BAB 4

IMPLEMENTASI DAN PEMBAHASAN

4.1 Implementasi

Pada bagian ini peneliti akan menjelaskan langkah-langkah membangun docker swarm *cluster* sebagai server hosting yang memiliki tingkat ketersediaan yang tinggi atau memiliki *High Availability* yang tinggi.

4.1.1 Docker Swarm

Gambar 4.1 merupakan konfigurasi file /etc/hosts yang digunakan untuk melakukan resolve hostname node ke ip address. Konfigurasi ini dilakukan pada ketiga node

127.0.0.1 localhost
10.10.10.75 manager
10.10.10.76 worker1
10.10.10.77 worker2

Gambar 4.1 Konfigurasi File /etc/hosts

Selanjutnya melakukan validasi apakah konfigurasi file /etc/hosts sudah berjalan dengan menggunakan perintah ping ke hostname seperti pada gambar 4.2

```
root@manager:/home/ghofarnugroho# ping worker1
PING worker1 (10.10.10.76) 56(84) bytes of data.
64 bytes from worker1 (10.10.10.76): icmp seq=1
ttl=64 time=0.652 ms
64 bytes from worker1 (10.10.10.76):
                                        icmp seq=2
ttl=64 time=0.310 ms
64 bytes from worker1 (10.10.10.76):
                                        icmp seq=3
ttl=64 time=0.323 ms
64 bytes from worker1 (10.10.10.76): icmp seq=4
ttl=64 time=0.361 ms
--- worker1 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss,
time 3068ms
rtt min/avg/max/mdev = 0.310/0.411/0.652/0.141 ms
root@manager:/home/ghofarnugroho# ping worker2
PING worker2 (10.10.10.77) 56(84) bytes of data.
64 bytes from worker2 (10.10.10.77): icmp_seq=1
ttl=64 time=0.686 ms
64 bytes from worker2 (10.10.10.77):
                                        icmp seq=2
ttl=64 time=0.238 ms
64 bytes from worker2 (10.10.10.77): icmp seq=3
ttl=64 time=0.260 ms
--- worker2 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss,
time 2031ms
rtt min/avg/max/mdev = 0.238/0.394/0.686/0.207 ms
```

Gambar 4.2 Perintah Validasi Konfigurasi File /etc/hosts

Gambar 4.3 merupakan perintah untuk melakukan update package

pada Ubuntu Server. Perintah ini dijalankan pada ketiga node

```
root@manager:/home/ghofarnugroho# apt update
Hit:1 http://id.archive.ubuntu.com/ubuntu bionic
InRelease
Hit:2 http://id.archive.ubuntu.com/ubuntu bionic-
updates InRelease
Hit:3 http://id.archive.ubuntu.com/ubuntu bionic-
backports InRelease
Hit:4 http://id.archive.ubuntu.com/ubuntu bionic-
security InRelease
Reading package lists... Done
Building dependency tree
Reading state information... Done
85 packages can be upgraded. Run 'apt list --
upgradable' to see them.
```

Gambar 4.3 Perintah untuk Update Package Ubuntu Server

Selanjutnya, melakukan instalasi docker pada ketiga node dengan

perintah pada gambar 4.4

root@manager:/home/ghofarnugroho# apt install docker.io Reading package lists... Done Building dependency tree Reading state information... Done The following additional will packages be installed: bridge-utils containerd pigz runc ubuntu-fan Suggested packages: ifupdown aufs-tools cgroupfs-mount | cgroup-lite debootstrap docker-doc rinse zfs-fuse | zfsutils The following NEW packages will be installed: bridge-utils containerd docker.io piqz runc ubuntu-fan 0 upgraded, 6 newly installed, 0 to remove and 85 not upgraded. Need to get 74.2 MB of archives. After this operation, 360 MB of additional disk space will be used. Do you want to continue? [Y/n] y

Gambar 4.4 Perintah untuk Instalasi Docker

Langkah selanjutnya adalah melakukan generate token pada node manager agar node worker dapat melakukan join ke node manager dengan menggunakan token yang degenerate tersebut. Perintah untuk generate token terdapat pada gambar 4.5

```
root@manager:/home/ghofarnugroho# docker swarm init
Swarm
            initialized:
                                               node
                                current
(mlc6zpodvemfxggpi87f1xvft) is now a manager.
To add a worker to this swarm, run the following
command:
   docker
                                          SWMTKN-1-
             swarm
                       join
                               --token
1kx8rryt9ocbb7masu4k71t7dw39n2axhmdyal2ggwcxdjcirr-
104t4w7p251lbu8fy1gtyytoq 10.10.10.75:2377
To add a manager to this swarm, run 'docker swarm
join-token manager' and follow the instructions.
```

Gambar 4.5 Perintah untuk Generate Token

Selanjutnya peneliti melakukan join cluster pada kedua node worker agar dapat dimanage oleh node manager seperti pada gambar

4.6

```
root@worker1:/home/ghofarnugroho# vi /etc/hosts
root@worker1:/home/ghofarnugroho# docker swarm join
--token SWMTKN-1-
1kx8rryt9ocbb7masu4k71t7dw39n2axhmdyal2ggwcxdjcirr-
104t4w7p251lbu8fy1gtyytoq 10.10.10.75:2377
This node joined a swarm as a worker.
```

Gambar 4.6 Perintah untuk Join Cluster Manager

Selanjutnya peneliti melakukan check apakah kedua worker sudah

terjoinkan. Perintahnya seperti pada gambar 4.7

root@manager:/home/ghofarnugroho# docker node ls					
ID		HOSTNAME	STATUS		
AVAILABILITY	MANAGER STATUS	ENGINE VERSI	ON		
mlc6zpodvemfx;	ggpi87flxvft *	manager	Ready		
Active	Leader	20.10.7			
laua81aonwp1w8	3nt8rchff21j	worker1	Ready		
Active		20.10.7			
8kuxgelg7c1tv	vya3ajnuquf0	worker2	Ready		
Active		20.10.7			

