BAB III
METODE PENELITIAN

Metodologi Penelitian berisi uraian diagram alur penelitian. Diagram alur penelitian menjelaskan mengenai tahap-tahap penelitian. Dalam penelitian ini diperlukan pula alat pendukung untuk menunjang penelitian. Penelitian ini juga membutuhkan topologi yang berfungsi sebagai objek pengambilan data pada proses penelitian.

3.1 ALAT YANG DI GUNAKAN DALAM PENELITIAN

3.1.1 Perangkat Lunak

Perangkat lunak sebagai *tool* dan alikasi yang digunakan pada penelitian ini dapat dilihat pada tabel 3.1.

Tabel 3. 1 Perangkat Lunak

No	*Software*	Versi	Fungsi
1	*Virtual*box	7.04	Tempat penginstalan proxmox
2	Ubuntu	22.04	Sistem operasi untuk ceph *cluster* dan raid
3	FIO	3.35	*Tool*s pengambilan data
4	Proxmox	8.0	Wadah penginstalan VM

Tabel 3. 2 Spesifikasi *Virtual Machine*

Virtual Machine			
Node 1 *Server* Utama	ceph-*client*	OS	Ubuntu *Server* 22.04 LTS
		Processor	Intel(R) Xeon(R) CPU E5-1650 v3 @ 3.50GHz
		System Memori (RAM)	5 Gb
		Storage(*Hardisk*)	20 GB
		IP	10.50.0.147/24

Virtual Machine			
	raid-*client*	OS	Ubuntu *Server* 22.04 LTS
		Processor	Intel(R) Xeon(R) CPU E5-1650 v3 @ 3.50GHz
		System Memori (RAM)	5 Gb
		Storage(*Hardisk*)	20 GB
		IP	10.50.0.148/24
Node 2 *Storage Cluster*	ca-ceph1	OS	Ubuntu *Server* 22.04 LTS
		Processor	Intel(R) Xeon(R) CPU E5-1650 v3 @ 3.50GHz
		System Memori (RAM)	2 Gb
		Storage(*Hardisk*)	30 GB
		IP	10.50.50.11/24
	ca-ceph2	OS	Ubuntu *Server* 22.04 LTS
		Processor	Intel(R) Xeon(R) CPU E5-1650 v3 @ 3.50GHz
		System Memori (RAM)	2 Gb
		Storage(*Hardisk*)	30 GB
		IP	10.50.50.12/24
	ca-ceph3	OS	Ubuntu *Server* 22.04 LTS
		Processor	Intel(R) Xeon(R)

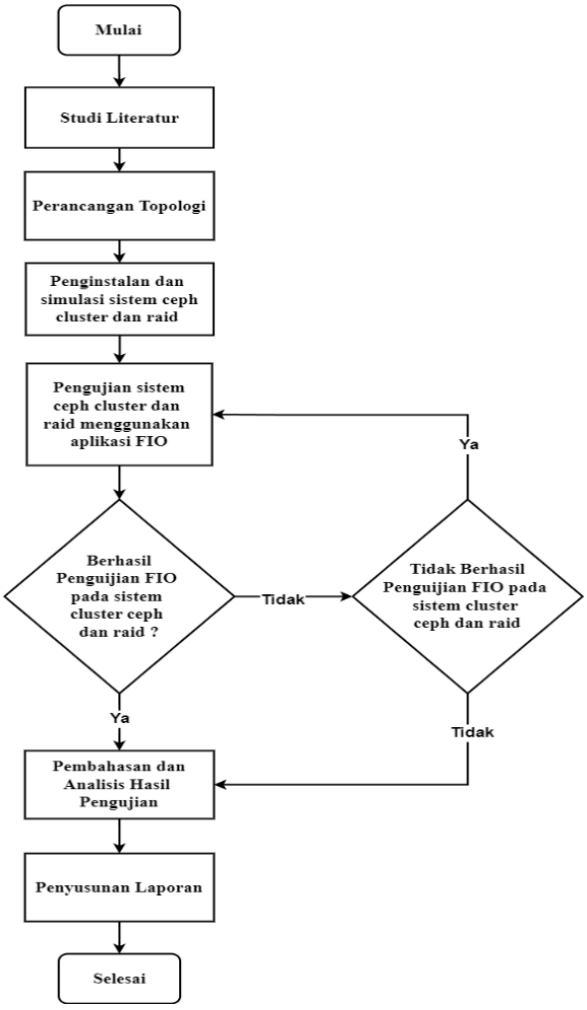
Virtual Machine			
			CPU E5-1650 v3 @ 3.50GHz
		System Memori (RAM)	2 Gb
		Storage(*Hardisk*)	30 GB
		IP	10.50.50.13/24
	ca-raid	OS	Ubuntu *Server* 22.04 LTS
		Processor	Intel(R) Xeon(R) CPU E5-1650 v3 @ 3.50GHz
		System Memori (RAM)	2 Gb
		Storage(*Hardisk*)	50 GB
		IP	10.50.50.15/24

