PT. SYNNEX METRODATA INDONESIA

Methode of Procedure
Rancher Implementation

Version: 2.6.1
Date: 2023-01-16

SUSE | RANCHER

Conducted & Prepared By

PT Synnex Metrodata Indonesia
Jakarta - Indonesia
APL Tower 42nd Floor, Jl. Letjen S. Parman Kav 28, Jakarta Barat 11470

Kontrol Dokumen

Tanda Tangan

	Organisasi	**Nama**	**Tanda Tangan**
Penyusun :	PT SMI	M Ibnu Rusydan	
Kualitas Kontrol :	PT SMI	Ardian Saputra Kurniawan	

Sejarah Rilis

Versi	Tanggal Dibuat	Halaman	Catatan
1.0		Semua	Rilis Pertama

DAFTAR ISI

Kontrol Dokumen **2**
1 Pendahuluan **5**
1.1 Tujuan 5
1.2 Ruang Lingkup 5
2 Daftar Node yang di Implementasi **6**
3 Daftar Product yang di Implementasi **7**
4 Topology **7**
5 Dokumen Referensi **8**
6 Install dan Konfigurasi **9**
6.1 Deploy kubernetes cluster base-on RKE for Rancher Management Server cluster 9
6.1.1 Initial config on all node 9
6.1.2 Install RKE 10
6.1.3 Deploy k8s cluster base-on RKE 10
6.1.4 Install kubectl 11
6.2 Deploy Rancher Management Server for centralized management kubernetes 12
6.3 Deploy downstream k8s cluster from Rancher Management Server 13
6.4 Deploy Longhorn Distribute Block Storage from Rancher Management Server 14
6.4.1 Minimum Recommended Hardware 14
6.4.2 Minimal Available Storage and Over-provisioning 14
6.4.3 Node Preparation 15
6.4.4 Deploy Longhorn 16
6.5 Deploy Load Balancer on downstream k8s cluster from Rancher Management Server 20
6.6 Deploy Monitoring for downstream k8s cluster from Rancher Management Server 21
6.7 Deploy Wordpress on downstream k8s cluster from Rancher Management Server 22
6.8 Deploy Rancher Backups from Rancher Management Server 24
6.9 Backup & Restore Persistent Storage application with Longhorn 25
6.9.1 How to Backup 25
6.9.2 How to Restore 26
6.10 Disaster Recovery 29
6.10.1 Concenpt Section 29
6.10.2 Requiment 30
6.10.3 Kondisi Existing 30
6.10.4 Use Case 30
6.10.5 Operation 31

1 Pendahuluan

1.1 Tujuan

Tujuan dari dokumen ini adalah untuk men-demo kan solusi terkait kebutuhan kubernetes environment dari SUSE Rancher.

Dokumen ini mencakup Methode of Procedure Implementation for Docker container engine, Rancher kubernetes engine, Rancher management server & Longhorn block storage on Suse Linux Enterprise Server.

1.2 Ruang Lingkup

Ruang lingkup dari dokumen ini mencakup:

- Topology
- Dokumen Referensi
- Deploy kubernetes cluster base-on RKE for Rancher Management Server cluster
- Deploy Rancher Management Server for centralized management kubernetes
- Deploy downstream k8s cluster from Rancher Management Server
- Deploy Longhorn Distribute Block Storage from Rancher Management Server
- Deploy Load Balancer on downstream k8s cluster from Rancher Management Server
- Deploy Monitoring for downstream k8s cluster from Rancher Management Server
- Deploy Wordpress on downstream k8s cluster from Rancher Management Server
- Deploy Rancher Backups from Rancher Management Server
- Backup & Restore Persistent Storage application with Longhorn
- Disaster Recovery

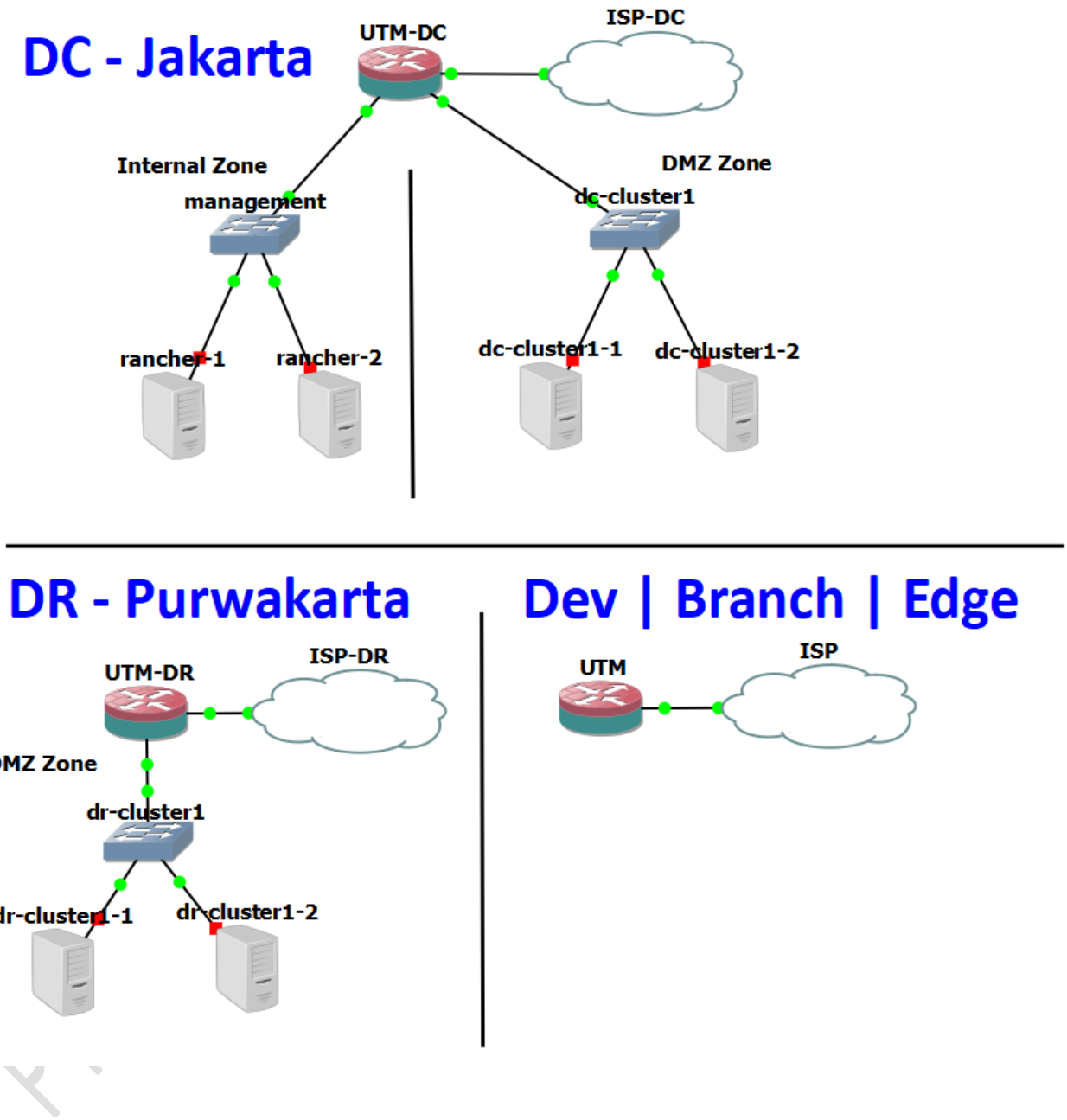
2 Daftar node yang di Implementasi

No	Node Name	Specification	Role	Description
1	rancher-1	cpu: 2vcpu, ram: 4gb, disk: 20gb	Controlplane, etcd, worker	RMS cluster node
2	rancher-2	cpu: 2vcpu, ram: 4gb, disk: 20gb	Controlplane, etcd, worker	RMS cluster node
3	dc-cluster1-1	cpu: 6vcpu, ram: 6gb, disk0-OS: 20gb, disk1-Longhorn: 5gb	Controlplane, etcd, worker	Downstream k8s dc-cluster1
4	dc-cluster1-2	cpu: 6vcpu, ram: 6gb, disk0-OS: 20gb, disk1-Longhorn: 5gb	Controlplane, etcd, worker	Downstream k8s dc-cluster1
5	dr-cluster1-1	cpu: 6vcpu, ram: 6gb, disk0-OS: 20gb, disk1-Longhorn: 5gb	Controlplane, etcd, worker	Downstream k8s dr-cluster1
6	dr-cluster1-2	cpu: 6vcpu, ram: 6gb, disk0-OS: 20gb, disk1-Longhorn: 5gb	Controlplane, etcd, worker	Downstream k8s dr-cluster1

3 Daftar Product yang di Implementasi

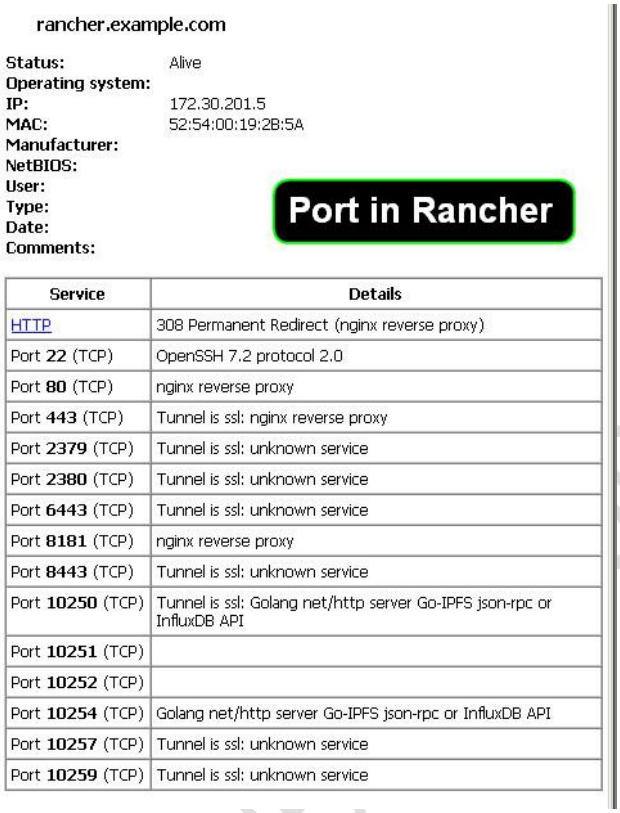
No	Product	Version
1	Suse Linux Enterprise Server	15 SP3
2	Docker Container Engine	20.10.9-ce
3	Rancher Kubernetes Engine	v1.21.7
4	Longhorn Distribute Block Storage	v1.2.3
5	Rancher Management Server	2.6.1

4 Topology


DC - Jakarta
UTM-DC
ISP-DC
Internal Zone
management
DMZ Zone
dc-cluster1
rancher-1
rancher-2
dc-cluster1-1
dc-cluster1-2
DR - Purwakarta
ISP-DR
UTM-DR
DMZ Zone
dr-cluster1
dr-cluster1-1
dr-cluster1-2
Dev | Branch | Edge
UTM
ISP

5 Dokumen Referensi

- Rancher Docs : https://ranchermanager.docs.rancher.com/v2.6
- SUSE Rancher support matrix : https://www.suse.com/suse-rancher/support-matrix/all-supported-versions/rancher-v2-6-1/
- Port Requiment : https://ranchermanager.docs.rancher.com/v2.6/getting-started/installation-and-upgrade/installation-requirements/port-requirements#downstream-kubernetes-cluster-nodes

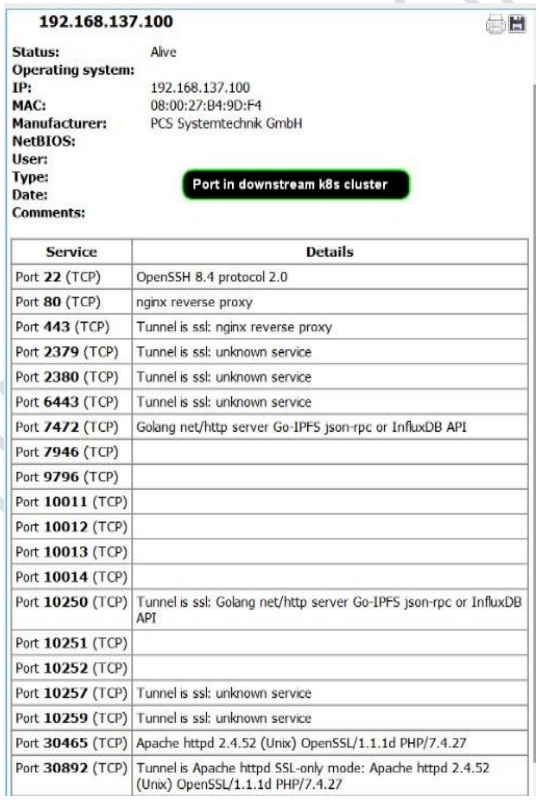


rancher.example.com

Status: Alive
Operating system:
IP: 172.30.201.5
MAC: 52:54:00:19:2B:5A
Manufacturer:
NetBIOS:
User:
Type:
Date:
Comments:

Port in Rancher

Service	Details
HTTP	308 Permanent Redirect (nginx reverse proxy)
Port **22** (TCP)	OpenSSH 7.2 protocol 2.0
Port **80** (TCP)	nginx reverse proxy
Port **443** (TCP)	Tunnel is ssl: nginx reverse proxy
Port **2379** (TCP)	Tunnel is ssl: unknown service
Port **2380** (TCP)	Tunnel is ssl: unknown service
Port **6443** (TCP)	Tunnel is ssl: unknown service
Port **8181** (TCP)	nginx reverse proxy
Port **8443** (TCP)	Tunnel is ssl: unknown service
Port **10250** (TCP)	Tunnel is ssl: Golang net/http server Go-IPFS json-rpc or InfluxDB API
Port **10251** (TCP)	
Port **10252** (TCP)	
Port **10254** (TCP)	Golang net/http server Go-IPFS json-rpc or InfluxDB API
Port **10257** (TCP)	Tunnel is ssl: unknown service
Port **10259** (TCP)	Tunnel is ssl: unknown service



192.168.137.100

Status: Alive
Operating system:
IP: 192.168.137.100
MAC: 08:00:27:B4:9D:F4
Manufacturer: PCS Systemtechnik GmbH
NetBIOS:
User:
Type:
Date:
Comments:

