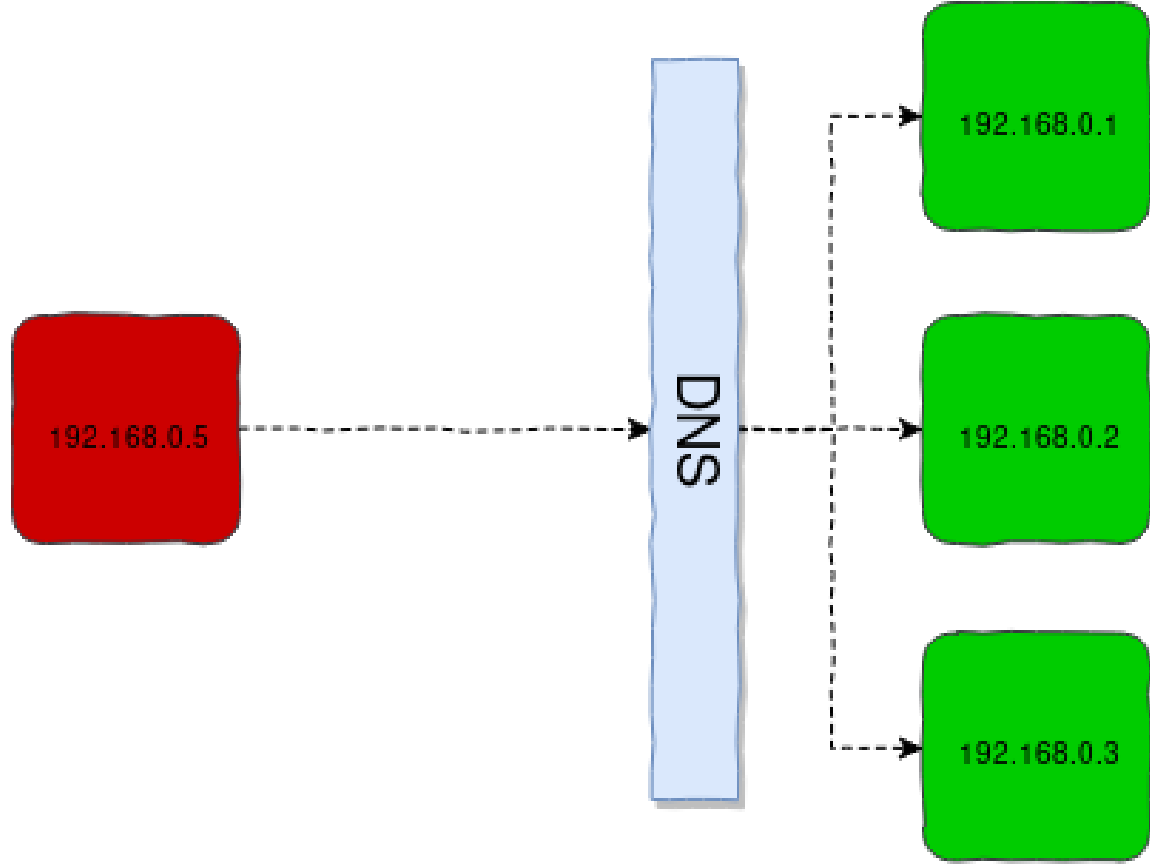
node5

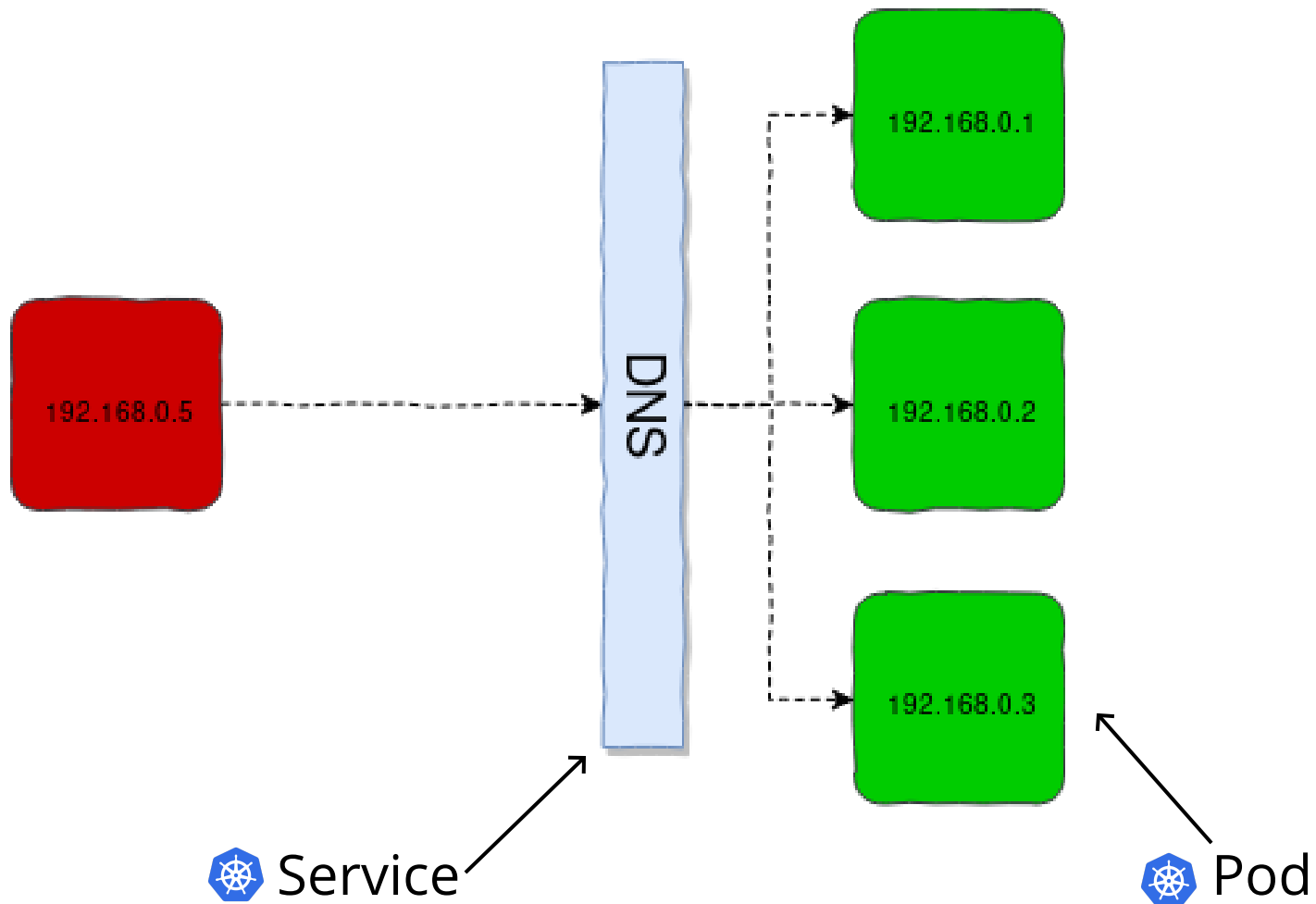
192.168.0.1

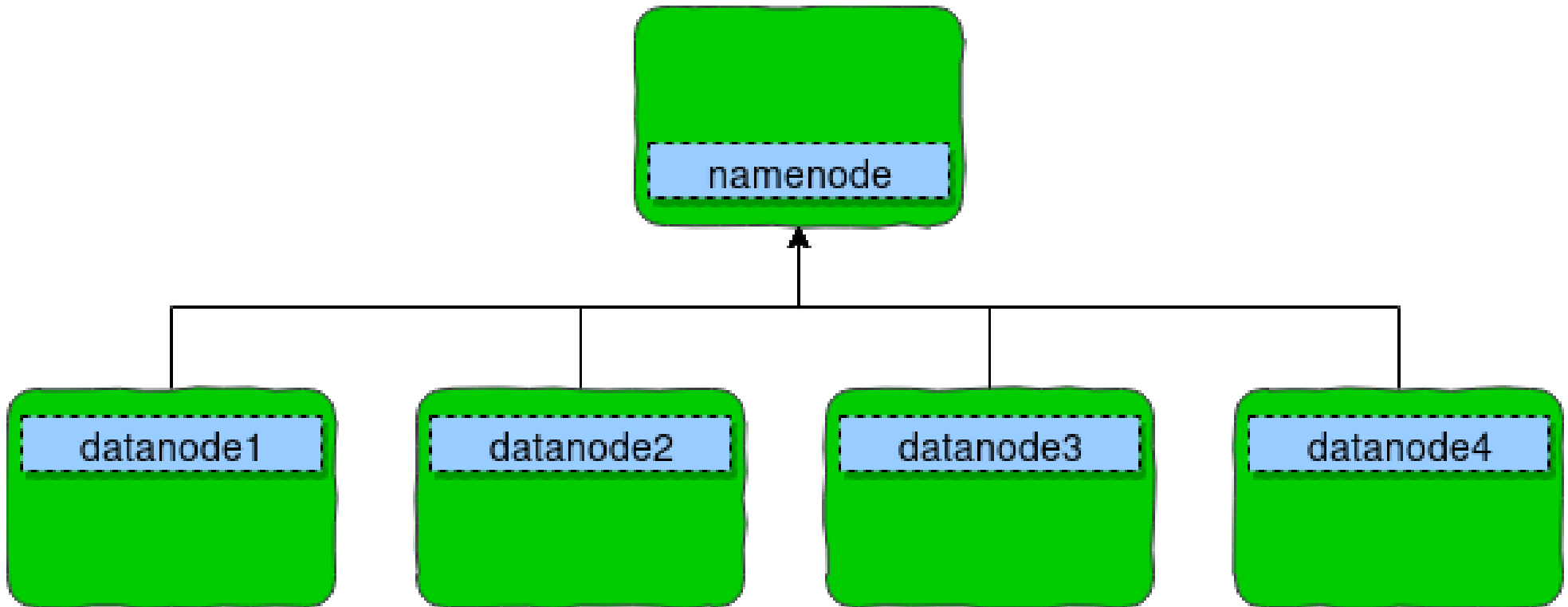
192.168.0.2