Gambar 4.7 Perintah Verifikasi Node Worker

Selanjutnya peneliti melakukan instalasi docker-compose pada node manager yang berfungsi untuk melakukan deployment image docker. Perintah untuk melakukan instalasi docker-compose terdapat pada gambar 4.8

root@manager:~# curl -L "https://github.com/docker/compose/releases/download /1.24.0/docker-compose-\$(uname -s)-\$(uname -m)" -0 /usr/local/bin/docker-compose % Total % Received % Xferd Average Speed Time Time Time Current Upload Dload Total Spent Left Speed 0 0 0 0 0 0 --:--0 0 0 :-- --:--:--100 15.4M 0 4821k 100 15.4M 0 0 0:00:03 0:00:03 --:-- 5638k root@manager:~# ls /usr/local/bin docker-compose root@manager:~# chmod +x/usr/local/bin/dockercompose

Gambar 4.8 Perintah untuk Instalasi docker-compose

Selanjutnya peneliti melakukan check apakah docker-compose sudah bias digunakan atau belum. Dengan perintah docker-compose – version

```
root@manager:~# docker-compose -version
docker-compose version 1.24.0, build 0aa59064
Gambar 4.9 Perintah Verifikasi docker-compose
```

4.1.2 Deploy Local Registry

Docker registry merupakan repository yang digunakan untuk meyimpan image docker yang akan dijalankan pada docker swarm. Pada saat kita akan menjalankan service pada docker swarm image akan dibuild pada node manager kemudian image tersebut akan didistribusikan ke semua node worker melalui registry. Gambar 4.10 merupakan perintah yang digunakan untuk melakukan deploy registry pada docker swarm. --publish merupakan setting expose port agar registry dapat diakses dari luar docker seperti akses dari workstation melalui laptop. --name digunakan untuk memberi nama service dalam hal ini adalah registry. registry:2 merupakan base image yang digunakan.

```
root@manager:~# docker service create --name
registry --publish published=5000,target=5000
registry:2
glvzjxysr68ohlnqmcs6fvcn9
overall progress: 1 out of 1 tasks
1/1: running
verify: Service converged
```

Gambar 4.10 Perintah untuk *Deploy Registry*

Selanjutnya peneliti melakukan verifikasi apakah service registry sudah berjalan dengan menggunakan perintah docker service ls seperti pada gambar 4.11

root@manager:				
ID	NAME	MODE		REPLICAS
IMAGE	PORTS			
glvzjxysr68o	regist	try rej	plicated	1/1
registry:2	*:5000->500	0/tcp		

Gambar 4.11 Perintah untuk Verifikasi Service Registry

4.1.3 Virtual IP

Virtual IP merupakan IP yang nantinya akan digunakan client untuk melakukan akses pada aplikasi - aplikasi yang dihosting pada docker swarm. Virtual IP ini bekerja dengan cara melakukan monitoring pada semua node, semisal virtual ip ini diinstall pada node manager dan sewaktu waktu node manager down maka virtual ip ini akan otomatis berpindah ke node worker. Sehingga aplikasi yang dihosting tetap bias diakses oleh client.

Langkah pertama peneliti melakukan install software cluster manager pada semua node.

root@manager:/home/ghofarnugroho#	apt	install
pacemaker corosync crmsh haveged		
Reading package lists Done		
Building dependency tree		
Reading state information Done		
The following additional pack	ages	will be
installed:		
cluster-glue ibverbs-providers	libcfg	6 libcib4
libcmap4 libcorosync-common4 libcpg	4	
libcrmcluster4 libcrmcommon3	libc	rmservice3
libesmtp6 libhavege1 libibverbs1 li	blrm2	liblrmd1
libltdl7 libnet1 libnl-route-	3-200	libnspr4
libnss3 libopenhpi3 libopenipmi0 li	bpe-ru	les2
libpe-status10 libpengine10 lib	pils2	libplumb2
libplumbgpl2 libpython-stdlib		
libpython2.7-minimal libpython2.	7-stdl	ib libqb-
dev libqb0 libquorum5 librdmacm1 li	bsenso	rs4
libsnmp-base libsnmp30 libstatgr	ab10 l	ibstonith1
libstonithd2 libtimedate-perl libto	tem-pg	5
libtransitioner2 libvotequorum8	lib	xml2-utils
openhpid pacemaker-cli-utils pacema	ker-co	mmon
pacemaker-resource-agents pyth	ion	python-bs4
python-chardet python-dateutil pyth	on-htm	15lib
Gambar 4 12 Perintah untuk Install Softwar	o Cluster	Manager

Gambar 4.12 Perintah untuk Install Software Cluster Manager

Selanjutnya, peneliti melakukan stop cluster manager service

agar bisa dikonfigurasi pada semua node.

```
root@manager:/home/ghofarnugroho# systemctl stop
corosync pacemaker
```

Gambar 4.13 Perintah untuk Stop Cluster Manager Service

Selanjutnya peneliti membuat authorization key untuk cluster

pada node manager

root@manager:/home/ghofarnugroho# corosync-keygen Corosync Cluster Engine Authentication key generator. Gathering 1024 bits for key from /dev/random. Press keys on your keyboard to generate entropy. Press keys on your keyboard to generate entropy (bits = 920). Press keys on your keyboard to generate entropy (bits = 1000). Writing corosync key to /etc/corosync/authkey.