Port in downstream k8s cluster

Service	Details
Port **22** (TCP)	OpenSSH 8.4 protocol 2.0
Port **80** (TCP)	nginx reverse proxy
Port **443** (TCP)	Tunnel is ssl: nginx reverse proxy
Port **2379** (TCP)	Tunnel is ssl: unknown service
Port **2380** (TCP)	Tunnel is ssl: unknown service
Port **6443** (TCP)	Tunnel is ssl: unknown service
Port **7472** (TCP)	Golang net/http server Go-IPFS json-rpc or InfluxDB API
Port **7946** (TCP)	
Port **9796** (TCP)	
Port **10011** (TCP)	
Port **10012** (TCP)	
Port **10013** (TCP)	
Port **10014** (TCP)	
Port **10250** (TCP)	Tunnel is ssl: Golang net/http server Go-IPFS json-rpc or InfluxDB API
Port **10251** (TCP)	
Port **10252** (TCP)	
Port **10257** (TCP)	Tunnel is ssl: unknown service
Port **10259** (TCP)	Tunnel is ssl: unknown service
Port **30465** (TCP)	Apache httpd 2.4.52 (Unix) OpenSSL/1.1.1d PHP/7.4.27
Port **30892** (TCP)	Tunnel is Apache httpd SSL-only mode: Apache httpd 2.4.52 (Unix) OpenSSL/1.1.1d PHP/7.4.27

- Rancher Administration Docs (*Permit for Internal or Partner Use*) :
  https://drive.google.com/file/d/12KU6Jys4aPOaQKR1Yz49Z3d3p1usilUY/view?usp=sharing
  https://drive.google.com/file/d/1_QGLstVklytDVJ5hmrNtBo9EEWDi_FIU/view?usp=sharing

- Longhorn Docs : https://longhorn.io/docs/
- RKE download : https://github.com/rancher/rke/releases/
- Rancher Management Server download :
  if use kubernetes, please use package source with helm repo : https://releases.rancher.com/server-charts/stable
  if use docker engine, please use this package : https://github.com/rancher/rancher

6 Install dan Konfigurasi

6.1 Deploy kubernetes cluster base-on RKE for Rancher Management Server cluster

6.1.1 Initial config on all node

#-- All node, example use **SUSE Linux Enterprise Server**
LVM disk | no SWAP | Enable root user | Enable ssh | SSH user in docker group
systemctl stop firewalld
systemctl disable firewalld

SSH config
vi /etc/ssh/sshd_config
AllowTcpForwarding yes

activate RegCode
Request active code subscription *bisa trial mode, 60 hari
activasi di sisi OS :
SUSEConnect -r xxxxxxxxxxxx -e user@email.com & config NTP

All node
vi /etc/hosts

ip-RMS/rancher-1	rancher.example.com	rancher	#to resolve RMS name
ip-rancher-1	rancher-1.example.com	rancher-1	#to resolve host self name

dst

All node
Query :
1. Generate SSH (all node)
ssh-keygen

2. Copy keygen SSH antar server (di fungsikan agar ketika ssh, tidak perlu credentials)
ssh-copy-id -i ~/.ssh/id_rsa.pub <ip_localhost> dan <ip_server_tujuan>

All node
Install Docker:
On SLES 15 :
regist your SLES OS
#Add container module from cli
SUSEConnect -p sle-module-containers/15.3/x86_64 -r ''
#or Add container module from yast
yast -> software -> system extensions -> select Containers Module -> next -> accept -> ok -> finish
zypper -n install docker
systemctl start docker
systemctl enable docker
systemctl status docker

6.1.2 Install RKE

#-- Installing RKE | run on rancher-1 & rancher-2
wget https://github.com/rancher/rke/releases/download/v1.2.14/rke_linux-amd64
mv rke_linux-amd64 rke
chmod +x rke
mv rke /usr/local/bin/
rke --version

6.1.3 Deploy k8s cluster base-on RKE

#-- Create the cluter configuration File | run on rancher-1
rke config
sesuaikan konfigurasi nya

```
rancher-1:~ # rke config
[+] Cluster Level SSH Private Key Path [~/.ssh/id_rsa]:
[+] Number of Hosts [1]: 2
[+] SSH Address of host (1) [none]: 172.30.201.5
[+] SSH Port of host (1) [22]:
[+] SSH Private Key Path of host (172.30.201.5) [none]:
[-] You have entered empty SSH key path, trying fetch from SSH key parameter
[+] SSH Private Key of host (172.30.201.5) [none]:
[-] You have entered empty SSH key, defaulting to cluster level SSH key: ~/.ssh/id_rsa
[+] SSH User of host (172.30.201.5) [ubuntu]: root
[+] Is host (172.30.201.5) a Control Plane host (y/n)? [y]: y
[+] Is host (172.30.201.5) a Worker host (y/n)? [n]: y
[+] Is host (172.30.201.5) an etcd host (y/n)? [n]: y
[+] Override Hostname of host (172.30.201.5) [none]: rancher-1
[+] Internal IP of host (172.30.201.5) [none]:
[+] Docker socket path on host (172.30.201.5) [/var/run/docker.sock]:
[+] SSH Address of host (2) [none]: 172.30.201.6
[+] SSH Port of host (2) [22]:
[+] SSH Private Key Path of host (172.30.201.6) [none]:
[-] You have entered empty SSH key path, trying fetch from SSH key parameter
[+] SSH Private Key of host (172.30.201.6) [none]:
[-] You have entered empty SSH key, defaulting to cluster level SSH key: ~/.ssh/id_rsa
[+] SSH User of host (172.30.201.6) [ubuntu]: root
[+] Is host (172.30.201.6) a Control Plane host (y/n)? [y]: y
[+] Is host (172.30.201.6) a Worker host (y/n)? [n]: y
[+] Is host (172.30.201.6) an etcd host (y/n)? [n]: y
[+] Override Hostname of host (172.30.201.6) [none]: rancher-2
[+] Internal IP of host (172.30.201.6) [none]:
[+] Docker socket path on host (172.30.201.6) [/var/run/docker.sock]:
[+] Network Plugin Type (flannel, calico, weave, canal, aci) [canal]:
[+] Authentication Strategy [x509]:
[+] Authorization Mode (rbac, none) [rbac]:
[+] Kubernetes Docker image [rancher/hyperkube:v1.20.12-rancher1]:
[+] Cluster domain [cluster.local]: example.com
[+] Service Cluster IP Range [10.43.0.0/16]:
[+] Enable PodSecurityPolicy [n]:
[+] Cluster Network CIDR [10.42.0.0/16]:
```

#-- Deploy k8s with RKE
rke up
or
rke up --ignore-docker-version

6.1.4 Install kubectl

```
#-- Install Kubectl | run on rancher-1
zypper -n install kubernetes1.18-client

#-- Verify k8s cluster
root@poc00rke00:~# kubectl --kubeconfig kube_config_cluster.yml get node
NAME        STATUS   ROLES                  AGE   VERSION
poc00rke00   Ready    controlplane,etcd,worker   30m   v1.17.4

#-- Setting kube-config
cd /root
cp kube_config_cluster.yml .kube/kube-rke
cat > kube-rke.sh <<EOF
export KUBECONFIG=/root/.kube/kube-rke
EOF
chmod +x kube-rke.sh
source kube-rke.sh
kubectl get node
```

6.2 Deploy Rancher Management Server for centralized management kubernetes

```
## Run on rancher-1
#-- Install Helm
sudo wget -O helm.tar.gz https://get.helm.sh/helm-v3.4.0-linux-amd64.tar.gz
sudo tar -zxf helm.tar.gz
sudo mv linux-amd64/helm /usr/local/bin/helm
sudo chmod +x /usr/local/bin/helm
sudo rm -rf linux-amd64
sudo rm -f helm.tar.gz
helm version --client

#-- Installing Rancher with Helm
# Install Cert Manager
kubectl create namespace cert-manager
kubectl apply --validate=false -f https://github.com/jetstack/cert-manager/releases/download/v0.15.0/cert-manager.crds.yaml
helm repo add jetstack https://charts.jetstack.io
helm repo update
helm install \
cert-manager jetstack/cert-manager \
--namespace cert-manager \
--version v0.15.0
kubectl get all -n cert-manager    #pastikan smw pod ter-create terlebih dahulu
kubectl -n cert-manager rollout status deploy/cert-manager
kubectl -n cert-manager rollout status deploy/cert-manager-webhook

# Add Rancher package
helm repo add rancher-stable https://releases.rancher.com/server-charts/stable
helm repo update

kubectl create namespace cattle-system

# Install rancher
helm install rancher rancher-stable/rancher \
--namespace cattle-system \
--set hostname=rancher.example.com \
--version v2.6.1 \
--set replicas=2
*sesuaikan hostname untuk RMS nya dan sesuaikan replicas nya, contoh disini di set 2, karna menggunakan 2 node

# Verify rancher ready
while true; do curl -kv https://rancher.example.com 2>&1 | grep -q "dynamiclistener-ca"; if [ $? != 0 ]; then echo "Rancher isn't ready yet"; sleep 5; continue; fi; break; done; echo "Rancher is Ready";
```

Pastikan pod di cert-manager & cattle-system running/completed

```
rancher-1:~ # kubectl get pod -n cert-manager
NAME                                       READY   STATUS    RESTARTS   AGE
cert-manager-86b8b4f4b7-sk4mp              1/1     Running   0          10m
cert-manager-cainjector-7f6686b94-g74vm    1/1     Running   0          10m
cert-manager-webhook-66d786db8c-wtbmc      1/1     Running   0          10m
rancher-1:~ # kubectl get pod -n cattle-system
NAME                                READY   STATUS      RESTARTS   AGE
helm-operation-g7j7v                0/2     Completed   0          2m56s
helm-operation-m6vkl                0/2     Completed   0          2m28s
helm-operation-nj66h                0/2     Completed   0          3m59s
helm-operation-plqmm                0/2     Completed   0          2m6s
helm-operation-xjs46                0/2     Completed   0          2m40s
rancher-7546f84756-gmwkp            1/1     Running     0          5m57s
rancher-7546f84756-lq6f9            1/1     Running     0          5m56s
rancher-webhook-6979fbd4bf-kfm7q    1/1     Running     0          2m19s
rancher-1:~ #
```

kubectl -n cattle-system rollout status deploy/rancher

Set A record atau dns di sisi laptop/client/lan :

ip-rancher-1	rancher.example.com
ip-rancher-2	rancher.example.com

access in web, https://rancher.example.com

6.3 Deploy downstream k8s cluster from Rancher Management Server

6.4 Deploy Longhorn Distribute Block Storage from Rancher Management Server

6.4.1 Minimum Recommended Hardware

- 3 nodes
- 4 vcpu per node
- 4 gb ram per node
- SSD/NVME or similar performance block device on the node for storage. We don't recommend using spinning disks with Longhorn, due to low IOPS.

6.4.2 Minimal Available Storage and Over-provisioning

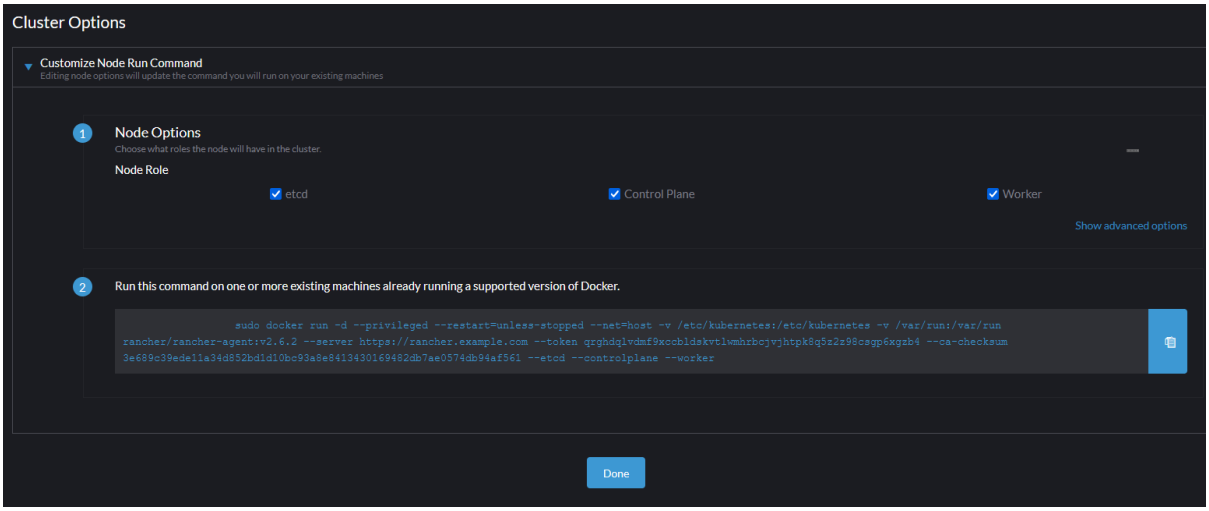
If you need to use the **root disk**, use the default minimal available storage percentage **setup which is 25%**, and set overprovisioning percentage to 200% to minimize the chance of DiskPressure.

If you're using a **dedicated disk for Longhorn**, you can lower the setting minimal available **storage percentage to 10%**.

For the Over-provisioning percentage, it depends on how much space your volume uses on average. For example, if your workload only uses half of the available volume size, you can set the Over-provisioning percentage to 200, which means Longhorn will consider the disk to have twice the schedulable size as its full size minus the reserved space.

Go to Rancher Management Server -> cluster management -> create -> Use existing nodes and create a cluster using RKE -> insert cluster name & Leave other settings as the defaults -> next.

Sesuaikan role dan follow instruksi nya



Langkah ini di gunakan untuk men-deploy dc-cluster1 dan dr-cluster1

Disk Space Management - Since Longhorn doesn't currently support sharding between the different disks, **we recommend using LVM to aggregate all the disks for Longhorn into a single partition**, so it can be easily extended in the future.

6.4.3 Node Preparation

```
## Run on dc-cluster1-1, dc-cluster1-2, dr-cluster1-1, dr-cluster1-2
# General info
default path : /var/lib/longhorn/
default disk stote : create partition with lvm
file system support : ext4, xfs, btrfs

# Make sure mandatory package
bash --version
curl --version
findmnt --version
grep --version
awk --version
blkid --version

# Install open-iscsi
zypper install open-iscsi
iscsid --version
systemctl start iscsid
systemctl enable iscsid
systemctl status iscsid

# Setup disk with LVM
lvmdiskscan
pvcreate /dev/sdb
vgcreate longhornvg /dev/sdb
lvcreate -l 100%FREE -n longhorn longhornvg -y
mkfs.xfs /dev/longhornvg/longhorn
echo "/dev/longhornvg/longhorn          /var/lib/longhorn/          xfs defaults 0 0" >> /etc/fstab
mkdir -p /var/lib/longhorn/
mount -a
df –hT /var/lib/longhorn
```

6.4.4 Deploy Longhorn

Deploy Longhorn

Goto Rancher Web -> Cluster management -> explore on your downstream k8s cluster -> project/namespaces -> create project, ex: Storage (for longhorn system) -> apps & marketplace -> filter (search longhorn) -> install -> install into project: Storage -> next -> follow this instructions.