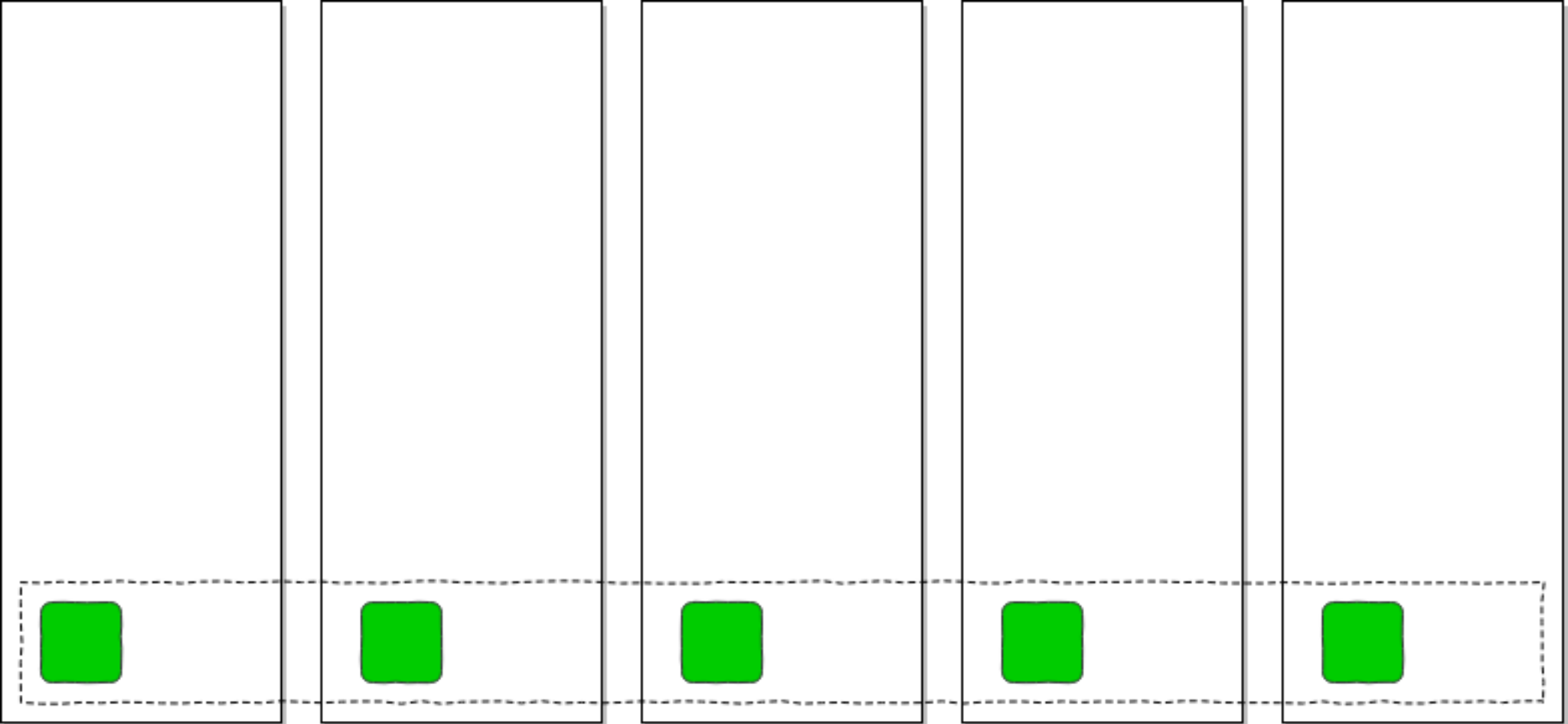
192.168.0.3











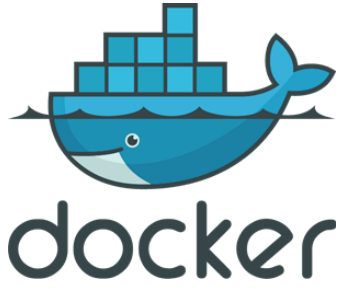
node1

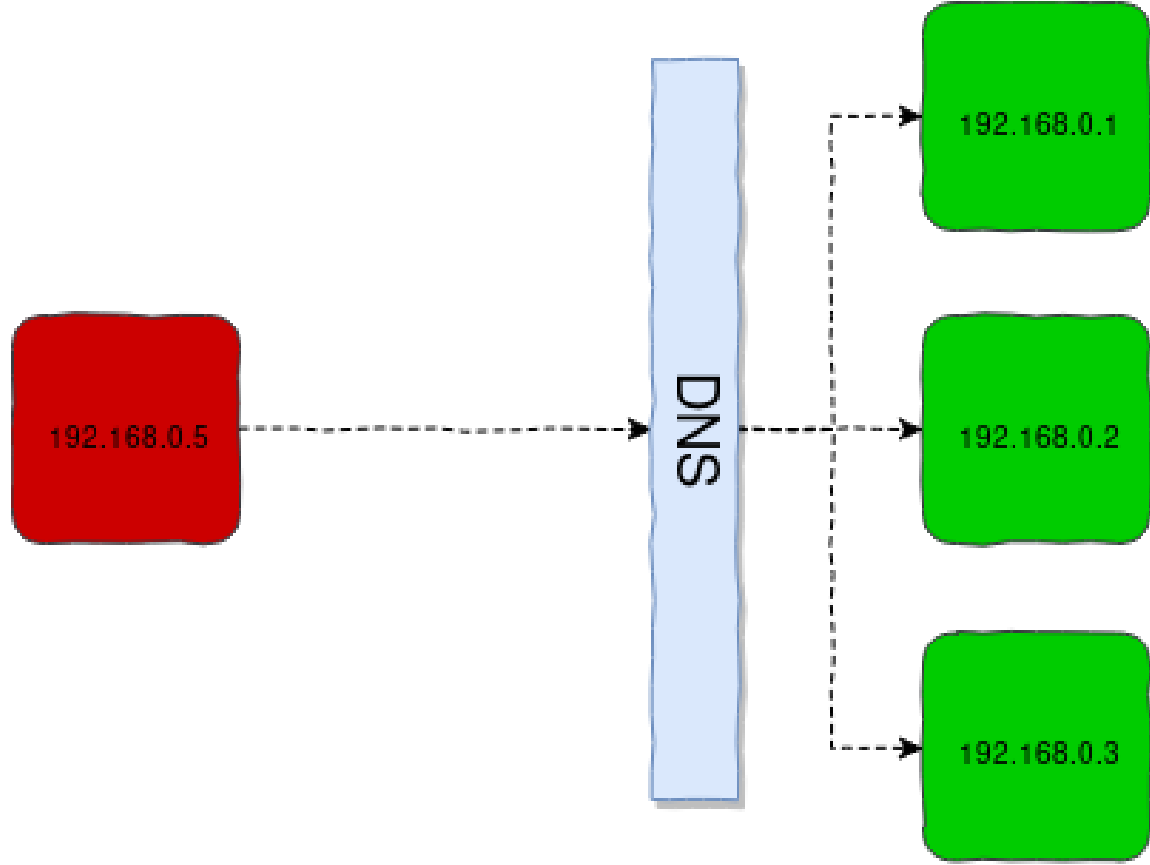
node2

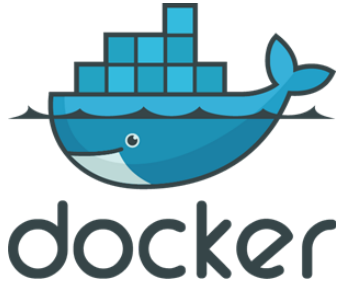