Pada penelitian ini menggunakan spesifikasi pada tabel 3.2 yang menggunakan dua *node* server dan sebuah *cluster storage*. *Node* 1 berperan sebagai *server* utama, menjalankan sistem operasi Ubuntu *server* 22.04 LTS dengan prosesor Intel Xeon 3.50GHz, RAM 5GB, dan penyimpanan 20GB. *server* ini terhubung ke jaringan dengan alamat IP 10.50.0.147/24 dan menjalankan *clien* ceph dan raid. *Node* 2 didedikasikan untuk *cluster storage*, juga menggunakan ubuntu *server* 22.04 LTS dengan prosesor Intel Xeon 3.50GHz, tetapi dengan RAM 2GB dan penyimpanan 30GB. *Node* ini memiliki alamat IP 10.50.0.112/24 dan menjalankan komponen ca-ceph1 dan ca-ceph2. Di sini, ca-ceph1 dan ca-ceph2 kemungkinan merujuk pada dua *instance* ceph *object gateway* yang menyediakan akses objek *storage* ke jaringan. Selain itu, terdapat tiga mesin *virtual*: ca-ceph3, ca-ceph4, dan ca-raid. Semua mesin *virtual* menggunakan ubuntu server 22.04 LTS dan prosesor Intel Xeon 3.50GHz. ca-ceph3 dan ca-ceph4 memiliki RAM 2GB dan penyimpanan 30GB, sedangkan ca-raid memiliki RAM 2GB dan penyimpanan 50GB. Alamat IP ca-ceph3 adalah 10.50.0.13/24,

sementara alamat ip ca-raid tidak dicantumkan. Sistem ini dirancang untuk penelitian yang berfokus pada ceph dan raid, dengan *Node* 1 sebagai pusat kendali dan *node* 2 menyediakan penyimpanan.

3.1 ALUR PENELITIAN

Dalam penulisan skripsi ini, terdapat beberapa tahapan yang harus dilakukan agar penelitian ini dapat disusun dengan baik, mulai dari mencari studi literatur hingga penulisan laporan. Tahapan-tahapan pada skripsi ini akan dijelaskan pada gambar 2.6.



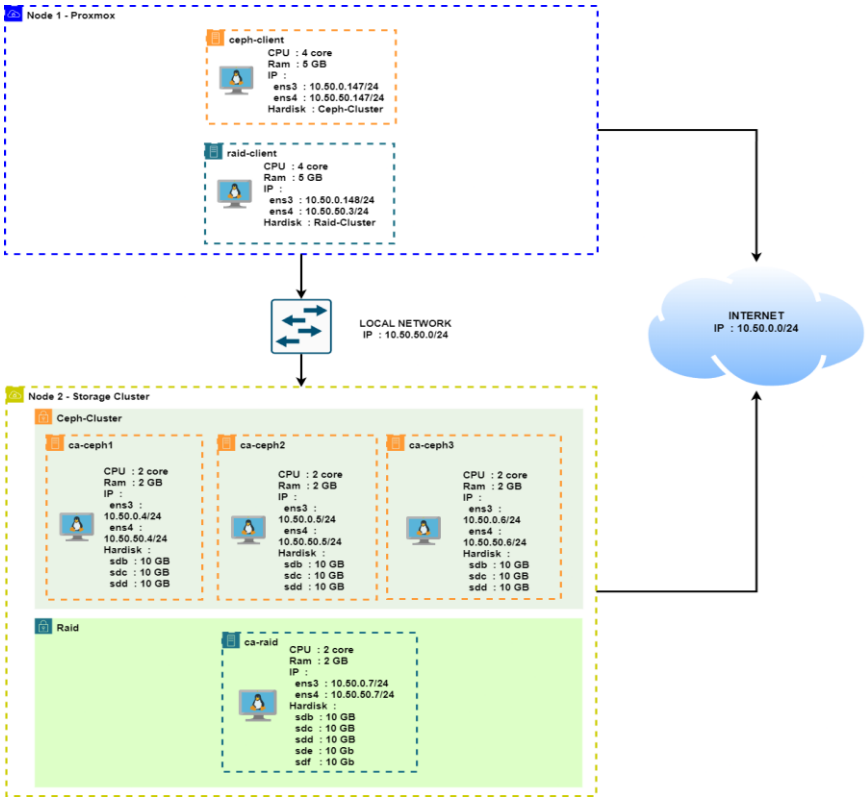
Gambar 2. 6 Flowchart Alur Penelitian.