Gambar 4.14 Perintah untuk Membuat Authorization Key

Selanjutnya peneliti melakukan konfigurasi file

/etc/corosync/corosync.conf pada node manager

```
totem {
 version: 2
 cluster_name: docker-swarm
 transport: udpu
 interface {
   ringnumber: 0
   bindnetaddr: 10.10.10.0
   broadcast: yes
   mcastport: 5407
  }
}
nodelist {
 node {
   ring0 addr: manager
  }
 node {
   ring0 addr: worker1
  }
 node {
   ring0 addr: worker2
  }
}
quorum {
 provider: corosync votequorum
}
logging {
 to logfile: yes
 logfile: /var/log/corosync/corosync.log
 to syslog: yes
 timestamp: on
}
service {
 name: pacemaker
  ver: 0
```

Gambar 4.15 Konfigurasi File corosync.conf

Selanjutnya copy file authorization dan konfigurasi corosync ke semua node worker

```
root@manager:/home/ghofarnugroho#
                                               -czvf
                                       tar
corosync.tar.gz /etc/corosync/
tar: Removing leading `/' from member names
/etc/corosync/
/etc/corosync/uidgid.d/
/etc/corosync/authkey
/etc/corosync/corosync.conf
root@manager:/home/ghofarnugroho#
                                                 scp
corosync.tar.gz ghofarnugroho@worker1:~/
ghofarnugroho@worker1's password:
corosync.tar.gz
100%
     666
           975.6KB/s
                        00:00
root@manager:/home/ghofarnugroho#
                                                 scp
corosync.tar.gz ghofarnugroho@worker2:~/
The authenticity of host 'worker2 (10.10.10.77)'
can't be established.
ECDSA
                             fingerprint
               key
                                                  is
SHA256:1yl2VL45WVv9psh7Qqcn009b9BHvjTf9c/xn690SBA4.
Warning: Permanently added 'worker2,10.10.10.77'
(ECDSA) to the list of known hosts.
ghofarnugroho@worker2's password:
corosync.tar.gz
100% 666
             1.3MB/s
                        00:00
```

Gambar 4.16 Perintah Copy File Authorization dan Konfigurasi corosync

Selanjutnya peneliti melakukan replace file konfigurasi corosync pada semua node worker.

```
root@worker2:/home/ghofarnugroho#
                                        rm
                                                 -rf
/etc/corosync/
root@worker2:/home/ghofarnugroho#
                                                  mv
corosync.tar.gz /etc/
root@worker2:/home/ghofarnugroho# cd /etc/
root@worker2:/etc# tar -xzvf co
console-setup/ corosync.tar.gz
root@worker2:/etc# tar -xzvf co
console-setup/
               corosync.tar.gz
root@worker2:/etc# tar -xzvf corosync.tar.gz
etc/corosync/
etc/corosync/uidgid.d/
etc/corosync/authkey
etc/corosync/corosync.conf
```

Gambar 4.17 Perintah untuk Replace File Konfigurasi corosync

Selanjutnya ubah owner pada folder /etc/corosync agar menjadi

root

root@worker2:/etc# chown -R root:root /etc/corosync Gambar 4.18 Perintah untuk Mengubah *Owner* folder /etc/corosync

Selanjutnya menjalankan service corosync dan pacemaker pada

semua node

root@manager:/etc# systemctl start corosync
root@manager:/etc# systemctl start pacemaker

Gambar 4.19 Perintah untuk Menjalankan Service corosync dan pacemaker

Selanjutnya peneliti melakukan setting agar cluster manager

dapat dijalankan otomatis ketika booting pada semua node

```
root@manager:/etc# systemctl enable corosync
Synchronizing state of corosync.service with SysV
service script with
                         /lib/systemd/systemd-sysv-
install.
Executing: /lib/systemd/systemd-sysv-install enable
corosync
root@manager:/etc# update-rc.d pacemaker
                                           defaults
20 01
root@manager:/etc# systemctl enable pacemaker
Synchronizing state of pacemaker.service with SysV
service
         script
                  with
                         /lib/systemd/systemd-sysv-
install.
Executing: /lib/systemd/systemd-sysv-install enable
pacemaker
```

Gambar 4.20 Perintah untuk Enable corosync

Selanjutnya cek status cluster, pastikan semua node online

```
root@manager:/etc# crm status
Stack: corosync
Current DC: worker1 (version 1.1.18-2b07d5c5a9) -
partition with quorum
Last updated: Fri May 13 16:48:27 2022
Last change: Fri May 13 16:43:04 2022 by hacluster
via crmd on worker1
3 nodes configured
0 resources configured
Online: [ manager worker1 worker2 ]
No resources
```

Gambar 4.21 Perintah Pengecekan Status Cluster

Selanjutnya peneliti melakukan disable STONITH dan Quorum

Policy agar tidak terjadi konflik pada konfigurasi pada node manager

rooccillanager ./ cco	CIM	contigure	property	stonith-
enabled=false				
root@manager:/etc#	crm	n configu	re prope	erty no-
quorum-policy=ignor	e			

Gambar 4.22 Perintah untuk Disable STONITH dan Quorum Policy

Selanjutnya peneliti melakukan konfigurasi virtual ip. IP: adalah virtual ip yang digunakan, nic: adalah interface network yang digunakan, interfal: adalah lama waktu pengecekan untuk monitoring status node apakah up atau down. Migration threshold: adalah batas toleransi node down untuk memenuhi syarat agar virtual ip dapat berpindah ke node yang lain.

root@manager:/etc	:# crm	configure	primitive
virtual ip	ocf:heartb	eat:IPaddr2	params
ip="10.10.10.78"	cidr_netma	sk="32" nic:	="ens160" op
monitor interval=	"3s" meta n	migration-the	reshold="5"