Sesuaikan point berikut :

Edit options -> Longhorn default settings -> customize default settings

- Storage minimal available percentage
- Default replica count
- Concurrent replica rebuild per node limit

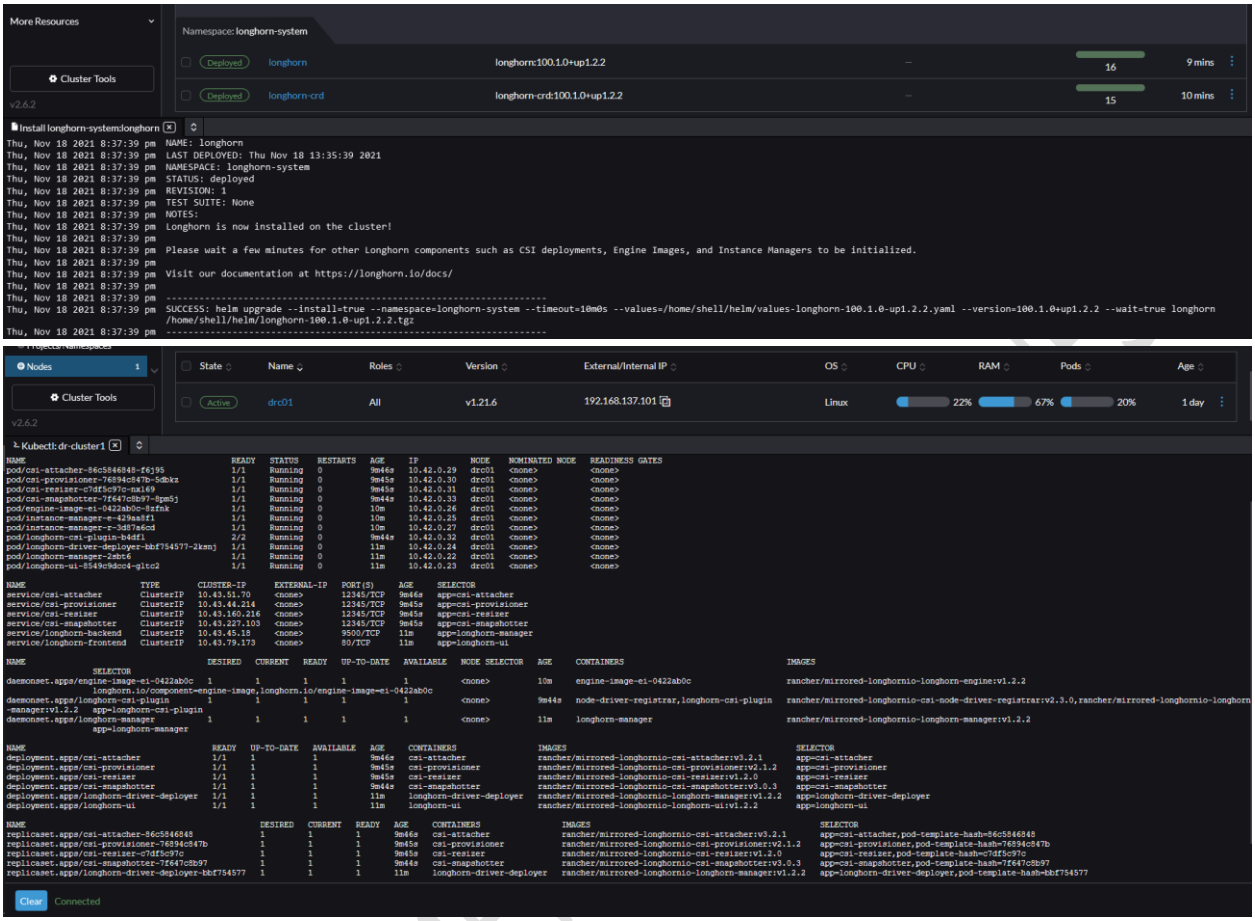
Longhorn CSI Driver Settings

Longhorn Storage Class Settings

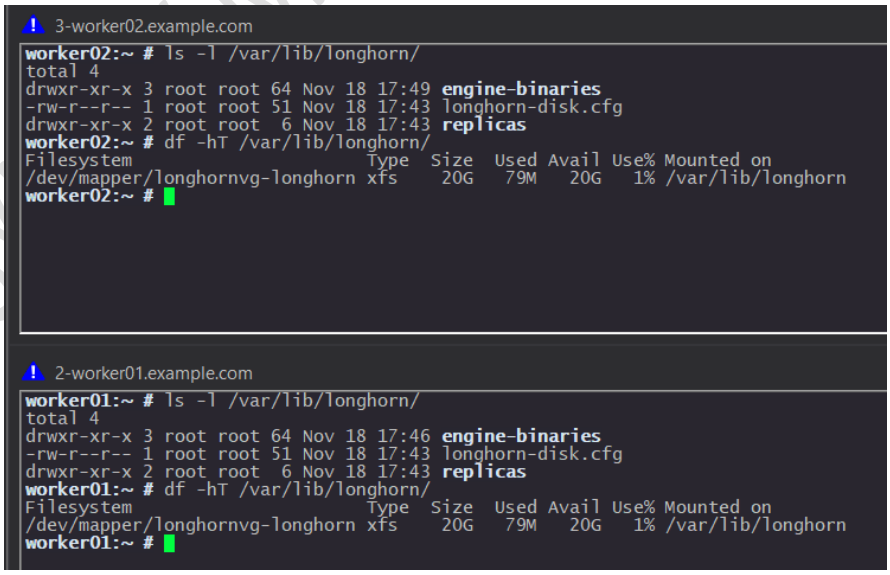
- Default storage class replica count

Install

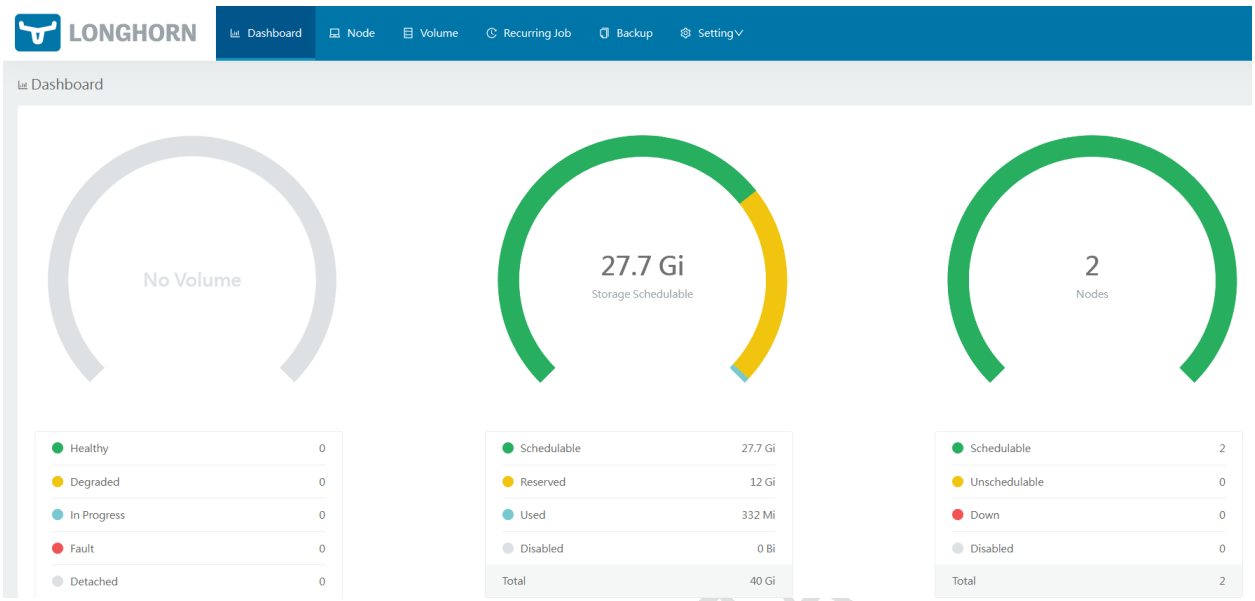
**dalam demo ini menggunakan 2 worker node, jadi replica di set 2, untuk di sebar ke 2 worker node*



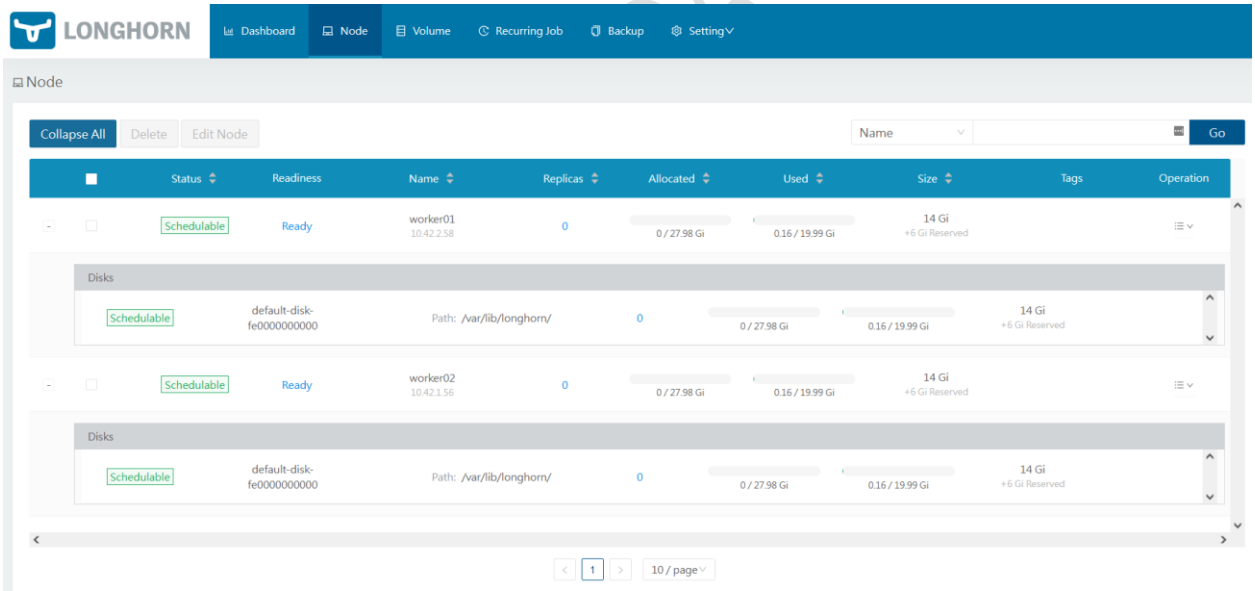
Longhorn installed



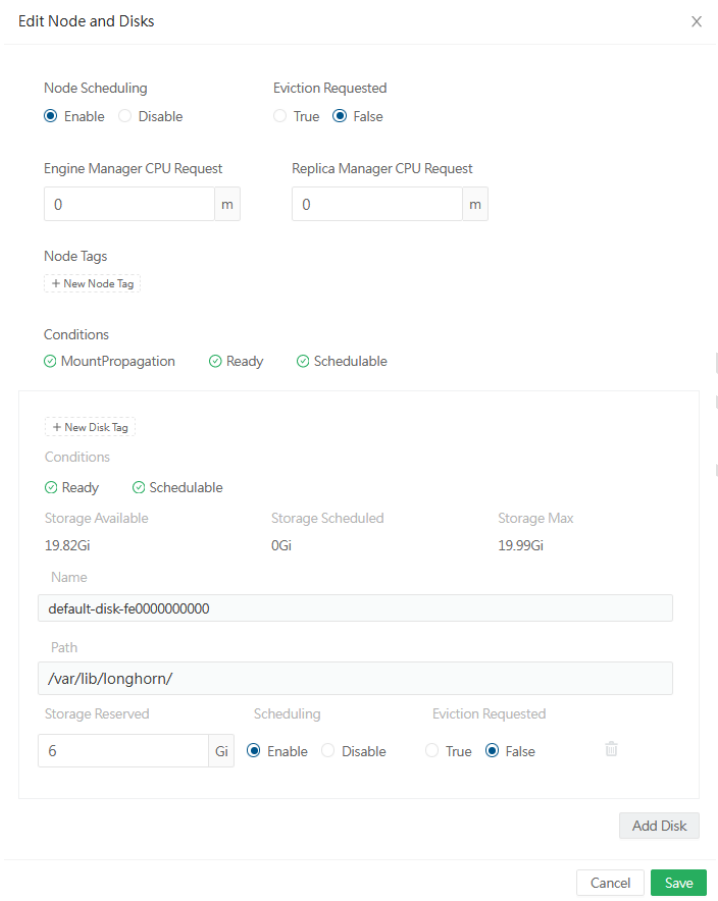
Longhorn on File System node



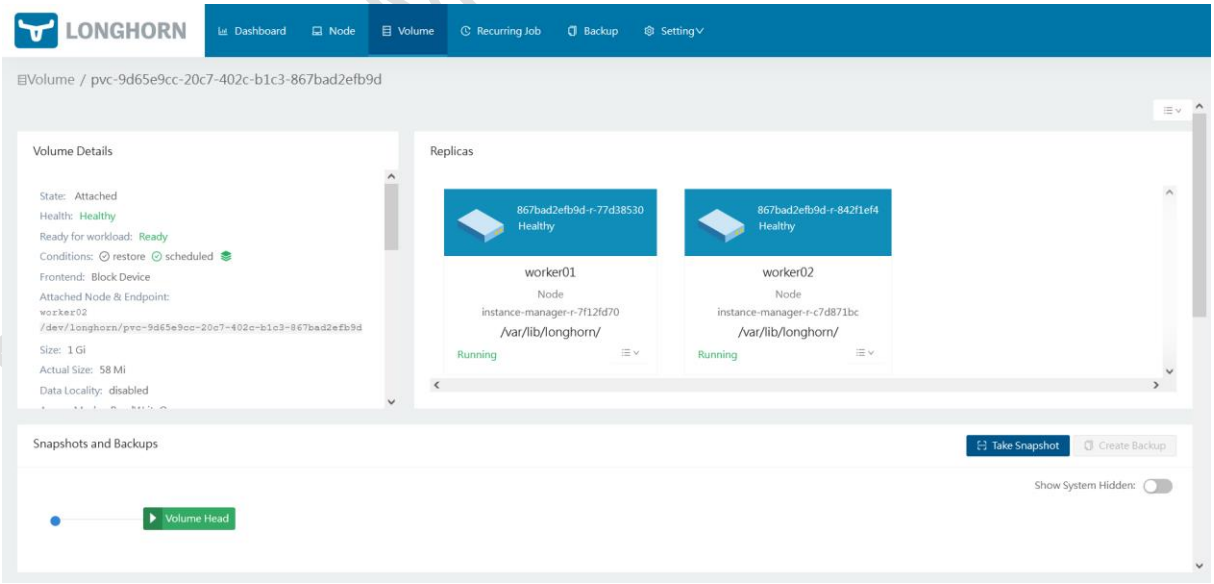
Longhorn UI



Longhorn node menu



Node and disk info



Volume pada longhorn di replicate menjadi 2

6.5 Deploy Load Balancer on downstream k8s cluster from Rancher Management Server

Ingress sudah default ter-install pada k8s cluster base-on RKE. Untuk load balancer masuk ke dalam optional install. Secara best practice kami merekomendasikan load balancer dari enterprise product seperti F5/Citrix/etc, dalam demo ini kami menggunakan MetalLB untuk keperluan load balancer nya.