Pada gambar 2.6 menjelaskan alur dari penelitian yang di mulai tahap pertama adalah pencarian studi literatur yang relevan sebagai landasan teoritis. Setelah itu, penelitian dilanjutkan ke perancangan topologi, termasuk pemilihan perangkat

keras dan infrastruktur yang sesuai. Topologi terdiri dari dua *node* utama: *Node* 1 dan *Node* 2, untuk menguji berbagai aspek sistem. Langkah berikutnya adalah mengkonfigurasi dan menginstal ceph *cluster* dan raid pada *Node* 2, dengan ceph *cluster* menggunakan tiga mesin *virtual*, masing-masing dengan tiga *disk* berukuran 10 GB, dan raid menggunakan satu mesin *virtual* dengan lima *disk* berukuran 10 GB. Pengujian dilakukan menggunakan aplikasi FIO pada masing-masing mesin *virtual* di *node* 1 dalam lingkungan proxmox. Pengujian ini mengukur parameter iops, *bandwidth*, dan latensi. Setelah pengujian, hasil dianalisis dan disusun dalam laporan yang menggabungkan semua analisis untuk kajian lebih lanjut.

3.2 TOPOLOGI JARINGAN

Pada penelitian ini menggunakan topologi jaringan seperti pada gambar 2.7 yang menjelaskan proses dari perancangan sistem yang di gunakan dalam hal ini penginstalan dan konfigurasi dari sistem ceph *cluster* dan raid dalam proxmox *envirotment*.



Gambar 2. 7 Topologi Jaringan Ceph *Cluster* dan Raid.

Pada gambar 2.7 merupakan topologi jaringan yang membandingkan kedua sistem ini dengan dua tahap konfigurasi yang di install pada dua *node* yang berbeda, Di dalam *node* pertama sudah terdapat sistem proxmox yang berfungsi untuk mengatur *virtual Machine* ceph *client* dan raid *client* yang masing—masing sudah di install *tool*s FIO, kedua *virtual machine* ini berfungsi sebagai alat untuk melakukan pengetesan kinerja dari kedua sistem tersebut. Ketika penginstalan kedua *virtual machine* ini akan di arahkan untuk penyimpanan-nya agar di install pada kedua masing—masing sistem penyimpanan tersebut. Kemudian pada *node* ke 2 digunakan untuk menginstall *virtual machine* untuk kedua sistem ini ceph *cluster* dan raid, untuk sistem ceph *cluster* mempunyai tiga *virtual machine* yang di konfigurasi dengan besar penyimpannya jika di gabung di masing—masing *virtual machine* mencapai 90 Gb.