Gambar 4.23 Perintah untuk Konfigurasi Virtual IP

Selanjutnya peneliti melakukan cek apakah konfigurasi cluster

manager sudah benar

root@manager:/etc# crm resource show virtual_ip (ocf::heartbeat:IPaddr2): Started

Gambar 4.24 Perintah Pengecekan Status Node

Selanjutnya cek virtual ip address

root@mana	ger:/etc#	ip addr	gr	ep 10.10.10.78	
inet	10.10.10	.78/32	brd	10.10.10.255	scope
global er	ns160				

Gambar 4.25 Verifikasi Virtual IP Address

4.2 Pengujian

Setelah proses implementasi selesai, selanjutnya dilakukan pengujian untuk membuktikan sistem yang dibuat memenuhi tujuan penelitian. Pengujian dilakukan sebanyak 4 kali. Pengujian pertama dilakukan dengan mematikan salah satu node worker, pengujian kedua dilakukan dengan mematikan node manager dan worker1, pengujian ketiga dilakukan dengan mematikan node manager, pengujian keempat dilakukan dengan mematikan semua node. Pengujian ini dilakukan untuk mengetahui tingkat ketersediaan system yang dibangun apakah tinggi atau rendah.

4.2.1 Deploy Service Web Dummy

Untuk dapat melakukan pengujian dibutuhkan sebuah service web dummy. Gambar 4.26 merupakan file docker-compose.yml yang berisi nama image yang akan dibuild. Nama image tersebut adalah nginx.

```
version: '3'
services:
nginx:
image: 127.0.0.1:5000/nginx
build: .
ports:
    - "80:80"
```

Gambar 4.26 Konfigurasi File docker-compose.yml

Membuat Dockerfile yang akan digunakan untuk melakukan build image nginx

FROM nginx

Gambar 4.27 Konfigurasi Dockerfile

Selanjutnya build dan push image nginx ke local registry menggunakan perintah pada gambar 4.28

```
root@manager:/home/ghofarnugroho/nginx# docker-
compose build --no-cache
Building nginx
Step 1/1 : FROM nginx
latest: Pulling from library/nginx
214ca5fb9032: Pull complete
f0156b83954c: Pull complete
5c4340f87b72: Pull complete
9de84a6a72f5: Pull complete
63f91b232fe3: Pull complete
860d24db679a: Pull complete
Digest:
sha256:2c72b42c3679c1c819d46296c4e79e69b2616fa28bea92
e61d358980e18c9751
Status: Downloaded newer image for nginx:latest
---> 7425d3a7c478
Successfully built 7425d3a7c478
Successfully tagged 127.0.0.1:5000/nginx:latest
root@manager:/home/ghofarnugroho/nginx# docker-
compose push
Pushing nginx (127.0.0.1:5000/nginx:latest)...
The push refers to repository [127.0.0.1:5000/nginx]
feb57d363211: Pushed
98c84706d0f7: Pushed
4311f0ea1a86: Pushed
6d049f642241: Pushed
3158f7304641: Pushed
fd95118eade9: Pushed
latest: digest:
sha256:787480bfb4297dc887f8655dbc51074ef87f16ea359bae
ea3af0a4dd92948124 size: 1570
```

Gambar 4.28 Perintah untuk Build dan Push Image Docker

Setelah image berhasil dipush ke registry selanjutnya membuat docker service nginx dengan jumlah replika 3. Penulis menentukan jumlah replika sebanyak 3 dikarenakan dengan 3 replika sudah dapat untuk mengatasi request dari user sebanyak 50.000 request. Perintah pada gambar 4.29 akan membuat service nginx dengan nama nginx dan melakukan publish port 80 ke network eksternal agar dapat diakses oleh client.

docker

```
service create --name nginx --replicas 3 -p 80:80
127.0.0.1:5000/nginx
lmaaqhwqt31z16lup9f4xn8fr
overall progress: 3 out of 3 tasks
1/3: running
2/3: running
3/3: running
verify: Service converged
```

Gambar 4.29 Perintah untuk Membuat Docker Service Nginx

Untuk melakukan verifikasi cluster yang sudah dibuat dapat menggunakan perintah docker service ls seperti pada gambar 4.30

root@manager:/h	docker						
service ls							
ID	NAME	MODE	REPLICAS				
IMAGE		PORTS					
lmaaqhwqt31z	nginx	replicated	d 3/3				
127.0.0.1:5000/	'nginx:latest	*:80->80/tcp					
Gambar 4.30 Perintah untuk Verifikasi Cluster							

Gambar 4.30 Perintah untuk Verifikasi Cluster

Selanjutnya peneliti melakukan pengujian service web menggunakan laptop workstation dengan mengakses virtual ip yang telah dibuat tadi, yaitu 10.10.10.78



Welcome to nginx!

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

For online documentation and support please refer to <u>nginx.org</u>. Commercial support is available at <u>nginx.com</u>.

Thank you for using nginx.

Gambar 4.31 Website Dummy Dengan Virtual IP

4.2.2 Pengujian High Availability

Untuk melakukan pengujian diperlukan sebuah tool yaitu JMeter. Membuat test plan pada JMeter dengan klik kanan Test Plan \rightarrow Add \rightarrow Threads (User) \rightarrow Thread Grup



Gambar 4.32 Step Pembuatan Thread Group

Number thread of user adalah banyaknya jumlah user yang akan melakukan http request ke website dummy, sedangkan loop count adalah banyaknya request yang dilakukan oleh 1 user. Dari gambar 4.33 maka website dummy akan diload sebanyak 50.000 kali