Add repository

Goto Rancher Web -> Cluster management -> explore on your downstream k8s cluster -> apps & marketplace -> repositories -> create -> follow this instructions.

Name : bitnami

Index URL : https://charts.bitnami.com/bitnami

Deploy MetalLB

Create namespace with metallb-system on Project System

Go to apps & marketplace -> charts -> metallb -> namespace: metallb-system -> next -> install

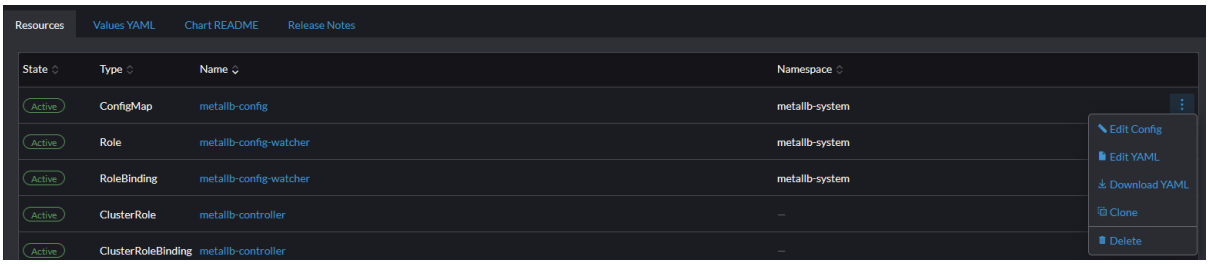
Make sure metallb already deployed via CLI

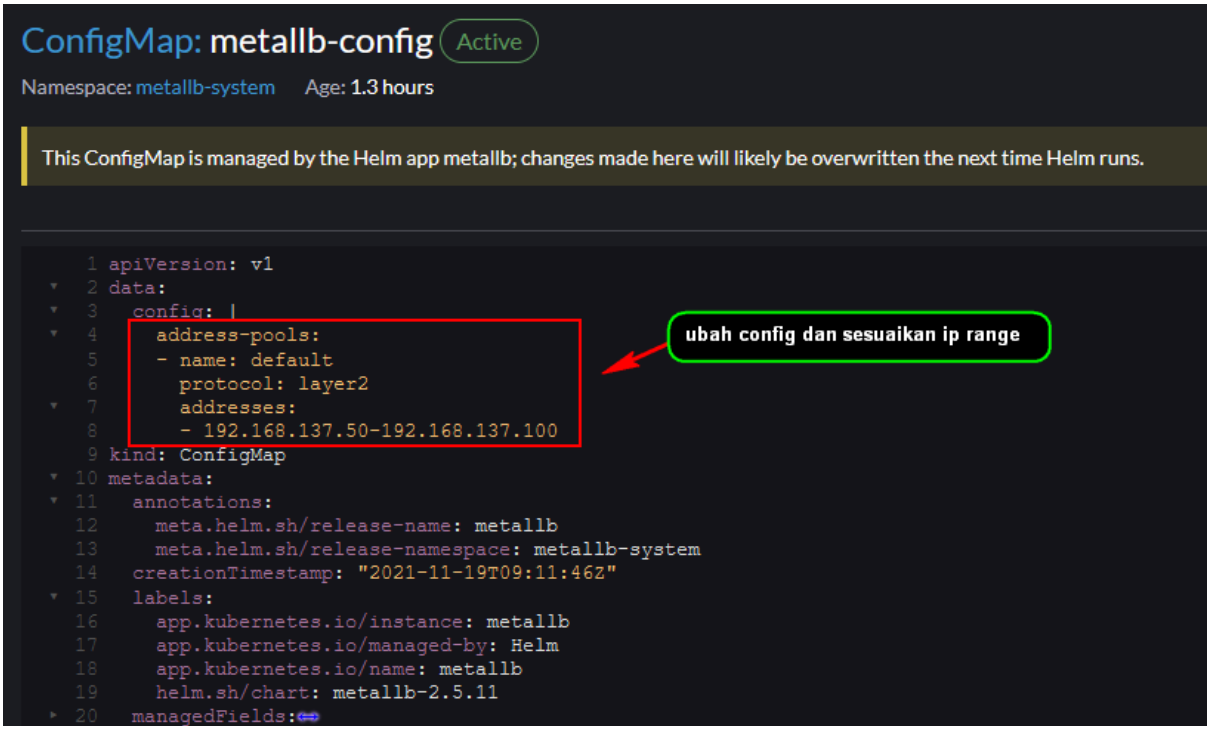
kubectl get all -n metallb-system

Config IP Pool for Load Balancer

Go to apps & marketplace -> installed apps -> metallb

Edit yaml on ConfigMap -> save





6.6 Deploy Monitoring for downstream k8s cluster from Rancher Management Server

Cluster management -> explore on your downstream k8s cluster -> project/namespaces -> create project, ex: Monitoring -> apps & marketplace -> filter (search monitoring) -> install -> install into project Monitoring -> next -> follow this instructions.

Sesuaikan point berikut :

Edit options -> Prometheus

- Retention Size
- Check list Persistent Storage for Prometheus
  - Size
- Storage Class Name: longhorn

Install

6.7 Deploy Wordpress on downstream k8s cluster from Rancher Management Server

Deploy Wordpress

Cluster management -> explore on your downstream k8s cluster -> project/namespaces -> create project, ex: TokonyaKita -> create namespace: tokonyakita-ui -> apps & marketplace -> filter (search wordpress) -> install -> install into namespace: tokonyakita-ui -> next -> follow this instructions.

Sesuaikan point berikut :

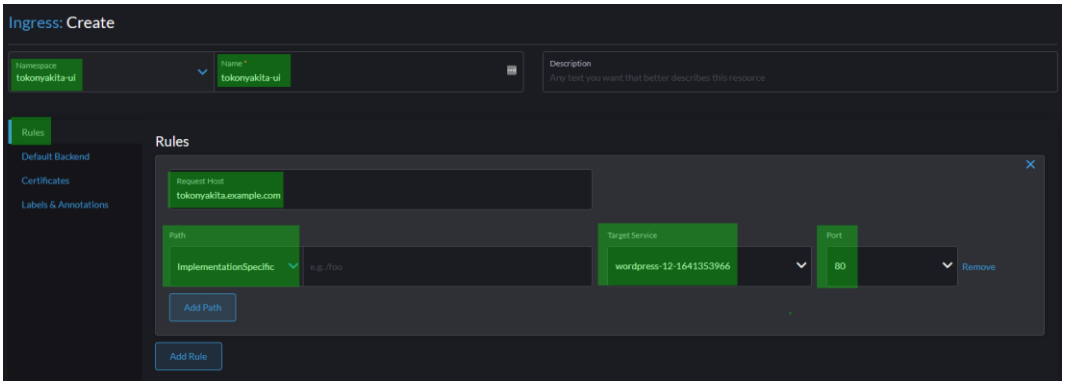
Edit YAML & search this point :

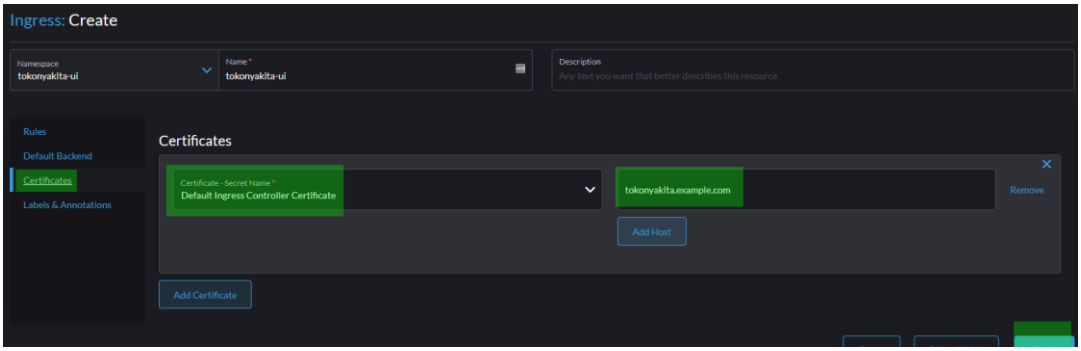
```
wordpressUsername          #for wordpress admin use: admin
wordpressPassword          #for wordpress admin password: linux
wordpressBlogName          #for your apps name: TokonyaKita
10Gi                       #this size for apps: 1Gi
8Gi                        #this size for database: 1Gi
database: bitnami_wordpress
  existingSecret: ''       #this secret db: linux
  host: localhost
  password: ''             #this secret db: linux
  port: 3306
  user: bn_wordpress

database: bitnami_wordpress
  password: ''             #this secret db: linux
  rootPassword: ''         #this secret db: linux
  username: bn_wordpress
storageClass: ''           #gunakan storageClass untuk persistent storage nya
```

Install

Ingress config

Cluster management -> explore on your downstream k8s cluster -> Service Discovery -> Ingress -> Create -> follow this picture -> if already, klik create.

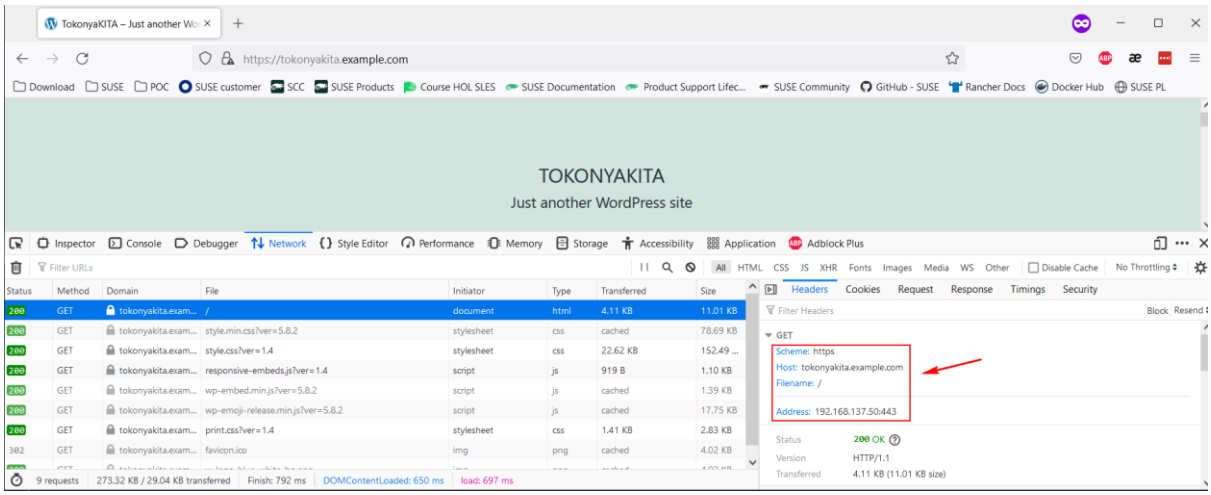




Example workflow

Apps <-> svc-loadbalancer cluster-ip <-> svc-loadbalancer external-ip <-> ingress <-> user access

```
masterO1:~ # kubectl get all -n tokonyakita-ui
NAME                                          READY   STATUS    RESTARTS   AGE
pod/tokonyakita-ui-mariadb-0                  1/1     Running   0          4m56s
pod/tokonyakita-ui-wordpress-58f54d5765-2thn9 0/1     Running   0          4m56s

NAME                                  TYPE           CLUSTER-IP      EXTERNAL-IP      PORT(S)                      AGE
service/tokonyakita-ui-mariadb        ClusterIP      10.43.204.80    <none>           3306/TCP                     4m56s
service/tokonyakita-ui-wordpress      LoadBalancer   10.43.103.145   192.168.137.50   80:31008/TCP,443:30273/TCP   4m56s

NAME                                     READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/tokonyakita-ui-wordpress 0/1     1            0           4m56s

NAME                                                DESIRED   CURRENT   READY   AGE
replicaset.apps/tokonyakita-ui-wordpress-58f54d5765 1         1         0       4m56s

NAME                                    READY   AGE
statefulset.apps/tokonyakita-ui-mariadb 1/1     4m56s
masterO1:~ # kubectl get services -o wide --all-namespaces | grep --color=never -E 'LoadBalancer|NAMESPACE'
NAMESPACE        NAME                      TYPE           CLUSTER-IP      EXTERNAL-IP      PORT(S)                      AGE   SELECTOR
tokonyakita-ui   tokonyakita-ui-wordpress  LoadBalancer   10.43.103.145   192.168.137.50   80:31008/TCP,443:30273/TCP   12m   app.kubernetes.io/instance=tokonyakita-ui,app.kubernetes.io/name
=wordpress
masterO1:~ # kubectl get ingress -A
NAMESPACE        NAME             CLASS    HOSTS                   ADDRESS                                  PORTS     AGE
cattle-system    rancher          <none>   rancher.example.com     192.168.137.21,192.168.137.22            80, 443   2d2h
tokonyakita-ui   tokonyakita-ui   <none>   tokonyakita.example.com 192.168.137.21,192.168.137.22            80, 443   4m53s
masterO1:~ #
```