3.3 KONFIGURASI PERANGKAT

3.3.1 Penginstalan dan Konfigurasi Ceph *Cluster*

Penginstalan ceph di lakukan pada tiga *node* yang nantinya *node* 1 akan menangani kedua *node* yang ada.

```
apt-get install software-properties-common
add-apt-repository cloud-archive:wallace
apt-get update
apt-get install ceph-deploy
```

Perintah akan di jalankan ke semua *node* yang ada pada ceph *cluster*, perintah berfungsi untuk menginstall ceph-*deploy* yang berfungsi untuk mengotomatiskan proses pemasangan, konfigurasi, dan pemeliharaan kluster ceph. Kemudian penginstalan di lanjutkan pada *node* 1, dari *node* 1 akan mendistribusikan penginstalan pada *node* 2 dan *node* 3.

```
ceph-deploy new ca-ceph1 ca-ceph2 ca-ceph3
ceph-deploy install ca-ceph1 ca-ceph2 ca-ceph3
ceph-deploy mon create-initial
ceph-deploy mgr create ca-ceph1 ca-ceph2 ca-ceph3
```

Fungsi perintah di baris pertama digunakan untuk membuat konfigurasi *cluster* ceph baru dengan *node* ca-ceph1, ca-ceph2 dan ca-ceph3, kemudian

perintah di lanjutkan dengan menginstal perangkat lunak ceph di semua *node* yang ada, dengan ini semua *node* yang terlibat akan memiliki paket-paket yang di perlukan untuk menjalankan ceph. Perintah di lanjutkan dengan menginstal ceph mon dan mgr pada *cluster* ceph yang berfungsi untuk memonitor sistem ceph dan ceph mgr digunakan untuk menginstall *module* yang di perlukan untuk menjalankan sistem ceph *cluster*.

```
ceph-deploy osd create --data /dev/vdb ca-ceph1
ceph-deploy osd create --data /dev/vdb ca-ceph2
ceph-deploy osd create --data /dev/vdb ca-ceph3
ceph-deploy osd create --data /dev/vdc ca-ceph1
ceph-deploy osd create --data /dev/vdc ca-ceph2
ceph-deploy osd create --data /dev/vdc ca-ceph3
ceph-deploy osd create --data /dev/vdd ca-ceph1
ceph-deploy osd create --data /dev/vdd ca-ceph2
ceph-deploy osd create --data /dev/vdd ca-ceph3
```

Perintah digunakan untuk membuat konfigurasi *object storage daemon* (OSD) pada tiga *node* yang berbeda yaitu ca-ceph1, ca-ceph2, dan ca-ceph3 dengan menggunakan perangkat penyimpanan atau *disk* /dev/vdb, /dev/vdc, dan /dev/vdd yang ada di masing – masing *node*.

```
systemctl enable ceph.target
systemctl enable ceph-mon.target
systemctl enable ceph-osd.target
systemctl enable ceph-mgr.target
systemctl enable ceph-mds.target
systemctl enable ceph-radosgw.target
systemctl start ceph.target
```

```
systemctl start ceph-mon.target
systemctl start ceph-osd.target
systemctl start ceph-mgr.target
systemctl start ceph-mds.target
systemctl start ceph-radosgw.target
```

Setelah perintah untuk menginstal ceph sudah selesai dijalankan, langkah selanjutnya adalah dengan mengaktifkan dan menjalankan semua *service* ceph menggunakan perintah di atas. Untuk menjalankan perintah tersebut, cukup dijalankan pada *node* ca-ceph1 saja. Hal ini akan memastikan bahwa semua komponen ceph, seperti monitor, OSD, dan MDS, telah diaktifkan dan berjalan dengan benar.

```
ceph health
ceph osd status
ceph -s
```

Langkah terakhir dengan memverifikasi kondisi sistem ceph dengan perintah yang ada di atas.

3.3.2 Penginstalan dan Konfigurasi RAID

Penginstalan raid dilakukan pada satu *node* yang di dalamnya ada lima *hard disk* yang sudah dipasangkan ke dalam *node* 1 tersebut. Untuk langkah-langkah penginstalan raid, dapat dilakukan dengan perintah di bawah ini. Proses ini akan membuat suatu *volume* raid yang terdiri dari lima *hard disk* tersebut, sehingga dapat meningkatkan keamanan dan kehandalan penyimpanan data.

```
apt-get install mdadm
mdadm --create --verbose /dev/md0 --level=6 --raid-
Devices=5 /dev/vdb /dev/vdc /dev/vdd /dev/vde /dev/vdf
cat /proc/mdstat
mkfs.ext4 /dev/md0
mkdir /mnt/raid6
mount /dev/md0 /mnt/raid6
nano /etc/fstab  # --> masukin command di bawah
/dev/md0    /mnt/raid6   ext4    defaults    0    0
```