/ Apache JMeter (5.4.3)	-		×
<u>File Edit Search Run Options Tools H</u> elp			
	00:00:00	<u>^</u> 0 0	0
P Intead Group			
Name: Thread Group			
Comments:			
Action to be taken after a Sampler error			
Continue Start Next Inread Loop Stop Inread Stop lest Stop lest Now			
Thread Properties			
Number of Theorem (ADD			_
Number of infeats (users):			
Ramp-up period (seconds): 10			
Loop Count: Infinite 500			
✓ Same user on each iteration			
Delay Thread creation until needed			
Specify Thread lifetime			
Duration (seconds):			
Startup delay (seconds):			

Gambar 4.33 Step Input Loop Count pada Thread Group

Selanjutnya tambahkan HTTP Request dengan klik kanan pada Thread Grup \rightarrow Add \rightarrow Sampler \rightarrow HTTP Request. Isikan Virtual IP (10.10.10.78) pada kolom Server Name or IP

🖊 Apache JMeter (5	.4.3)						-		×
Eile Edit Search	Run Options Tools Help								
📑 🚳 🤷 F	- X G 🖸 +	4,	• 🕨 🔈 💿	۵ 🦋 🎬 🙈	> 🗄 👔		00:00:00	<u>^</u> 0 0	/0 😯
👇 🖕 Test Plan	าดเมต	Threa	ad Group		,				
-0-	Add	•	Sampler >	Flow Control Action					
	Add Think Times to children	1	Logic Controller >	HTTP Request					
	Start		Pre Processors >	Debug Sampler					
	Start no pauses		Post Processors >	JSR223 Sampler					
	Validate		Assertions >	AJP/1.3 Sampler	hread 🔘 Stop Test	Stop Test Now			
	Cut	Ctrl-X	Timer 🕨	Access Log Sampler					
	Сору	Ctrl-C	Test Fragment	BeanShell Sampler					
	Paste	Ctrl-V	Canfin Flamant A	Bolt Request					
	Duplicate	Ctrl+Shift-C	Config Element	FTP Request					
	Remove	Delete	Count: V Innin	GraphQL HTTP Request					
	Open		1	JDBC Request					
	Merge		ame user on each it	JMS Point-to-Point					
	Save Selection As		elay Thread creation	JMS Publisher					
	Save Node As Image	Ctrl-G	and the second life time	JMS Subscriber					
	Save Screen As Image	Ctrl+Shift-G	becity thread mean	JUnit Request					
	Enable		ion (seconds):	Java Request					
	Disable		ip delay (seconds):	LDAP Extended Request					
	Toggle	Ctrl-T		LDAP Request					
	Holp		-	Mail Reader Sampler					
	neih	3		SMTD Sempler					
				SWIP Sampler					
		3		ice sampler					

Gambar 4.34 Step Pembuatan HTTP Request

HTTP Request	
Name: HTTP Request	
Comments:	
Basic Advanced	
Web Server	
Protocol [http]: Server Name or IP: 10.10.10.78	Port Number:
HTTP Request GET V Path:	Content encoding:
Redirect Automatically 🖌 Follow Redirects 🖌 Use KeepAlive 🔄 Use multipart/form-data 🔄 Browser-con	mpatible headers
Parameters Body Data Files Upload	
Send Parameters With the Request:	
Name: Value URL Encode? Content-Type	Include Equals?
Detail Add Add from Clipboard Delete Up Down	n

Gambar 4.35 Step Input Virtual IP pada HTTP Request

Selanjutnya tambahkan Listener agar dapat melihat hasil testing yang telah dilakukan. Klik kanan pada Thread Grup \rightarrow Add \rightarrow Listener \rightarrow View Result in Table

🚺 Apache JMeter (5.	4.3)				-		×
<u>File Edit Search</u>	Run Options Tools Help						
📑 🚳 🔒 🖡	🔏 🗊 🗐 💠 – 🍕		o 🗑 🆋 🦀 🏷	E	00:00:00 🖌	<u>^</u> o o/	0
P La Test Plan	Threa	ad Group	_				
HTTP F	Add	Sampler					
-	Add Think Times to children	Logic Controller >					
	Start	Pre Processors	amplor orror				
	Start no pauses	Post Processors	ampierentor				
	Validate	Assertions	xt Thread Loop 🛛 Stop Thread 🤇	Stop Test O Stop Test Now			
	Cut Ctrl-X	Timer 🕨					
	Copy Ctrl-C	Testforment					
	Paste Ctrl-V	Test Flagment): 1				
	Duplicate Ctrl+Shift-C	Config Element		1			
	Remove Delete	Listener •	View Results Tree				
	Open	Joune 🕑 innine	Summary Report				
	Morao	me user on each ite	Aggregate Report				
	Sava Selection As	lay Throad creation	Backend Listener				
	Save Selection As	ay meau creation	Aggregate Graph				
	Save Node As Image Ctrl-G	ecify Thread lifetim	Assertion Results				
	Save Screen As Image Ctrl+Shift-G	on (seconds):	Comparison Assertion Visualizer				
	Enable	1000001100/1	Generate Summary Results				
	Disable	p delay (seconds):	Graph Results				
	Toggle Ctrl-T		JSR223 Listener				
	Help		Mailer Visualizer				
		_	Response Time Graph				
			Save Responses to a file				
	US/64 SCOPE 110K		Simple Data Writer			_	
orever preferre	d_lft forever		View Results in Table				
/ghofarnugroho			Bean Shell Listener				

Gambar 4.36 Step Pembuatan View Result in Table

Tambahkan Summary Report agar jumlah error dapat terdeteksi.

Klik kanan Thread Group \rightarrow Add \rightarrow Listener \rightarrow Summary Report



Gambar 4.37 Step Pembuatan Summary Report

Masing – masing pengujian akan dilakukan sebanyak 3 kali.