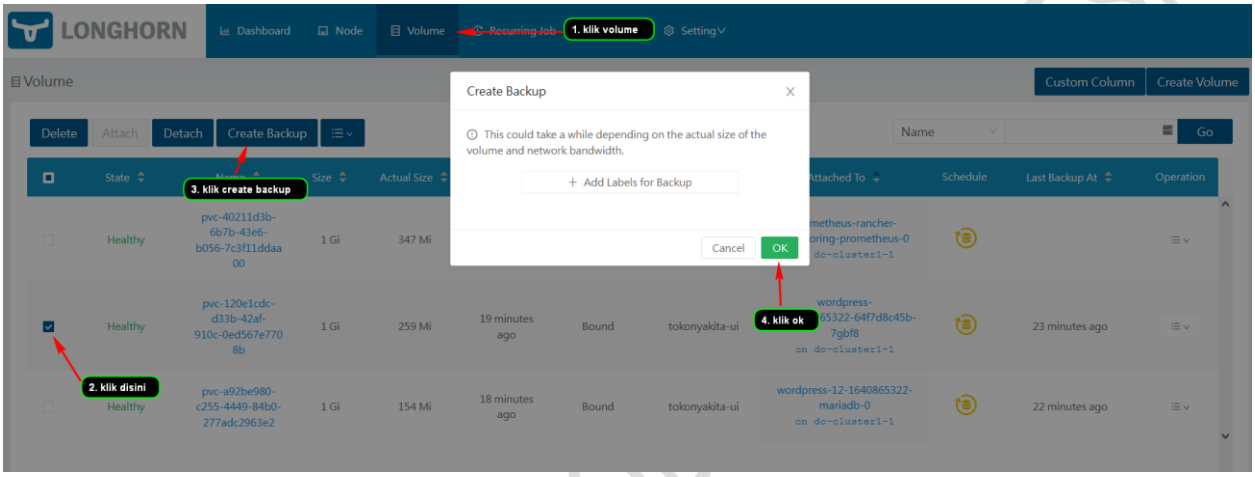
6.8 Deploy Rancher Backups from Rancher Management Server

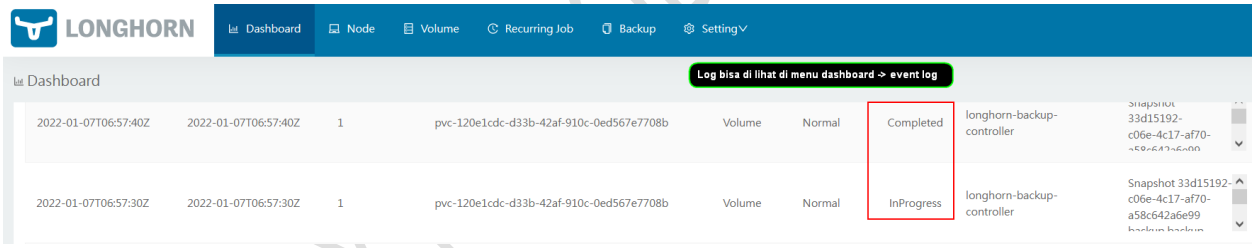
```
## Example we use local storage with persistent volume
# Create persistent volume
mkdir -p /persistentstorage
cat > pv-rancher-backup.yml <<EOF
kind: PersistentVolume
apiVersion: v1
metadata:
  name: rancher-backup                    #name pv
  labels:
    type: local                           #define for local storage type
spec:
  capacity:
    storage: 5Gi                          #size
  accessModes:
    - ReadWriteOnce
  hostPath:
    path: "/persistentstorage"            #folder or path location
EOF
kubectl apply -f pv-rancher-backup.yml

# Deploy Rancher Backups
Cluster management -> explore on your rancher cluster -> project/namespaces -> create project, ex: Backup -> apps & marketplace -> filter (search rancher backups) -> install -> install into project Backup -> next -> follow this instructions.

Sesuaikan point berikut :
Edit options -> Chart Options
        Select: Use an existing persistent volume -> rancher-backup -> install
```

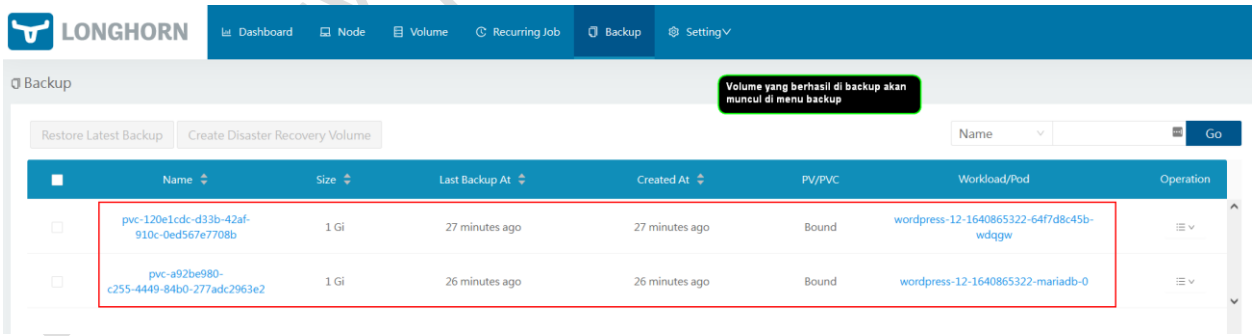
6.9 Backup & Restore Persistent Storage application with Longhorn

Pre-requiment, volume untuk data aplikasi pada longhorn telah di backup. Target backup yang dibutuhkan adalah external storage, bisa dengan NFS maupun S3. Kondisi disini backup target untuk longhorn sudah ready.

6.9.1 How to Backup



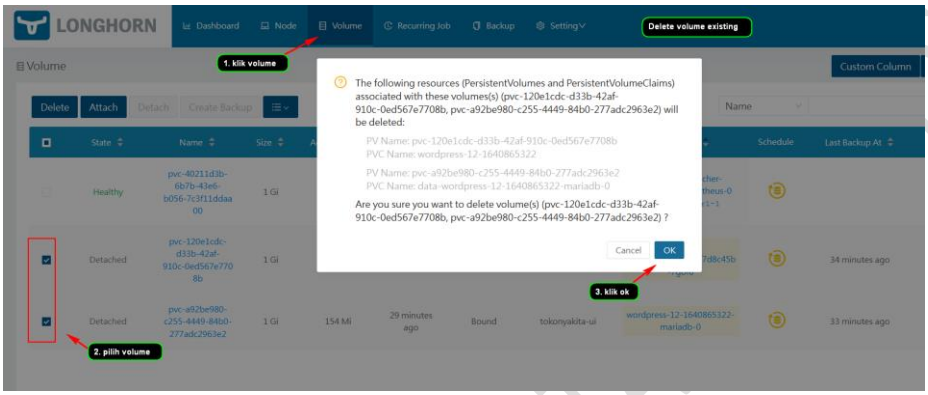


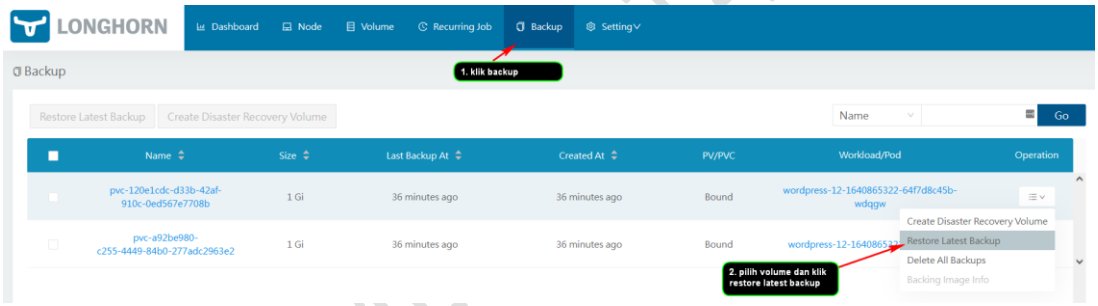


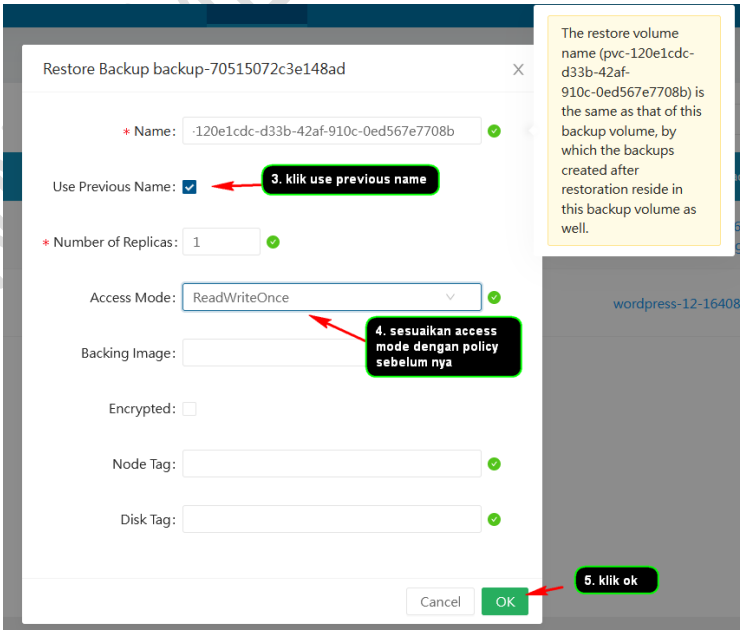
6.9.2 How to Restore

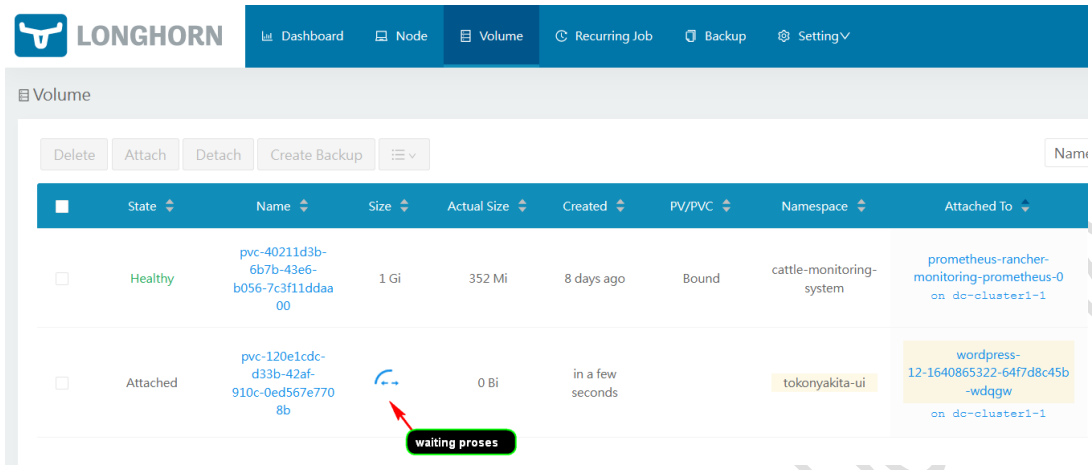
```
rancher-1:~ # kubectl get pod -n tokonyakita-ui
NAME                                    READY   STATUS    RESTARTS   AGE
wordpress-12-1640865322-64f7d8c45b-7gbf8   1/1     Running   2          24m
wordpress-12-1640865322-mariadb-0          1/1     Running   0          19m
rancher-1:~ # kubectl scale deployment wordpress-12-1640865322 --replicas 0 -n tokonyakita-ui
deployment.apps/wordpress-12-1640865322 scaled
rancher-1:~ # kubectl scale statefulset wordpress-12-1640865322-mariadb --replicas 0 -n tokonyakita-ui
statefulset.apps/wordpress-12-1640865322-mariadb scaled
rancher-1:~ #
```