Perintah-perintah di atas digunakan untuk mengatur raid 6 pada *node* ca-raid. Pertama, "*apt-get install mdadm*" berfungsi untuk menginstal perangkat lunak mdadm untuk mengelola raid. Kemudian, perintah "*mdadm --create --verbose /dev/md0 --level=6 --raid-Devices=5 /dev/vdb /dev/vdc /dev/vdd /dev/vde /dev/vdf*" membuat perangkat raid 6 bernama /dev/md0 dengan lima *disk*

(/dev/vdb, /dev/vdc, /dev/vdd, /dev/vde, dan /dev/vdf). Untuk memeriksa status raid yang baru dibuat, digunakan perintah *cat /proc/mdstat*. Setelah itu, *mkfs.ext4 /dev/md0* memformat perangkat raid dengan sistem *file* ext4. Direktori *mount point* dibuat dengan perintah *mkdir /mnt/raid6*, dan kemudian perangkat raid dipasang ke direktori tersebut menggunakan *mount* /dev/md0 /*mnt/raid6*. Agar raid ini tetap terpasang setelah sistem *reboot*, konfigurasi *fstab* diubah dengan membuka *file* /*etc*/*fstab* menggunakan *nano* /etc/fstab dan menambahkan baris */dev/md0 /mnt/raid6 ext4 defaults 0 0*.

```
##install nfs server
apt install nfs-kernel-server
chown -R nobody:nogroup /mnt/raid6
chmod 777 /mnt/raid6
nano /etc/exports
/mnt/raid6  10.50.50.0/24(rw,sync,no_subtree_check)
systemctl restart nfs-kernel-server
```

Perintah-perintah di atas menginstal dan mengonfigurasi NFS *server* pada *node* ca-raid. Pertama, "*apt install nfs-kernel-server*" menginstal NFS *server*. Kemudian, "*chown -R nobody:nogroup /mnt/raid6*" dan "*chmod 777 /mnt/raid6*" mengatur kepemilikan dan izin direktori */mnt/raid6*. Selanjutnya, nano "/*etc/exports*" membuka *file* konfigurasi NFS, di mana */mnt/raid6 10.50.50.0/24(rw,sync,no_subtree_check)* ditambahkan untuk berbagi direktori tersebut di jaringan 10.50.50.0/24. Terakhir, "*systemctl restart nfs-kernel-server*" me-*restart* layanan NFS untuk menerapkan perubahan.

3.3.3 Penginstalan dan Konfigurasi FIO

Aplikasi fio di gunakan untuk melakukan pengujian kecepatan jaringan dan *disaster recovery* yang ada pada sistem penyimpanan ceph dan raid, aplikasi ini di install pada *client* dari sistem penyimpanan ceph dan raid.

```
apt update -y
apt -y install fio
systemctl status fio.service
```

Setelah melakukan penginstalan langkah selanjutnya melakukan konfigurasi pada aplikasi fio yang berfungsi sebagai rujuan oleh sistem fio untuk melakukan skenario pengujian pada penelitian ini. Berikut konfigurasi fio yang digunakan beserta dengan fungsi dari masing – masing parameter yang ada.

```
[global]
size=1GB
direct=1
rw=randrw
bs=1k
ioengine=libaio
iodepth=256
runtime=120
numjobs=1
time_based=1
group_reporting=1
[iops-test-job]
filename=/root/hasil
```