Pengujian pertama yang dilakukan adalah mematikan satu node worker.

root@manager:/home/ghofarnugroho# crm status											
Stack: corosync											
Current DC: manager (version 1.1.18-2b07d5c5a9) - partition with quorum											
Last updated: Fri May 13 20:45:21 2022											
Last change: Fri May 13 16:54:31 2022 by root via cibadmin on manager											
3 nodes configured 1 resource configured											
Online: [manager worker2]											
OFFLINE: [worker]]											
Full list of resources:											
<pre>virtual_ip (ocf::heartbeat:IPaddr2): Started manager</pre>											

Gambar 4.38 Perintah Pengecekan Status Node

Hasil yang diperoleh sebagai berikut :

Summary	Summary Report													
Name:	Summary Report													
Comments:														
Write results to file / Read from file														
Filename						Browse	Log/Display Only	y: 🗌 Errors 📄	Successes	Configure				
Label	# Samples	Average	Min	Max	Std. Dev.	Error %	Throughput	Received KB/s	Sent KB/sec	Avg. Bytes				
HTTP Request	500	- 97	72	332	22.62	0.00%	10.3/sec	8.54	1.16	853.0				
TOTAL	500	97	72	332	22.62	0.00%	10.3/sec	8.54	1.16	853.0				

Gambar 4.39 Hasil Testing Ke-1 Node Worker1 Down

Summary	Summary Report												
Name:	Summary Report												
Comments:													
Write results to file / Read from file													
Filename						Browse	Log/Display Only	y: 🗌 Errors 📄	Successes	Configure			
Label	#Samples	Average	Min	Max	Std. Dev.	Error %	Throughput	Received KB/s	Sent KB/sec	Avg. Bytes			
HTTP Request	500	96	69	410	22.80	0.00%	10.4/sec	8.63	1.17	853.0			
TOTAL	500	96	69	410	22.80	0.00%	10.4/sec	8.63	1.17	853.0			

Gambar 4.40 Hasil Testing Ke-2 Node Worker1 Down

Summary Report												
Name:	Summary Report											
Comments:												
Write results to file / Read from file												
Filename						Browse	Log/Display Onl	y: 🗌 Errors 📄	Successes	Configure		
Label	# Samples	Average	Min	Max	Std. Dev.	Error %	Throughput	Received KB/s	Sent KB/sec	Avg. Bytes		
HTTP Request	500	96	67	1069	52.71	0.00%	10.3/sec	8.60	1.17	853.0		
TOTAL	500	96	67	1069	52.71	0.00%	10.3/sec	8.60	1.17	853.0		

Gambar 4.41 Hasil Testing Ke-3 Node Worker1 Down

Tabel 4.1 Hasil Pengujian High Availability Ketika Node Worker1 Down

Pengujian Ke :	Waktu Failover
Pertama	0
Kedua	0
Ketiga	0
Rata - rata	0

Berdasarkan pengujian diatas presentasi error yang didapatkan adalah 0% karena server tidak membutuhkan waktu untuk *failover* untuk sebuah *error*.

Pengujian kedua dilakukan dengan mematikan semua node worker.



Gambar 4.42 Perintah Pengecekan Status Node

Hasil yang diperoleh sebagai berikut

Summar	Summary Report												
Name:	Summary Report												
Comments:													
Write results to file / Read from file													
Filename					Browse	Log/Display	Only: 🔲 Err	rors 📃 Suce	cesses C	onfigure			
Label	# Samples	Average	Min	Max	Std Dev	Error %	Throughput	Received K	Sent KB/sec	Ava Bytes			
HTTP Req	50000	29	2	2193	43.41	0.07%	1948.6/sec	1624.71	220.59	853.8			
TOTAL	50000	29	2	2193	43.41	0.07%	1948.6/sec	1624.71	220.59	853.8			

Gambar 4.43 Hasil Testing Ke-1 Manager dan Worker1 Down

Summar	Summary Report													
Name:	Summary Report													
Comments:														
Write results to file / Read from file														
Filename					Browse	Log/Display	Only: 🗌 Eri	rors 🗌 Suc	cesses C	onfigure				
Label	# Samples	Average	Min	Max	Std. Dev.	Error %	Throughput	Received K	Sent KB/sec	Avg. Bytes				
HTTP Req	50000	29	2	2193	43.41	0.07%	1948.6/sec	1624.71	220.59	853.8				
TOTAL	50000	29	2	2193	43.41	0.07%	1948.6/sec	1624.71	220.59	853.8				

Gambar 4.44 Hasil Testing Ke-2 Manager dan Worker1 Down

Summar	Summary Report													
Name:	Summary Report													
Comments:														
Write results to file / Read from file														
Filename					Browse	Log/Display	Only: 🔲 Err	rors 🗌 Suco	cesses Co	onfigure				
Label		A			Old Day	F 0/	There is a set	Deschartic	0	Ave Dates				
Laber	# Samples	Average	MIN	Max	Sid. Dev.	EII01%	Throughput	Received K	Sent KB/Sec	Avg. Bytes				
HTTP Req	50000	29	2	2193	43.41	0.07%	1948.6/sec	1624.71	220.59	853.8				
TOTAL	50000	29	2	2193	43.41	0.07%	1948.6/sec	1624.71	220.59	853.8				

Gambar 4.45 Hasil Testing Ke-3 Manager dan Worker1 Down

Pengujian Ke :	Waktu Failover
Pertama	0.07
Kedua	0.07
Ketiga	0.07
Rata - rata	0.07

Tabel 4.2 Hasil Pengujian High Availability Node Manager & Worker1 Down

Berdasarkan pengujian diatas presentasi error yang didapatkan adalah 0.07% karena server tidak membutuhkan waktu untuk failover untuk sebuah error.

Pengujian ketiga dilakukan dengan mematikan node manager.