node3

node4

node5







Benefits of Hadoop + k8s?



**Benefits of
Hadoop + k8s?**

**Ecosystem
Flexibility**



Example:

Monitor Hadoop with Prometheus

Enable query history

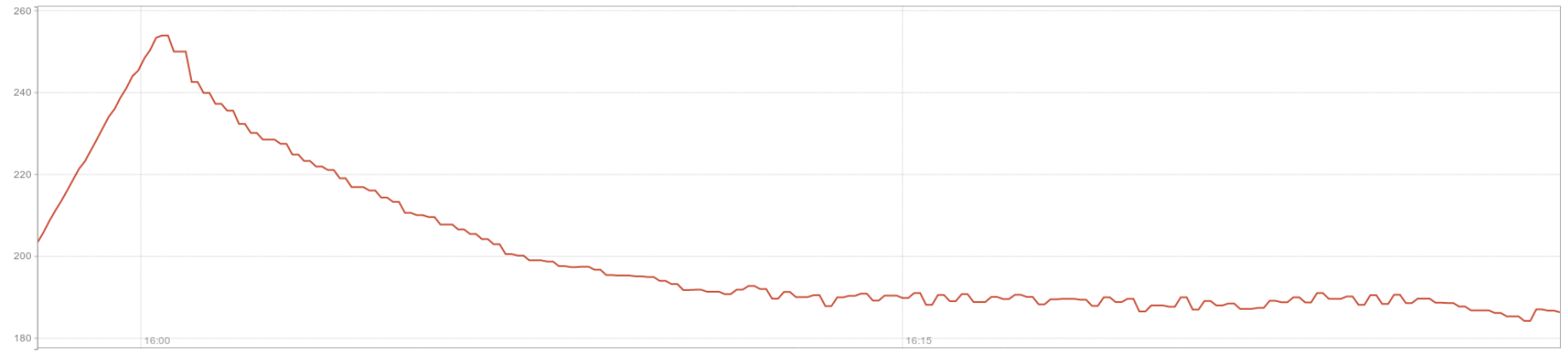
rate(Hadoop_KeySpaceManager_NumKeyCommits[10m])

Load time: 187ms
Resolution: 7s
Total time series:

Execute java_lang_OperatingSystem

Graph Console

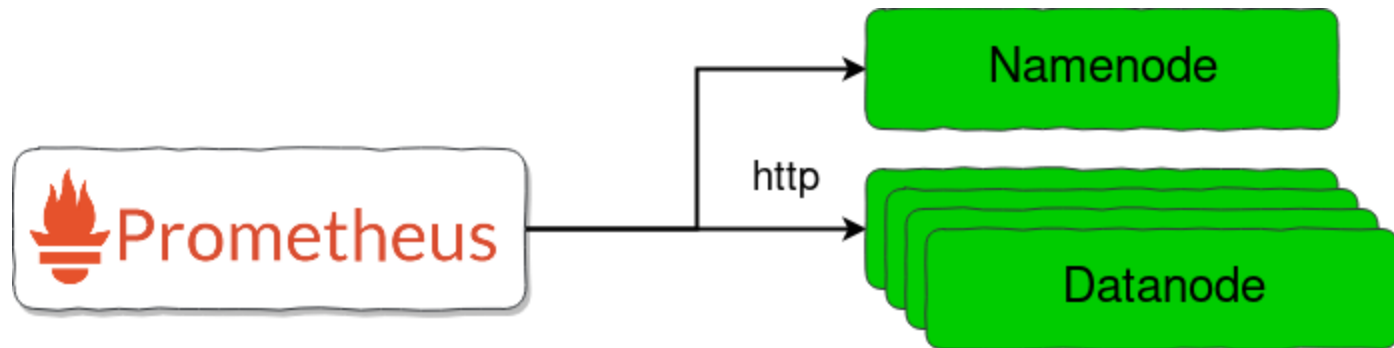
30m Until Res. (s) stacked

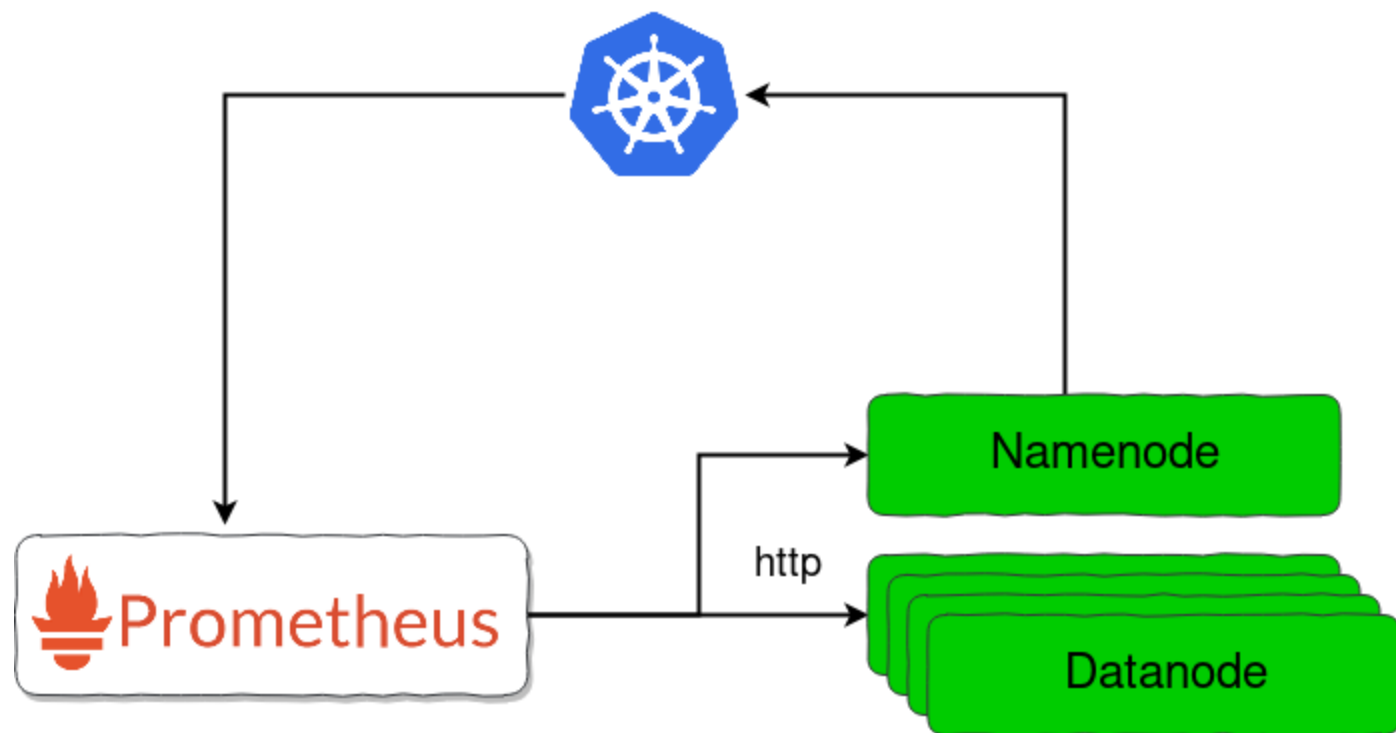


(instance="c3e21ade589b:38271",job="jmxexporter",name="KSMMetrics")