Fungsi dari [*global*] di gunakan untuk mendefinisikan pengaturan *global* yang akan di ikuti ketika program di jalankan, *Size* berfungsi menetapkan jumlah total data yang di kirim. ”*direct*” berfungsi untuk Mengaktifkan I/O langsung, melewati *buffer cache* untuk memastikan data dibaca/ditulis langsung ke *disk*, “rw” berfungsi Menentukan pola I/O yang digunakan. '*Randrw*' menunjukkan campuran operasi baca dan tulis acak, “*bs*” digunakan untuk Menetapkan ukuran blok untuk operasi I/O menjadi 1 kilobyte. Lalu “*ioengine*” yang berfungsi Menentukan mesin I/O yang digunakan. 'Libaio' adalah Linux AIO (*asynchronous I/O*), *iodepth* Menetapkan kedalaman antrian I/O menjadi 256, artinya bisa ada hingga 256 operasi I/O yang berjalan sekaligus. *Runtime* Menetapkan durasi pengujian menjadi 120 detik, numjobs Menentukan jumlah pekerjaan yang dijalankan secara bersamaan. Di sini, ditetapkan menjadi 1. Selanjutnya untuk *time_based* akan Menunjukkan bahwa tes harus dijalankan selama waktu yang

ditentukan (ditetapkan oleh *'runtime'*), daripada berhenti setelah sejumlah data tertentu telah di *transfer*. *Group_reporting* berfungsi Mengaktifkan pelaporan kelompok, artinya hasil dari semua pekerjaan akan dilaporkan bersama. [*iops-test-job*] program yang akan mendefinisikan pekerjaan khusus yang diberi nama *'iops-test-job*' dan parameter yang terakhir *file*name=/root/hasil Menentukan *file* atau perangkat yang digunakan untuk tes. Di sini, tes akan dilakukan pada *file* *'/root/hasil'*. Untuk menjalankan konfigurasi bisa menggunakan perintah di bawah ini.

```
fio test.fio
```

Pengujian di lakukan pada *virtual machine* yang dimana vm tersebut telah di install pada proxmox *envirotment* dengan konfigurasi *storage* yang di ambil dari sistem penyimpanan ceph dan raid. Proses pengujian akan berlangsung dengan parameter – parameter seperti yang telah di konfigurasi pada *script* di atas.

3.4 SKENARIO PENGUJIAN

Pada penelitian ini, digunakan dua skenario pengujian, yaitu pengujian kecepatan *transfer* menggunakan aplikasi fio dan pengujian *disaster recovery* pada sistem penyimpanan data ceph dan raid.

3.4.1 Kecepatan *Transfer*

Pada pengujian kecepatan *transfer* ini digunakan aplikasi FIO untuk mengukur kecepatan *transfer* data yang dikirim pada kedua sistem penyimpanan tersebut. Sebelum itu, penjelasan blok diagram dapat dilihat pada gambar 3.1 agar mempermudah pemahaman untuk skenario pengujian kecepatan *transfer* data menggunakan aplikasi FIO.

Gambar 3. 1 Diagram Alur Kecepatan *Transfer*

Pada Gambar 3.1 menjelaskan tahapan skenario pengujian pada FIO menggunakan parameter kecepatan *transfer*. Proses dimulai dengan tahap persiapan, di mana parameter-parameter yang akan digunakan untuk menguji

sistem penyimpanan dikonfigurasi. Langkah ini melibatkan pemilihan ukuran data, jumlah pekerjaan, dan ukuran blok yang sesuai dengan skenario pengujian yang diinginkan. Langkah ini bertujuan untuk mendapatkan parameter-parameter yang sesuai dengan kecepatan *transfer* yang telah ditetapkan. Selanjutnya, proses dilanjutkan dengan memasukkan perintah FIO untuk menjalankan *file* konfigurasi sesuai dengan skrip di bawah ini. Saat perintah dijalankan, FIO akan memproses pengambilan data sesuai dengan konfigurasi yang telah diatur sebelumnya. Proses ini berlangsung hingga parameter-parameter yang telah dikonfigurasi selesai dijalankan, mencakup berbagai ukuran data, jumlah pekerjaan, dan ukuran blok untuk memastikan hasil yang komprehensif. Hasil data yang dihasilkan oleh aplikasi FIO kemudian dapat digunakan dan dievaluasi untuk mendapatkan pemahaman mendalam tentang performa sistem penyimpanan dalam konteks kecepatan *transfer* data. Evaluasi ini mencakup penilaian terhadap sejauh mana parameter kecepatan *transfer* telah terpenuhi, serta analisis perbandingan antara skenario yang berbeda untuk mengidentifikasi tren performa dan potensi *bottleneck*.