Gambar 4.46 Perintah Pengecekan Status Node

Hasil yang diperoleh sebagai berikut :

Summar	Summary Report												
Name:	Summary Report												
Comments:													
Write results to file / Read from file													
Filename					Browse	Log/Display	Only: 🗌 Err	rors 🗌 Suco	cesses Co	onfigure			
Label	# Samples	Average	Min	Max	Std. Dev.	Error %	Throughput	Received K	Sent KB/sec	Avg. Bytes			
HTTP Req	50000	39	1	504	38.98	0.07%	1796.2/sec	1497.87	203.33	853.9			
TOTAL	50000	39	1	504	38.98	0.07%	1796.2/sec	1497.87	203.33	853.9			

Gambar 4.47 Hasil Ke-1 Testing Node Manager Down

Summar	Summary Report													
Name:	Summary Re	eport												
Comments:														
Write results to file / Read from file														
Filename					Browse	Log/Display	only: 🔲 Err	rors 🗌 Suco	cesses Co	onfigure				
Label	# O	A	115-	Mari	Otd David	E 0/	Therewalkersk	Deschard IC	0	Aug. Didag				
Label	# Samples	Average	MIN	Max	Std. Dev.	Effor %	Inrougnput	Received K	Sent KB/Sec	Avg. Bytes				
HTTP Req	50000	39	1	504	38.98	0.07%	1796.2/sec	1497.87	203.33	853.9				
TOTAL	50000	39	1	504	38.98	0.07%	1796.2/sec	1497.87	203.33	853.9				

Gambar 4.48 Hasil Testing Ke-2 Node Manager Down

Summary Report										
Name:	Summary Report									
Comments:	Comments:									
Write results to file / Read from file										
Filename					Browse	Log/Display	Only: 🗌 Err	rors 🗌 Suco	cesses Co	onfigure
Label	#Samples	Average	Min	Max	Std. Dev.	Error %	Throughput	Received K	Sent KB/sec	Avg. Bytes
HTTP Req	50000	39	1	504	38.98	0.07%	1796.2/sec	1497.87	203.33	853.9
TOTAL	50000	39	1	504	38.98	0.07%	1796.2/sec	1497.87	203.33	853.9

Gambar 4.49 Hasil Testing Ke-3 Node Manager Down

Pengujian Ke :	Waktu Failover
Pertama	0.07
Kedua	0.07
Ketiga	0.07
Rata - rata	0.07

Tabel 4.3 Hasil Pengujian High Availability Ketika Node Manager Down

Berdasarkan pengujian diatas waktu yang diperlukan sistem untuk melakukan failover error selama 0.07 detik. Sistem juga dapat melakukan failover virtual ip ke node yang lain ketika node manager mengalami downtime. Sistem yang dibangun tetap dapat beroperasi ketika 2/3 mengalami crash. Koneksi client akan secara otomatis dialihkan ke node yang up, sehingga sistem tidak mengalami downtime. Pengujian keempat dilakukan dengan mematikan semua node, baik itu node manager maupun node worker

Pertama - tama peneliti mematikan node worker1 terlebih dahulu.

root@worker2:/home/ghofarnugroho# crm status						
Stack: corosync						
Current DC: worker2 (version 1.1.18-2b07d5c5a9) - partition with quorum						
Last updated: Mon May 16 06:31:56 2022						
Last change: Fri May 13 16:54:31 2022 by root via cibadmin on manager						
3 nodes configured						
l resource configured						
Online: [manager worker2]						
OFFLINE: [workerl]						
Full list of resources:						
<pre>virtual_ip (ocf::heartbeat:IPaddr2): Started manager</pre>						

Gambar 4.50 Perintah Pengecekan Status node

Hasil setelah node worker1 dimatikan

Summary Report										
Name:	Summary Report									
Comments:	Comments:									
Write results to file / Read from file										
Filename						Browse	Log/Display Only	y: Errors	Successes	Configure
Label	# Commiss	August 10	Min	Man	Ohd Davi	E 9/	These shout	Deserved I/D/s	On at I/D/ana	Aug Dates
Laber	# samples	Average	MIN	Max	Std. Dev.	EITOI %	Throughput	Received KB/S	Sent KB/Sec	Avg. Bytes
HTTP Request	1060	66	43	1174	37.15	0.00%	15.0/sec	12.51	1.70	853.0
TOTAL	1060	66	43	1174	37.15	0.00%	15.0/sec	12.51	1.70	853.0

Gambar 4..51 Hasil Testing Setelah Node Worker1 Dimatikan

Selanjutnya peneliti mematikan node worker2.



Gambar 4.52 Perintah Pengecekan Status node

Summary Report
Name: Summary Report
Comments:
Write results to file / Read from file
Filename
Filename
Browse...
Log/Display Only: Errors Successes Configure

Hasil setelah node worker1 dan node worker2 dimatikan

Gambar 4.53 Hasil Testing Node Worker1 dan Worker2 Dimatikan

Std. Dev.