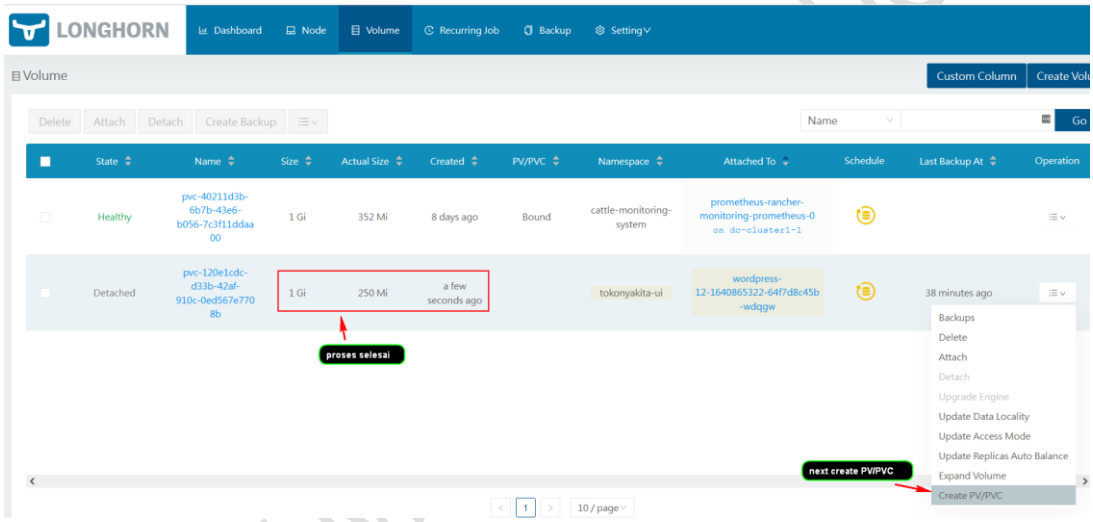
Lakukan scale out pod

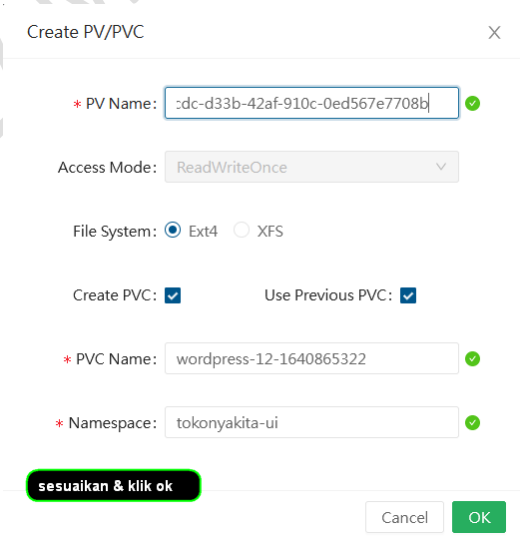


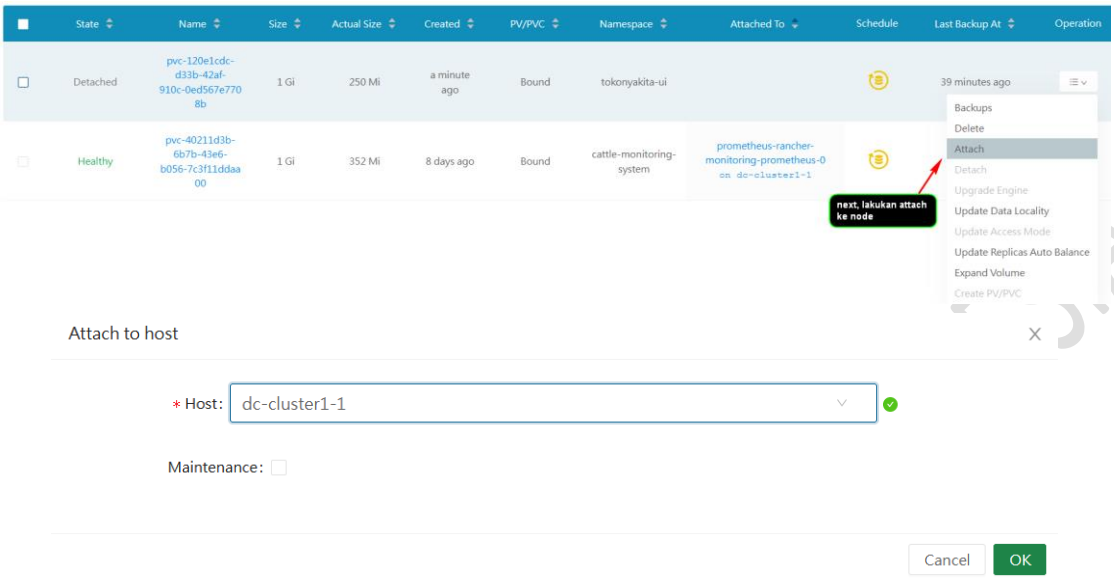




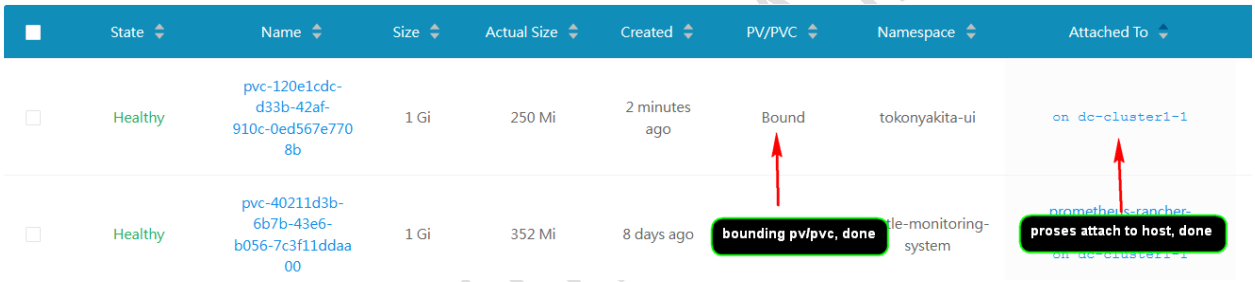

LONGHORN
Dashboard
Node
Volume
Recurring Job
Backup
Setting
Volume
Delete
Attach
Detach
Create Backup
State
Name
Size
Actual Size
Created
PV/PVC
Namespace
Attached To
Healthy
pvc-40211d3b-6b7b-43e6-b056-7c3f11ddaa00
1 Gi
352 Mi
8 days ago
Bound
cattle-monitoring-system
prometheus-rancher-monitoring-prometheus-0
on dc-cluster1-1
Attached
pvc-120e1cdc-d33b-42af-910c-0ed567e7708b
0 Bi
in a few seconds
tokonyakita-ui
wordpress-12-1640865322-64f7d8c45b-wdqgw
on dc-cluster1-1
waiting proses

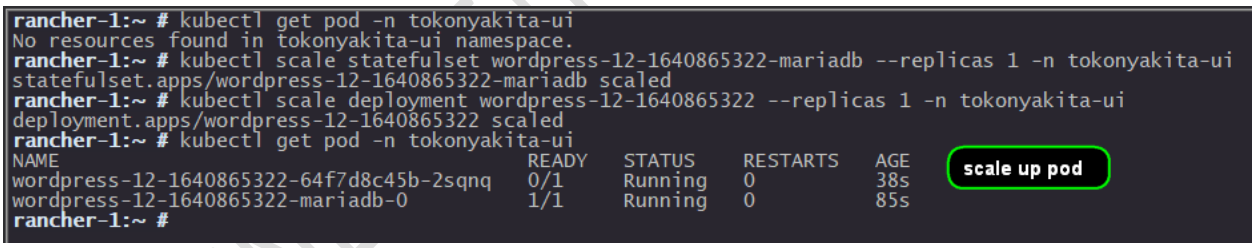

LONGHORN
Dashboard
Node
Volume
Recurring Job
Backup
Setting
Volume
Custom Column
Delete
Attach
Detach
Create Backup
Name
Go
State
Name
Size
Actual Size
Created
PV/PVC
Namespace
Attached To
Schedule
Last Backup At
Operation
Healthy
pvc-40211d3b-6b7b-43e6-b056-7c3f11ddaa00
1 Gi
352 Mi
8 days ago
Bound
cattle-monitoring-system
prometheus-rancher-monitoring-prometheus-0
on dc-cluster1-1
Detached
pvc-120e1cdc-d33b-42af-910c-0ed567e7708b
1 Gi
250 Mi
a few seconds ago
tokonyakita-ui
wordpress-12-1640865322-64f7d8c45b-wdqgw
38 minutes ago
proses selesai
Backups
Delete
Attach
Detach
Upgrade Engine
Update Data Locality
Update Access Mode
Update Replicas Auto Balance
Expand Volume
Create PV/PVC
next create PV/PVC
1
10 / page

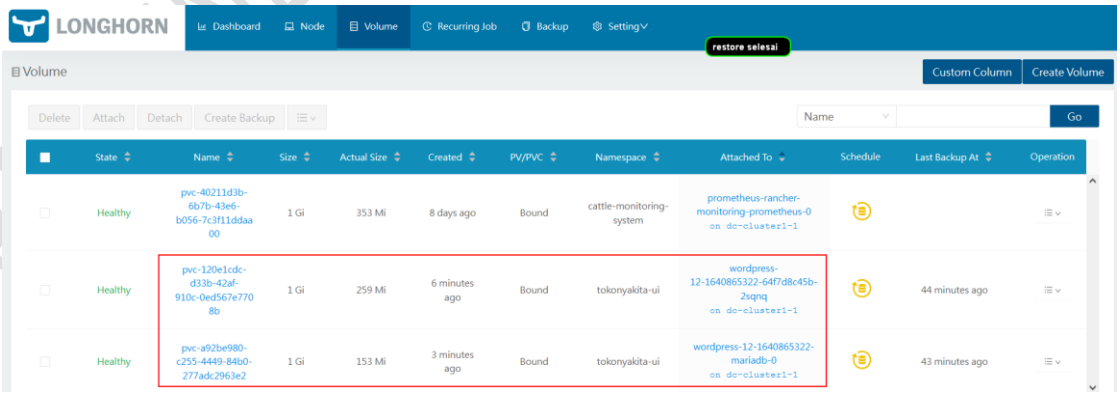

Create PV/PVC
* PV Name: :dc-d33b-42af-910c-0ed567e7708b
Access Mode: ReadWriteOnce
File System: Ext4 XFS
Create PVC:
Use Previous PVC:
* PVC Name: wordpress-12-1640865322
* Namespace: tokonyakita-ui
sesuaikan & klik ok
Cancel
OK


State
Name
Size
Actual Size
Created
PV/PVC
Namespace
Attached To
Schedule
Last Backup At
Operation
Detached
pvc-120e1cdc-d33b-42af-910c-0ed567e7708b
1 Gi
250 Mi
a minute ago
Bound
tokonyakita-ui
39 minutes ago
Healthy
pvc-40211d3b-6b7b-43e6-b056-7c3f11ddaa00
1 Gi
352 Mi
8 days ago
Bound
cattle-monitoring-system
prometheus-rancher-monitoring-prometheus-0
on dc-cluster1-1
next, lakukan attach ke node
Backups
Delete
Attach
Detach
Upgrade Engine
Update Data Locality
Update Access Mode
Update Replicas Auto Balance
Expand Volume
Create PV/PVC

Attach to host
* Host:
dc-cluster1-1
Maintenance:
Cancel
OK


State
Name
Size
Actual Size
Created
PV/PVC
Namespace
Attached To
Healthy
pvc-120e1cdc-d33b-42af-910c-0ed567e7708b
1 Gi
250 Mi
2 minutes ago
Bound
tokonyakita-ui
on dc-cluster1-1
Healthy
pvc-40211d3b-6b7b-43e6-b056-7c3f11ddaa00
1 Gi
352 Mi
8 days ago
bounding pv/pvc, done
system
proses attach to host, done


rancher-1:~ # kubectl get pod -n tokonyakita-ui
No resources found in tokonyakita-ui namespace.
rancher-1:~ # kubectl scale statefulset wordpress-12-1640865322-mariadb --replicas 1 -n tokonyakita-ui
statefulset.apps/wordpress-12-1640865322-mariadb scaled
rancher-1:~ # kubectl scale deployment wordpress-12-1640865322 --replicas 1 -n tokonyakita-ui
deployment.apps/wordpress-12-1640865322 scaled
rancher-1:~ # kubectl get pod -n tokonyakita-ui
NAME READY STATUS RESTARTS AGE
wordpress-12-1640865322-64f7d8c45b-2sqnq 0/1 Running 0 38s
wordpress-12-1640865322-mariadb-0 1/1 Running 0 85s
rancher-1:~ #
scale up pod


LONGHORN
Dashboard
Node
Volume
Recurring Job
Backup
Setting
restore selesai
Volume
Custom Column
Create Volume
Delete
Attach
Detach
Create Backup
Name
Go
State
Name
Size
Actual Size
Created
PV/PVC
Namespace
Attached To
Schedule
Last Backup At
Operation
Healthy
pvc-40211d3b-6b7b-43e6-b056-7c3f11ddaa00
1 Gi
353 Mi
8 days ago
Bound
cattle-monitoring-system
prometheus-rancher-monitoring-prometheus-0
on dc-cluster1-1
Healthy
pvc-120e1cdc-d33b-42af-910c-0ed567e7708b
1 Gi
259 Mi
6 minutes ago
Bound
tokonyakita-ui
wordpress-12-1640865322-64f7d8c45b-2sqnq
on dc-cluster1-1
44 minutes ago
Healthy
pvc-a92be980-c255-4449-84b0-277adc2963e2
1 Gi
153 Mi
3 minutes ago
Bound
tokonyakita-ui
wordpress-12-1640865322-mariadb-0
on dc-cluster1-1
43 minutes ago

6.10 Disaster Recovery

6.10.1 Concept Section

A disaster recovery (DR) volume is a special volume that stores data in a backup cluster in case the whole main cluster goes down. DR volumes are used to increase the resiliency of Longhorn volumes.

Because the main purpose of a DR volume is to restore data from backup, this type of volume doesn't support the following actions before it is activated:

- Creating, deleting, and reverting snapshots
- Creating backups
- Creating persistent volumes
- Creating persistent volume claims

A DR volume can be created from a volume's backup in the backup store. After the DR volume is created, Longhorn will monitor its original backup volume and incrementally restore from the latest backup. A backup volume is an object in the backupstore that contains multiple backups of the same volume.

If the original volume in the main cluster goes down, the DR volume can be immediately activated in the backup cluster, so it can greatly reduce the time needed to restore the data from the backup store to the volume in the backup cluster.

When a DR volume is activated, Longhorn will check the last backup of the original volume. If that backup has not already been restored, the restoration will be started, and the activate action will fail. Users need to wait for the restoration to complete before retrying.

The Backup Target in the Longhorn settings cannot be updated if any DR volumes exist.

After a DR volume is activated, it becomes a normal Longhorn volume and it cannot be deactivated.

6.10.2 Requiment

Di sisi aplikasi :

- Arsitektur, deployment dan credentials di buat sama antara DC-DR

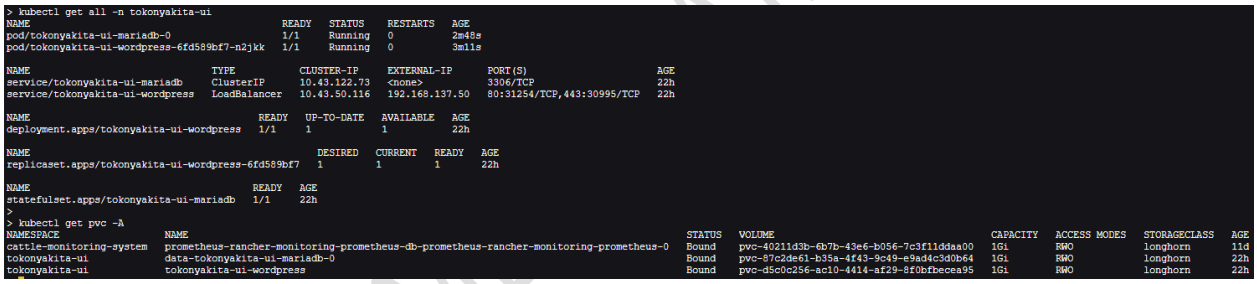
Di sisi Persistent Storage with longhorn :

- K8s downstream cluster pada DC-DR menggunakan longhorn untuk persistent storage
- DC-DR menggunakan target backup yang sama, support NFS atau S3
- Memiliki backup volume, baik dengan metode sekali backup maupun berulang

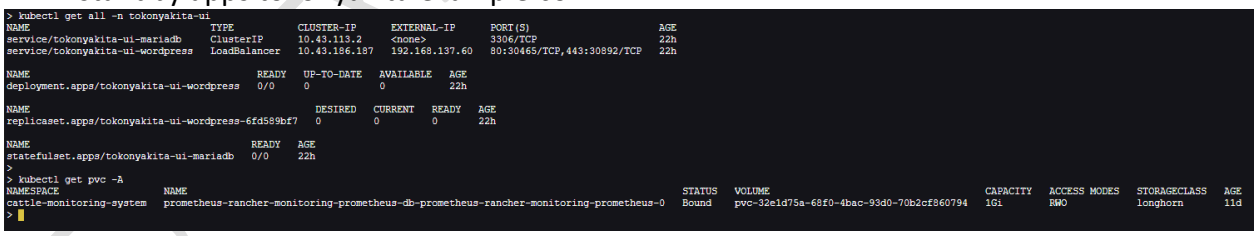
Memiliki inventory terkait persistent volume (PV) & persistent volume claim (PVC) yang related dengan aplikasi.