Remove Graph

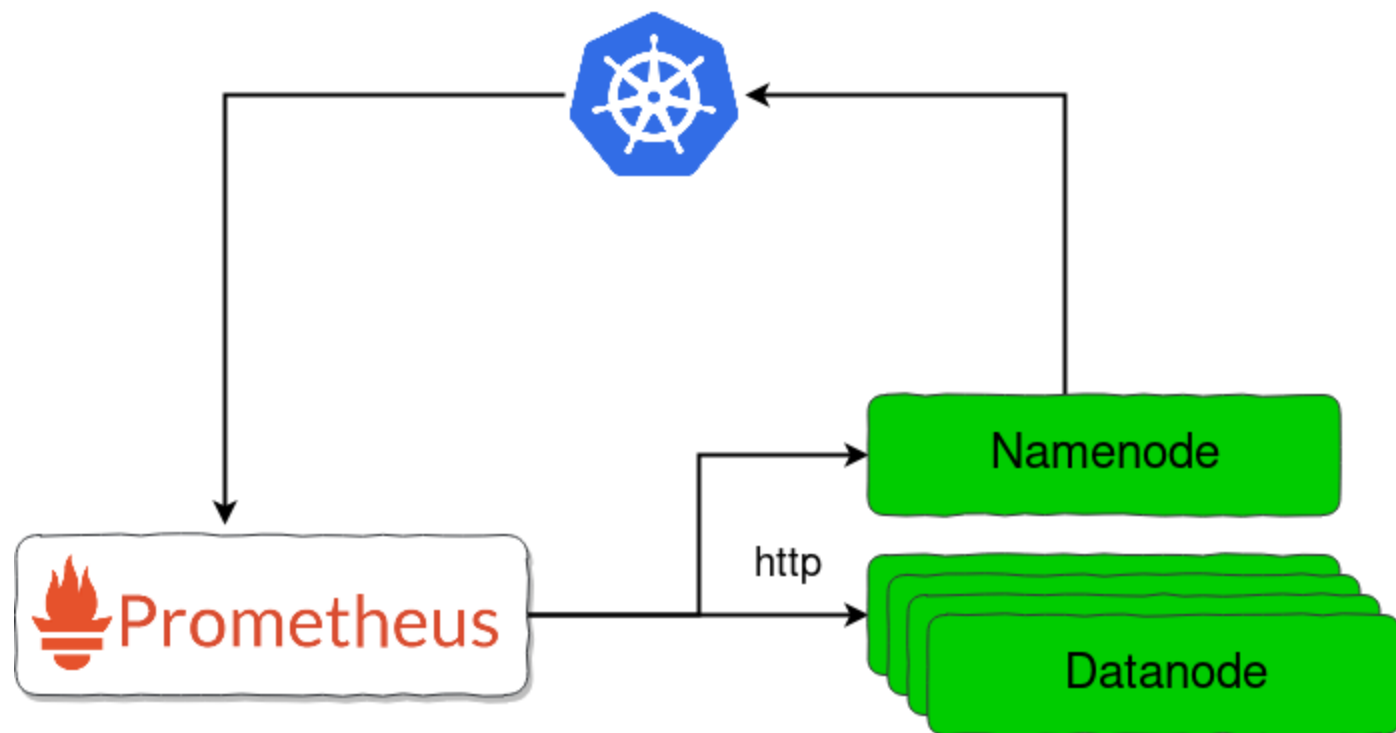
Add Graph

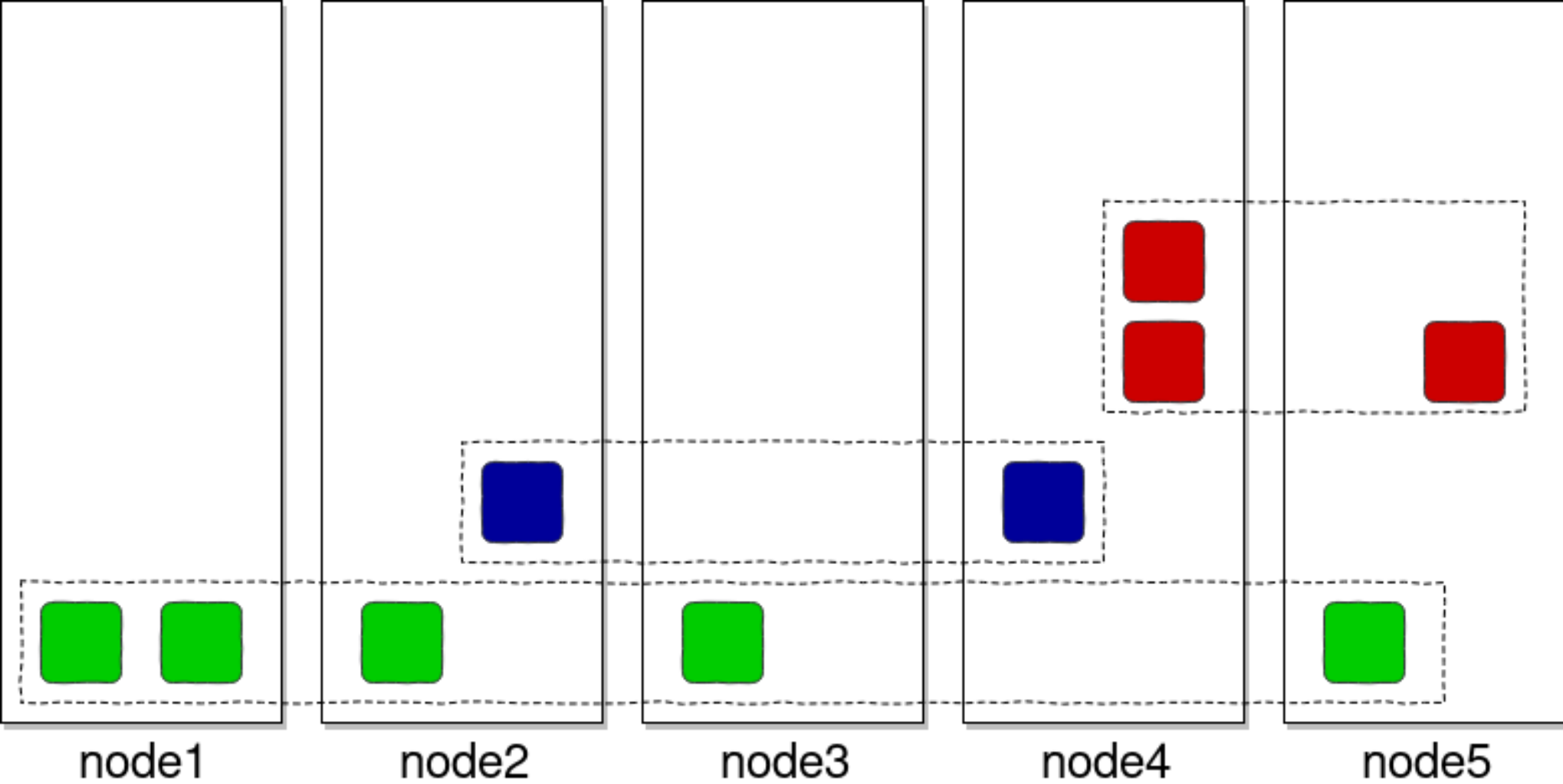


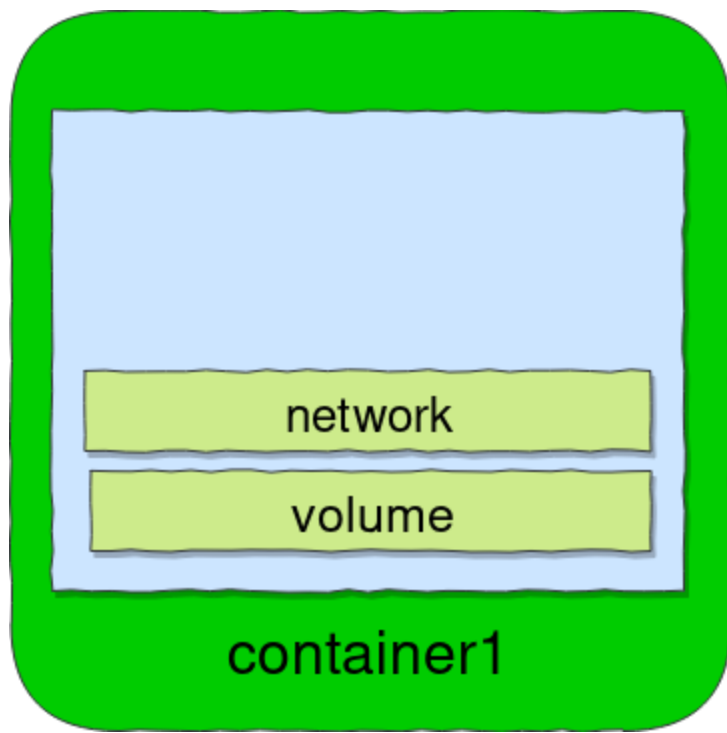


```
apiVersion: apps/v1beta1
kind: StatefulSet
metadata:
  name: ozone-hdfs-namenode
spec:
  serviceName: ozone2-hdfs-namenode
  replicas: 1
  template:
    metadata:
      labels:
        app: ozone
    spec:
      containers:
        - name: hdfs-namenode
          image: flokkkr/ozone:2.1.0
          args: [ "hdfs", "namenode" ]
```

```
apiVersion: apps/v1beta1
kind: StatefulSet
metadata:
  name: ozone-hdfs-namenode
spec:
  serviceName: ozone2-hdfs-namenode
  replicas: 1
  template:
    metadata:
      labels:
        app: ozone
      annotations:
        prometheus.io/scrape: "true"