Error %

Throughput Received KB/s

15.2/sec 15.2/sec Avg. Bytes

853.(853.(

Sent KB/

12.68 12.68

Yang terakhir peneliti mematikan node manager

Max

Samples 47

47

Average

Min

Label HTTP Request

TOTAL

Ð	Ghofar1
Ð	Ghofar2
Ð	Ghofar3

Gambar 4.54 Semua Node Dimatikan

Hasil setelah semua node dimatikan

View Results in Table										
Name:	View Results in Table									
Comments:	Comments:									
Write result	Write results to file / Read from file									
Filename						Brow	se Log/Display	Only: 🔲 Errors	Successes	Configure
Sample #		Start Time	Thread Name	Label	Sample Time(m	Status	Bytes	Sent Bytes	Latency	Connect Time(
	1	14:22:41.448	Thread Group 1-1	HTTP Request	21040	8	2805	0	0	21040
	2	14:23:02.490	Thread Group 1-1	HTTP Request	21036	8	2805	0	0	21036
	3	14:23:23.529	Thread Group 1-1	HTTP Request	21042	8	2805	0	0	21042
	4	14:23:44.572	Thread Group 1-1	HTTP Request	21025	8	2805	0	(21025

Gambar 4.55 Hasil View Result in Table Setelah Semua Node Dimatikan

Summary Report										
Name:	Summary Report									
Comments:	Comments:									
Write results to file / Read from file										
Filename	Filename Browse Log/Display Only: Errors Successes Configure							Configure		
Lobal	# Complea	Aueroae	Min	Max	Otd Day	Error W	Throughput	Dessived KD/s	Cant KD/aca	Aug Diteo
Laver UTTR Request	# Samples	Average	21026	MdX 210.40	Siu. Dev.	100.00%	2.0/min	Received KD/S	Selic KD/Sec	Avg. bytes
TOTAL	2	21038	21030	21040	2.00	100.00%	2.9/min	0.13	0.00	2805.0

Gambar 4.56 Hasil Summary Report Setelah Semua Node Dimatikan



Gambar 4.57 Website Down Setelah Semua Node Dimatikan

Berdasarkan pengujian diatas cluster tidak dapat melakukan failover dikarenakan semua node mati. untuk mengatasi hal itu maka sebaiknya node diletakkan pada environment yang berbeda untuk memperoleh *redundancy* sehingga kemungkinan semua node down rendah.

Summary Hasil testing High Avaibility :

Tabel 4.4 Summary Hasil Pengujian

Pengujian Ke	Waktu Failover	Waktu Failover	Waktu Failover
:	saat Worker1	saat Manager	saat Manager dan
	Down	Down	Worker1 Down
Pertama	0	0.07	0.07
Kedua	0	0.07	0.07
Ketiga	0	0.07	0.07
Rata - rata	0	0.07	0.07

Berdasarkan pengujian yang dilakukan, dapat dikatakan sistem yang dibangun memiliki tingkat ketersediaan yang tinggi. Sistem tetap dapat melayani request user walaupun 2 node down. Error yang terjadi dikarenakan adanya perpindahan replika image container dari node yang down ke node yang up. Error yang terjadi masih bisa ditolerir karena sangat kecil sekali yaitu 0.07 detik, sedangkan website dummy tetap bisa diakses oleh user ketika 2 node mengalami down

4.2.3 Pengujian Replika Service

Pengujian ini dilakukan untuk mengetahui apakah docker swarm dapat menjaga agar replica tetap sesuai dengan yang telah kita tentukan. Pengujian ini dilakukan dengan membuat service dengan jumlah replika 5, kemudian dua node worker akan dimatikan.

Pada awalnya service akan dibagi secara merata kepada seluruh node yang tergabung pada docker swarm cluster seperti pada gambar 4.58

root@manager:/home/gho	ofarnugroho/nginx# docker servi	ce ps nginx	
ID NAME	IMAGE	NODE	DESIRED STATE
CURRENT STATE	ERROR PORTS		
fxolw93kmpja nginx.1	1 127.0.0.1:5000/nginx:latest	worker2	Running
Running 20 seconds ag	go		
04ix97hb4kdl nginx.2	2 127.0.0.1:5000/nginx:latest	manager	Running
Running 20 seconds ag	go		
ylr6f668rrkb nginx.3	3 127.0.0.1:5000/nginx:latest	workerl	Running
Running 20 seconds ag	go		
buu3skbu7ecz nginx.4	4 127.0.0.1:5000/nginx:latest	worker2	Running
Running 20 seconds ag	go		
w2bcu2r2eac6 nginx.5	5 127.0.0.1:5000/nginx:latest	workerl	Running
Running 20 seconds ag	Jo		

Gambar 4.58 Perintah Pengecekan Cluster

Kemudian jika salah satu node worker crash seperti pada gambar

4.59



Gambar 4.59 Perintah Pengecekan Status Node

Maka node manager akan menjaga agar jumlah replika yang running tetap sesuai dengan yang kita tentukan dengan membuat container baru pada node yang up seperti pada gambar 4.60

root@manager:/h	nome/ghofarnugr	oho/nginx#	docker service	ps nginx	
ID	NAME	IMAGE		NODE	DESIRED STA
TE CURRENT ST	TATE	ERROR	PORTS		
fxolw93kmpja	nginx.l	127.0.0.1:5	5000/nginx:late	st worker2	Running
Running 2	minutes ago				
04ix97hb4kdl	nginx.2	127.0.0.1:5	5000/nginx:late	st manager	Running
Running 2	minutes ago				
mgoeghkskhhx	nginx.3	127.0.0.1:5	5000/nginx:late	st manager	Running
Running 23	8 seconds ago				
ylr6f668rrkb	_ nginx.3	127.0.0.1:5	5000/nginx:late	st workerl	Shutdown
Running 2	minutes ago				
buu3skbu7ecz	nginx.4	127.0.0.1:5	5000/nginx:late	st worker2	Running
Running 2	minutes ago				
iq8i8wx6rsgg	nginx.5	127.0.0.1:5	5000/nginx:late	st worker2	Running
Running 23	3 seconds ago				
w2bcu2r2eac6	_ nginx.5	127.0.0.1:5	5000/nginx:late	st workerl	Shutdown
Running 2	minutes ago		_		

Gambar 4.60 Perintah Pengecekan Jumlah Replika

Dengan metode *auto replicated* maka jumlah client yang dapat ditangani oleh sistem akan tetap sama meskipun ada salah satu node yang *crash* karena *docker swarm manager* menjaga agar jumlah replika tetap sama.