6.10.3 Kondisi Existing

- DC running apps tokonyakita.example.com dengan persistent storage with longhorn, load mengarah ke DC



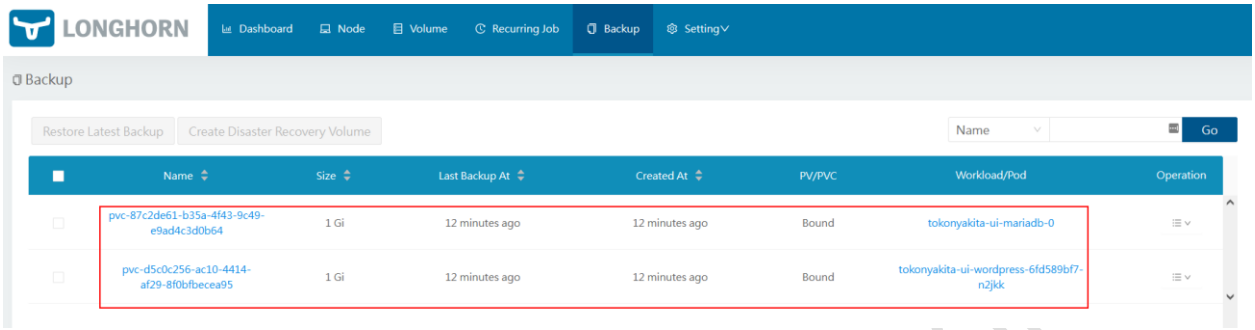
- DR standby apps tokonyakita.example.com



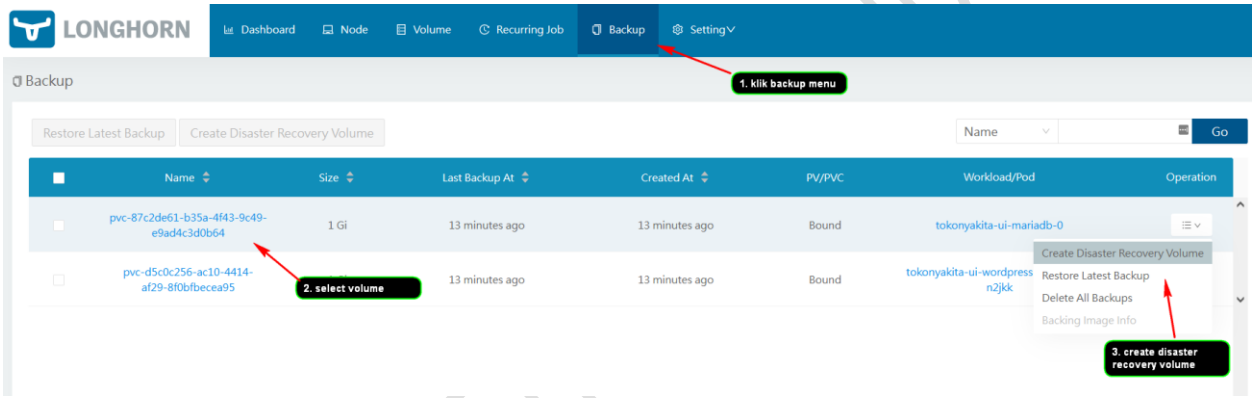
6.10.4 Use Case

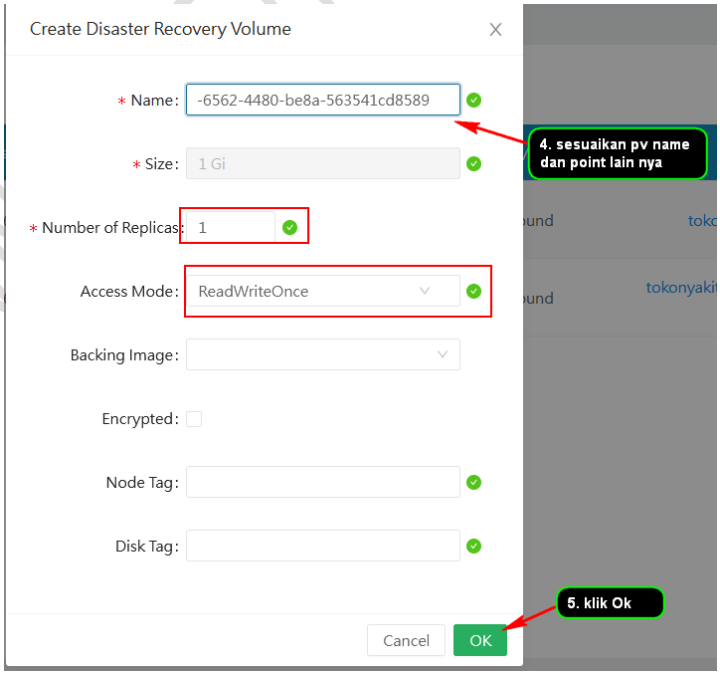
- Metode aktif-passive
- DC down, perlu tindakan untuk up system di sisi DR
- Untuk peralihan koneksi bisa dengan metode pergantian ip publik dc/dr pada main domain

6.10.5 Operation

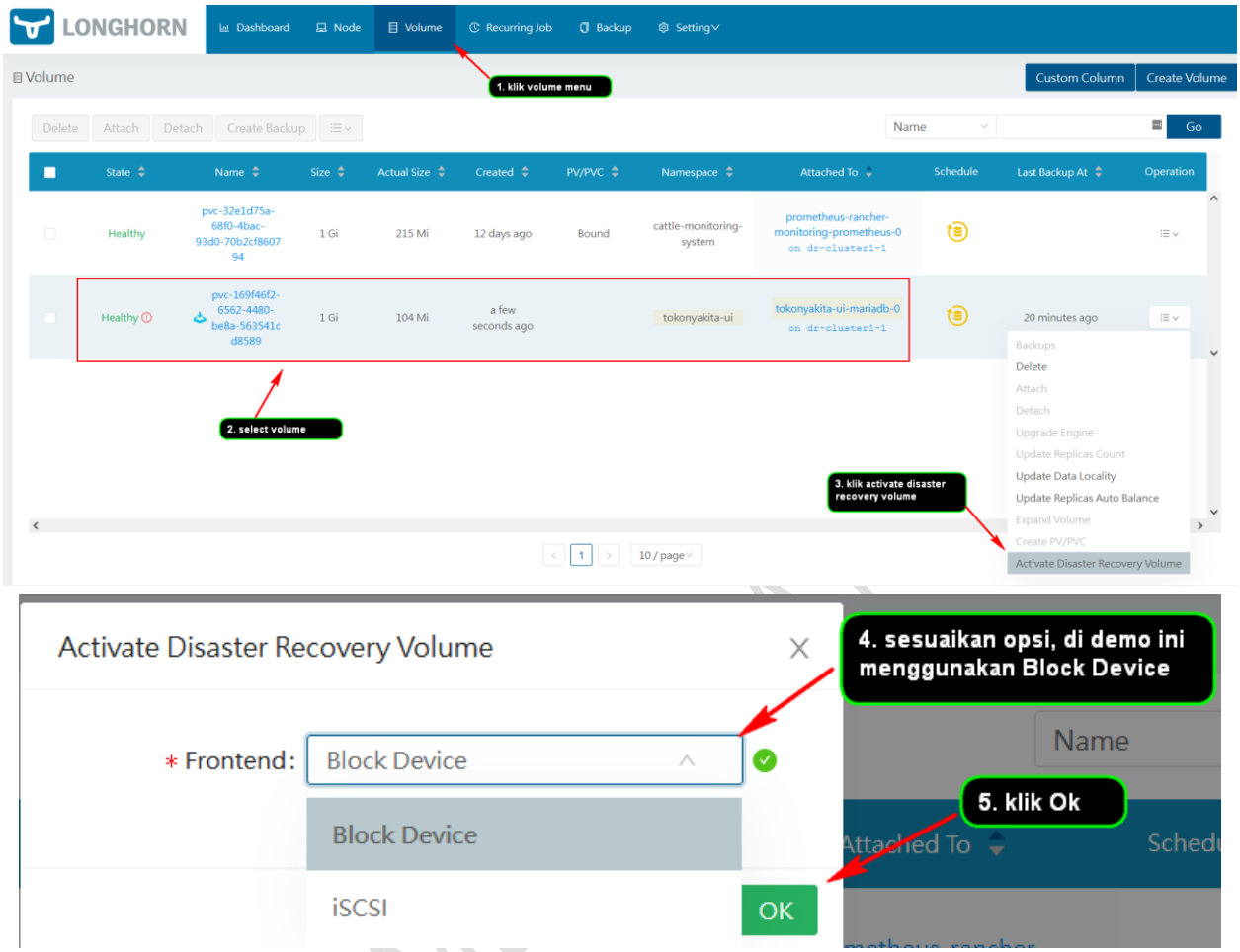


Backup volume, ready

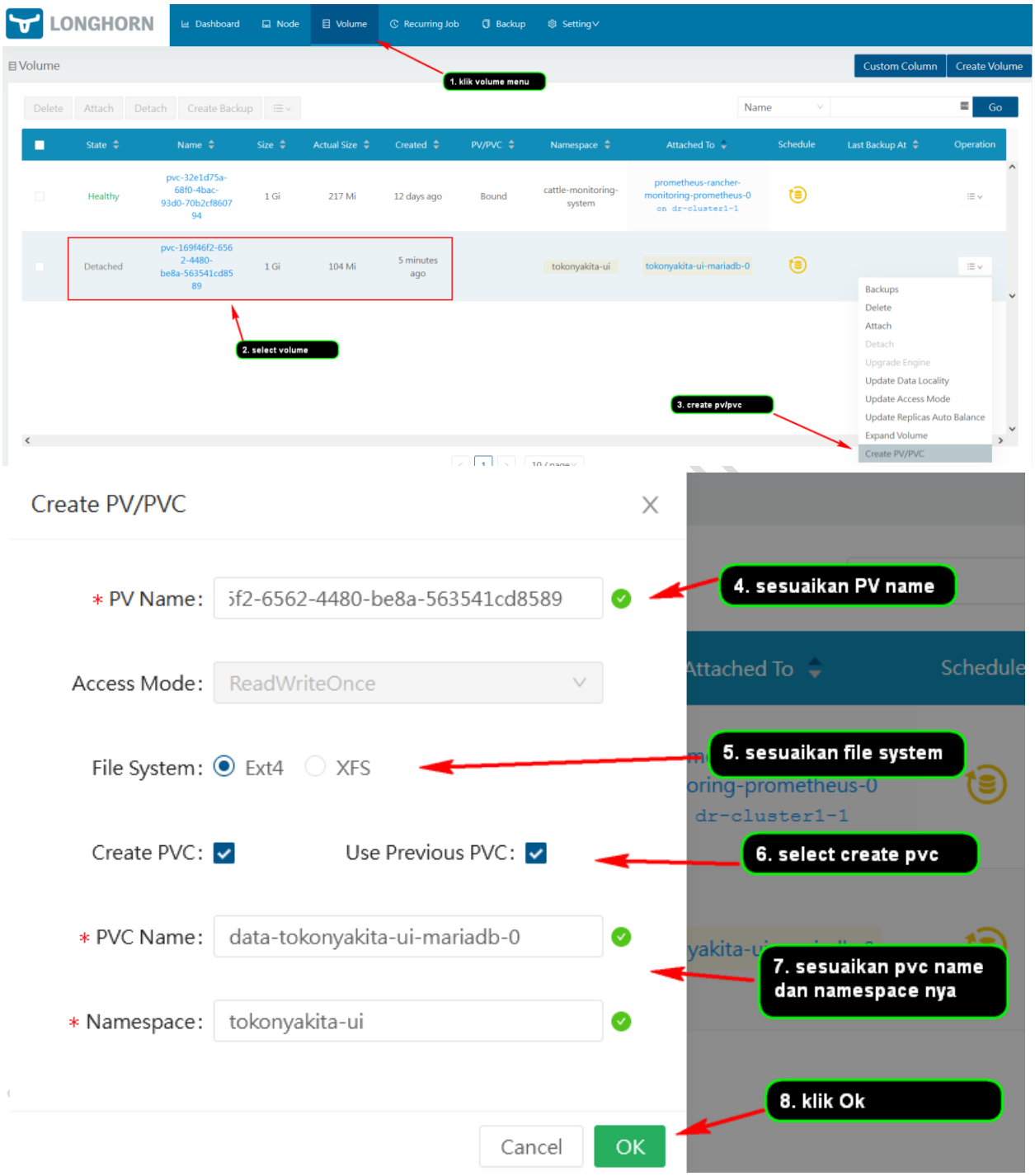




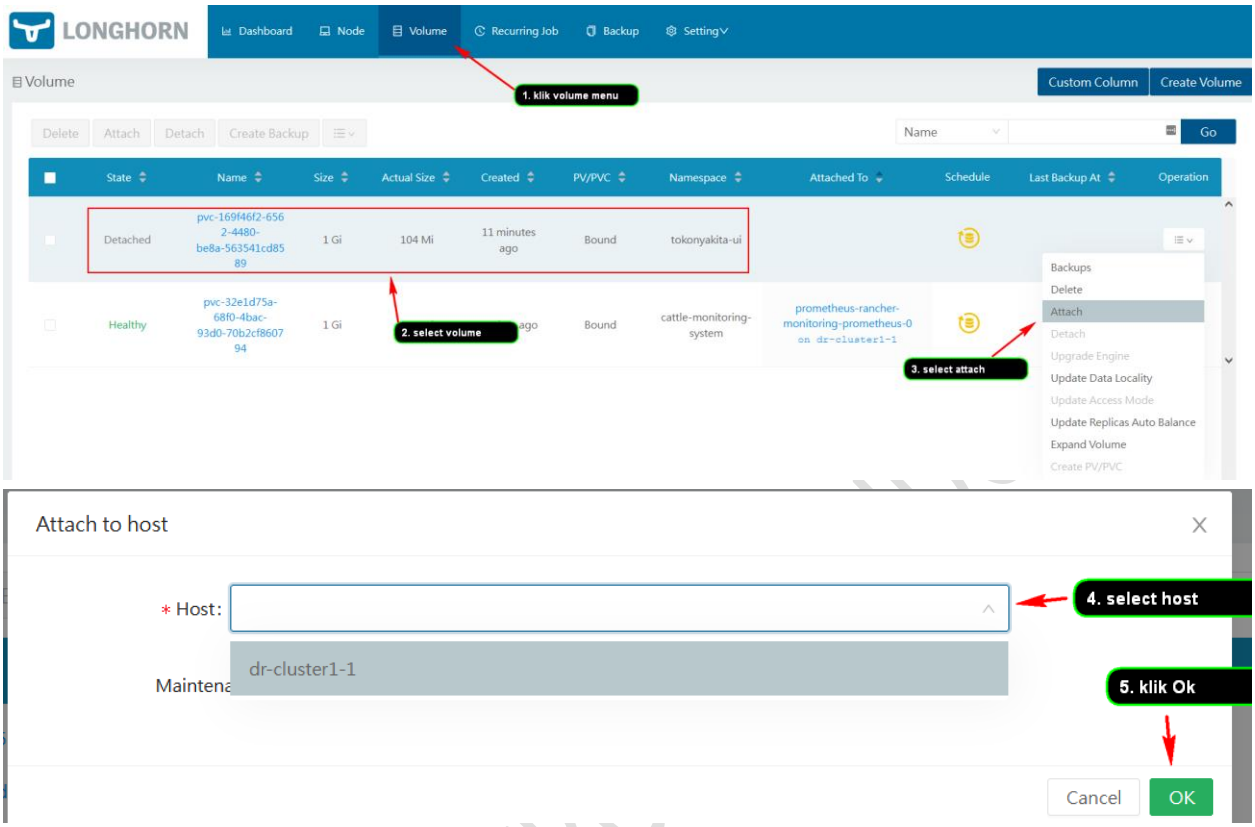
Create volume pada disaster recovery, dari volume DC yang sudah ter-backup