        prometheus.io/port: "28942"
    spec:
      containers:
        - name: hdfs-namenode
          image: flokkkr/ozone:2.1.0
          args: [ "hdfs", "namenode" ]
```

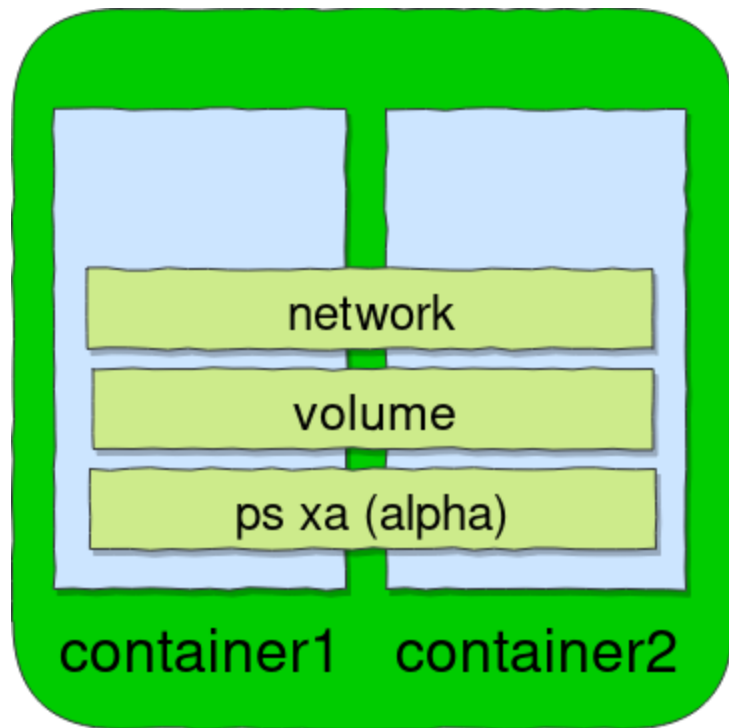







 Pod

Sidecar pattern



 Pod

```
apiVersion: apps/v1beta1
kind: StatefulSet
metadata:
  name: ozone-hdfs-namenode
spec:
  serviceName: ozone2-hdfs-namenode
  replicas: 1
  template:
    metadata:
      labels:
        app: ozone
    annotations:
      prometheus.io/scrape: "true"
      prometheus.io/port: "28942"
  spec:
    shareProcessNamespace: true
    containers:
      - name: hdfs-namenode
        image: flokkar/ozone:2.1.0
        args: ["hdfs", "namenode"]
      - name: jmxpromo
        image: flokkar/jmxpromo-sidecar
```

Less efficient



Configuration management

Source

Preprocessing

On change

configmap
helm
n/a

Provisioning, Scheduling

Multihost support

Scheduling

Cluster definition

Scaling

Multi tenancy

Failover

CNI
kubectl
helm, yaml
yes
namespaces
yes

Network

Intraservice network

DNS

Service discovery


Data locality

Availability of the ports

CNI
statefuset
DNS
no
service/ingress

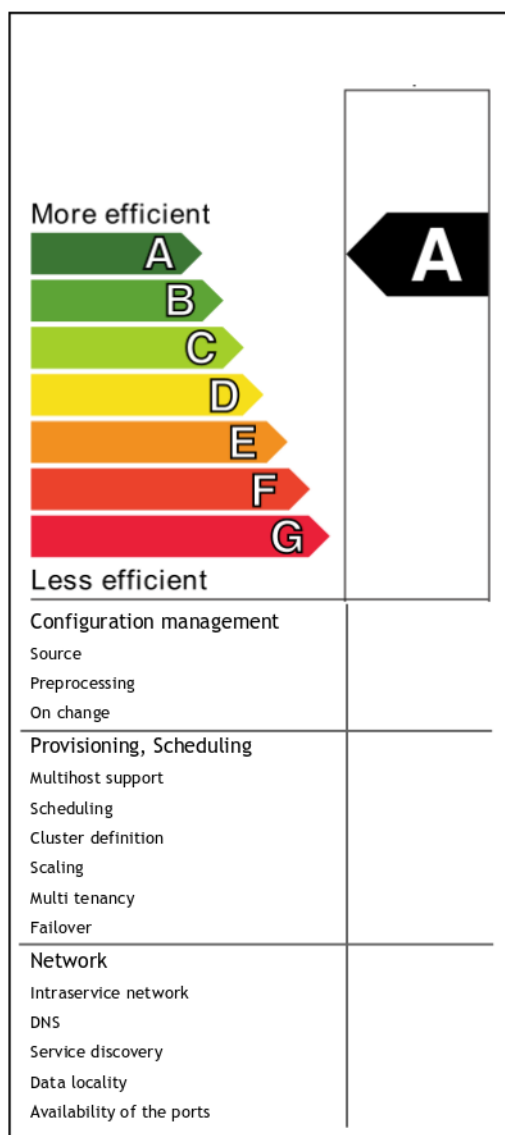
<p>More efficient</p> <p>A</p> <p>B</p> <p>C</p> <p>D</p> <p>E</p> <p>F</p> <p>G</p> <p>Less efficient</p>	
<p>Configuration management</p> <p>Source</p> <p>Preprocessing</p> <p>On change</p>	
<p>Provisioning, Scheduling</p> <p>Multihost support</p> <p>Scheduling</p> <p>Cluster definition</p> <p>Scaling</p> <p>Multi tenancy</p> <p>Failover</p>	
<p>Network</p> <p>Intraservice network</p> <p>DNS</p> <p>Service discovery</p> <p>Data locality</p> <p>Availability of the ports</p>	

Summary

<p>More efficient</p> <p>A</p> <p>B</p> <p>C</p> <p>D</p> <p>E</p> <p>F</p> <p>G</p> <p>Less efficient</p>	
<p>Configuration management</p> <p>Source</p> <p>Preprocessing</p> <p>On change</p>	
<p>Provisioning, Scheduling</p> <p>Multihost support</p> <p>Scheduling</p> <p>Cluster definition</p> <p>Scaling</p> <p>Multi tenancy</p> <p>Failover</p>	
<p>Network</p> <p>Intraservice network</p> <p>DNS</p> <p>Service discovery</p> <p>Data locality</p> <p>Availability of the ports</p>	

Summary

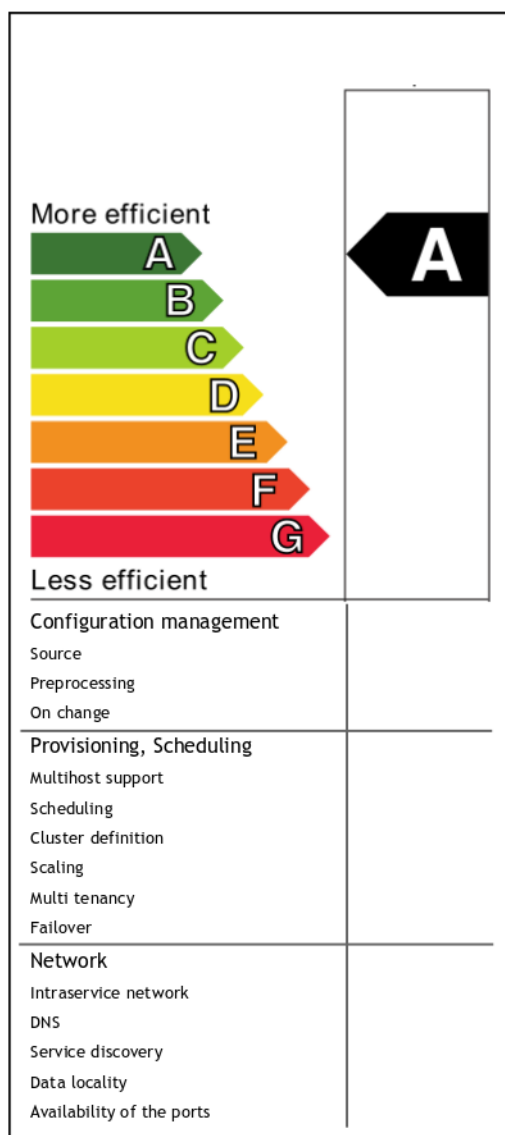
Don't buy without checking the label



Summary

Don't buy without checking the label

Hadoop is first class citizen of cloud-native/containerized environments*



Summary

Don't buy without checking the label

Containerization can help a lot to manage our *Bigdata* clusters

Hadoop is first class citizen of cloud-native/containerized environments*

Q&A

Apache Roadshow EU:
Kubernetes + Hadoop + Ozone
13th June, 14:20

Márton Elek @anzix

<https://flokkr.github.io> (bigdata + containers project)

<https://github.com/flokkr> (source)

elek@apache.org

Image credits



Yan Pritzker (CC)



Carrie Cizauskas (CC)