Activate disaster recovery volume

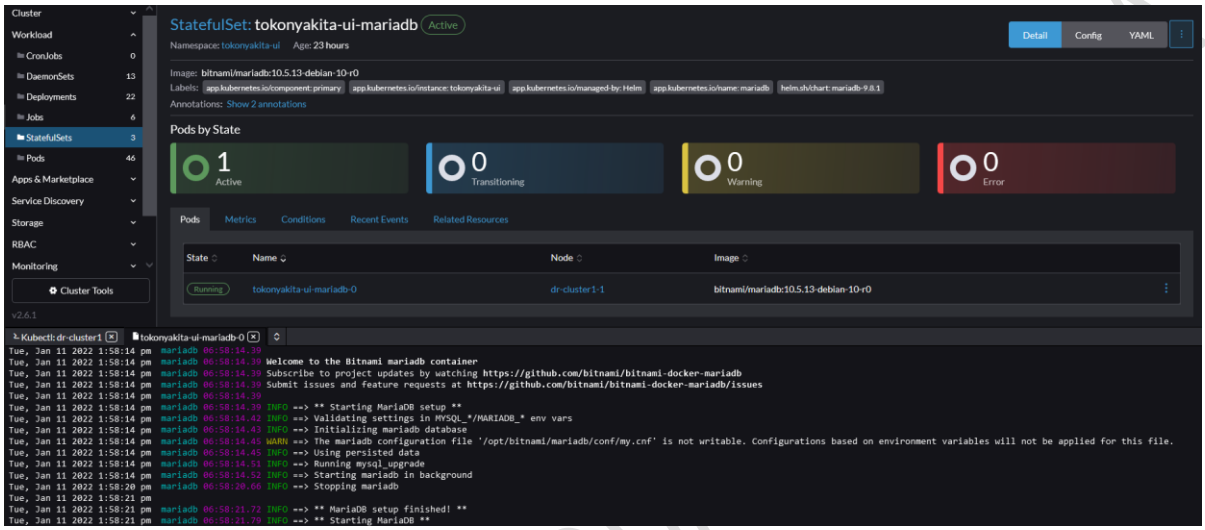


Create PV/PVC



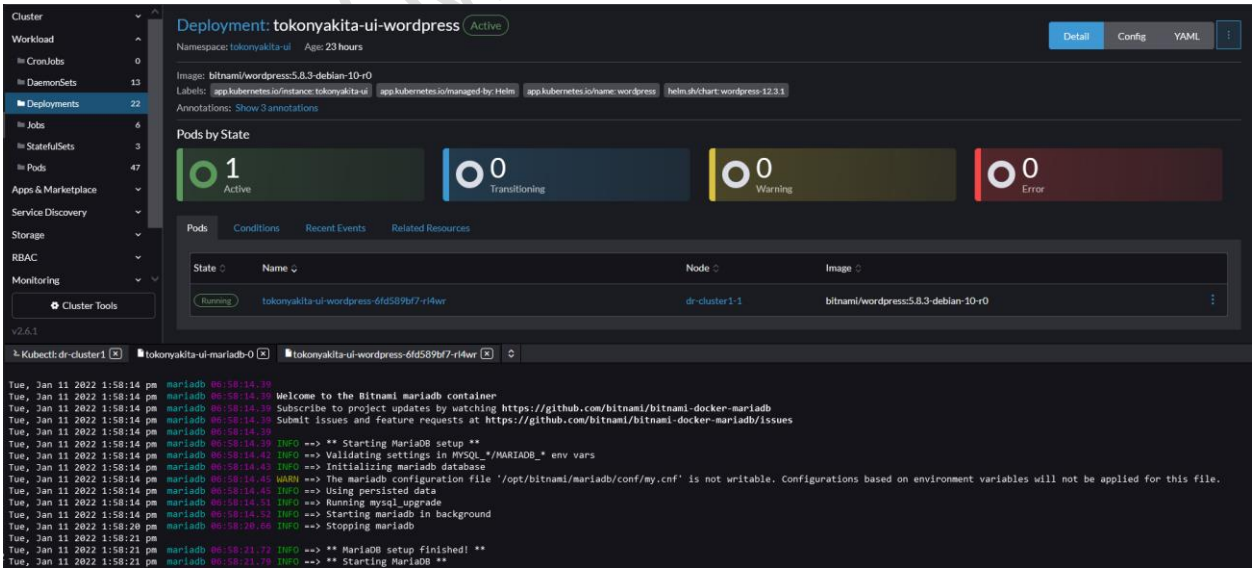
Attach volume

```
> kubectl get pod -n tokonyakita-ui
No resources found in tokonyakita-ui namespace.
>
> kubectl scale statefulset tokonyakita-ui-mariadb --replicas 1 -n tokonyakita-ui
statefulset.apps/tokonyakita-ui-mariadb scaled
>
> kubectl get pod -n tokonyakita-ui
NAME                      READY   STATUS    RESTARTS   AGE
tokonyakita-ui-mariadb-0   1/1     Running   0          91s
>
```



Scale up database

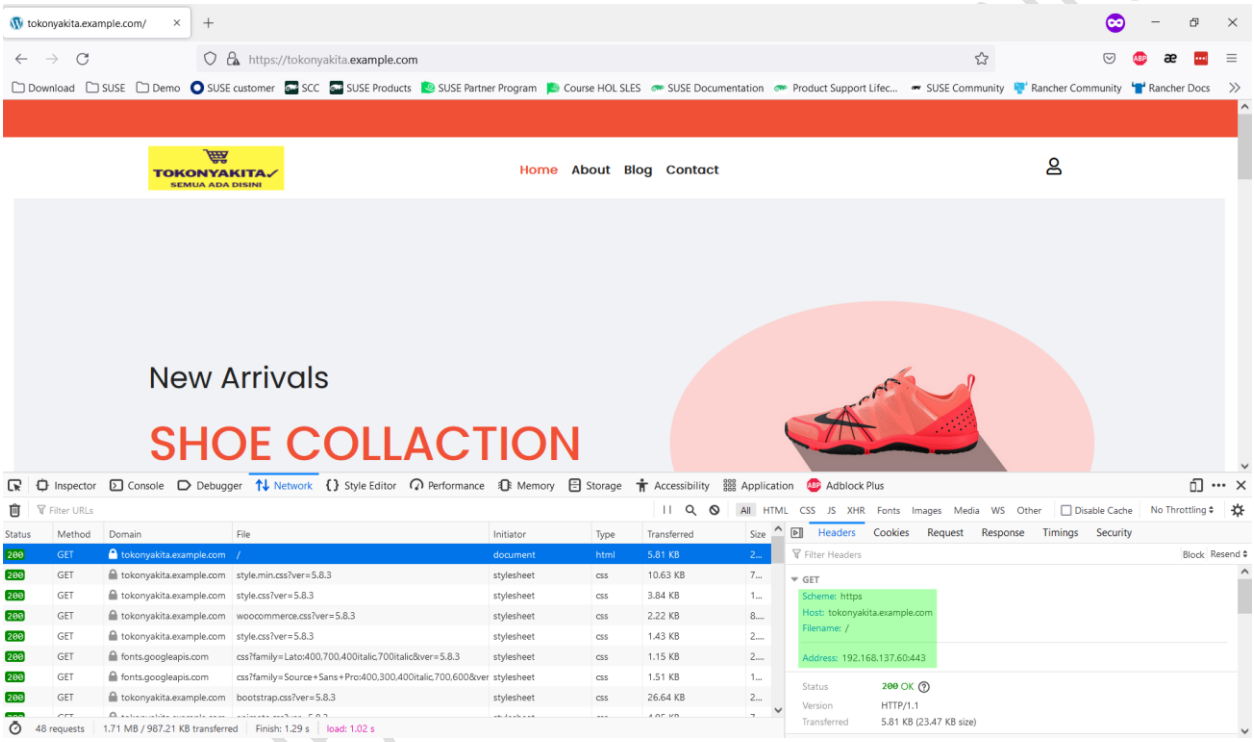
```
> kubectl scale deployment tokonyakita-ui-wordpress --replicas 1 -n tokonyakita-ui
deployment.apps/tokonyakita-ui-wordpress scaled
> kubectl get pod -n tokonyakita-ui
NAME                                    READY   STATUS    RESTARTS   AGE
tokonyakita-ui-mariadb-0                1/1     Running   0          9m2s
tokonyakita-ui-wordpress-6fd589bf7-rl4wr   1/1     Running   0          70s
>
```



Lakukan langkah yang sama untuk up aplikasi

Healthy	pvc-169f46f2-6562-4480-be8a-563541cd8589	1 Gi	126 Mi	27 minutes ago	Bound	tokonyakita-ui	tokonyakita-ui-mariadb-0 on dr-cluster1-1	
Healthy	pvc-afbd4254-dc5a-46be-bb66-7c9d33d08ea0	1 Gi	262 Mi	7 minutes ago	Bound	tokonyakita-ui	tokonyakita-ui-wordpress-6fd589bf7-rl4wr on dr-cluster1-1	

Volume Ok, sudah dalam status healthy



Di Demo ini, DC menggunakan ip 50, DR menggunakan ip 60


PT Metrodata Electronics Tbk
METRODATA
Digital Solution &
Distribution Company

METRODATA

SYNNEX

Network & Security

AppCAMO, ARISTA, ASUS, Barracuda, Cisco, Cisco Meraki, Citrix, CommScope Ruckus, DELL Technologies, domotz, Edgecore Networks, efficient iP, EnGenius, Forcepoint, FORESCOUT, FORTINET, Horangi Cybersecurity, HUAWEI, INDEED ID, Infoblox, KEYSIGHT Technologies, LiveAction, MICRO FOCUS, NETSCOUT, NOMADIX, NOZOMI NETWORKS, Progress, SAMSUNG, SOPHOS, TELTONIKA Networks, tp-link, TREND MICRO, UBIQUITI, VERSA NETWORKS

Server & Storage

asustor, Cisco, DELL Technologies, FUJITSU, Hewlett Packard Enterprise, HUAWEI, intel., Lenovo, NetApp, nimblestorage, NUTANIX, ORACLE, SEAGATE, simplivity, SOLIDFIRE, SUPERMICRO, Synology

Productivity

AUTODESK Value Added Distributor, Citrix, DELL Technologies, HULFT, logitech, Microsoft 365, neverware, Progress, Sherpa soft, SOLID EDGE, StorMagic, SUSE, TeamViewer, VEEAM

Data Protection

Barracuda, CARBONITE, DELL Technologies, FalconStor, HYCU, MICRO FOCUS, rubrik, StorMagic, VEEAM, VERITAS

Audio Visual

acer, EPOS, JBL, SAMSUNG

Surveillance & Security System

AXIS Communications, BOSCH Invented for life, Cisco Meraki, GeoVision, HIKVISION, uniarch, unv, V5 SYSTEMS, Hanwha Techwin WISENET

Collaboration

Cisco

Data & Analitycs

acer, ADLINK, ADVANTECH, anyVision, AVIRTECH, Gorilla, Informatica Authorised Distributor, MICRO FOCUS, Microsoft 365, ORACLE, Progress, TigerGraph, VERTICA

IaaS (Infrastructure as a Service)

Azure, ORACLE

SaaS (Software as a Service)

Acronis, Barracuda, BitTitan, cloudm, Deskera, DigitalSafe, nomadesk, Office 365, PrivaTalk, Progress, SAMSUNG Knox

Printing, Peripheral & Tools

acer, AOC, ASUS, DELL, EPSON, FUJITSU, hp, Kodak alaris, Lenovo, Lexmark, PHILIPS, ViewSonic, ZEBRA

Devices & Smartphone

ASUS, Infinix, itel, SAMSUNG, vivo

Memory, Flash & Components

Apacer, ASUS, INNO3D, intel., Kingston, msi, GEFORCE, SAMSUNG, SEAGATE, TEAMGROUP, thermaltake, Transcend

Electronics

beko, Panasonic, PHILIPS, PHILIPS hue

PC End Computing

acer, AMD, ASUS, AVITA, DELL, FUJITSU, GIGABYTE, hp, intel., Lenovo, msi, NEXSTGO

Gaming & Accesories

ROG, ALIENWARE, AORUS, REPUBLIC OF GAMERS, GIGABYTE, HYPER X, JBL, logitech, msi, OMEN, RAZER, TUF GAMING

Rack, Power & Cooling

APC, DELL Technologies, EATON, Hewlett Packard Enterprise, VERTIV, V5 SYSTEMS

Education

EC-Council

Indonesia Technology Solution

baktiku, HEATRAX, Myscorecard, pajak, Paques, POKA YOKE, qiscus, SMARTernak, TapTalk.io, TERADOCU