3.4.1.1 Skenario Pengujian IOPS Ceph *Cluster* dan Raid

Skenario pengujian iops merupakan pengujian yang di lakukan untuk melihat kinerja sistem ceph *cluster* dan raid dengan menggunakan parameter – parameter yang ada, Perbandingan ini akan memberikan gambaran yang jelas mengenai efisiensi dan performa dari kedua sistem penyimpanan tersebut dalam berbagai skenario penggunaan. Dalam penelitian ini, tabel tersebut akan berfungsi sebagai acuan utama dalam proses pengujian IOPS yang akan dilakukan menggunakan aplikasi FIO (*Flexible I/O Tester*).

Tabel 3. 3 Skenario IOPS dengan Jumlah kerja (1) dan Ukuran Blok (1)

IOPS							
No	**Jumlah *Size* (Gb)**	**Jumlah Kerja**	**Ukuran Blok (k)**	**Ceph *Cluster***		**Raid**	
				Read	***Write***	***Read***	***Write***
1	1	1	1				
2	2	1	1				
3	3	1	1				
4	4	1	1				
5	5	1	1				

Tabel 3.3 menunjukkan hasil pengujian iops pada ceph *cluster* dan raid dengan variasi ukuran data (1-5 GB), jumlah kerja (1 th*read*), dan ukuran blok (1 kB). Pengujian dilakukan dalam lima tahap, masing-masing dengan ukuran data yang meningkat dari 1 GB hingga 5 GB, sementara parameter lainnya tetap sama. Hasil iops untuk operasi baca dan tulis pada kedua jenis penyimpanan dicatat untuk setiap tahap guna membandingkan performa mereka saat ukuran data meningkat.

Tabel 3. 4 Skenario IOPS dengan Jumlah kerja (1) dan Ukuran Blok (2)

IOPS							
No	**Jumlah *Size* (Gb)**	**Jumlah Kerja**	**Ukuran Blok (k)**	**Ceph *Cluster***		**Raid**	
				Read	***Write***	***Read***	***Write***
1	1	1	2				
2	2	1	2				
3	3	1	2				
4	4	1	2				
5	5	1	2				

Tabel 3.4 menguraikan pengujian iops untuk ceph *cluster* dan raid dengan variasi ukuran data (1-5 GB), satu th*read* kerja, dan ukuran blok 2 kB. Pengujian terdiri dari lima tahap, dimulai dengan 1 GB data, diikuti peningkatan ukuran data hingga 5 GB, sementara parameter lainnya tetap konstan. Setiap tahap mencatat hasil iops untuk operasi baca dan tulis pada kedua jenis penyimpanan. Tujuan pengujian adalah untuk membandingkan kinerja iops ceph *cluster* dan raid saat ukuran data bertambah, menunjukkan perbedaan performa dalam mengelola berbagai ukuran data dengan parameter yang sama.

Tabel 3. 5 Skenario IOPS dengan Jumlah kerja (2) dan Ukuran Blok (2)

IOPS							
No	**Jumlah *Size* (Gb)**	**Jumlah Kerja**	**Ukuran Blok (k)**	**Ceph *Cluster***		**Raid**	
				Read	***Write***	***Read***	***Write***
1	1	2	2				
2	2	2	2				
3	3	2	2				

4	4	2	2				
5	5	2	2				

Tabel 3.5 memaparkan pengujian iops pada ceph *cluster* dan raid dengan variasi ukuran data (1-5 GB), jumlah kerja 2, dan ukuran blok 2 kB. Pengujian terdiri dari lima tahap, dimulai dari 1 GB dan meningkat hingga 5 GB, dengan parameter lainnya tetap. Hasil iops untuk operasi baca dan tulis pada kedua penyimpanan dicatat di setiap tahap. Pengujian ini bertujuan membandingkan performa iops ceph *cluster* dan raid seiring peningkatan ukuran data, menampilkan perbedaan kinerja dalam menangani berbagai ukuran data dengan parameter yang konsisten.

3.4.1.2 Skenario Pengujian *Bandwidth* Ceph *Cluster* dan Raid

Tabel Skenario penelitian *bandwidth* merupakan tabel perbandingan untuk ceph *cluster* dan raid yang akan di gunakan serta menjadi rujukan untuk proses pengujian *bandwidth* menggunakan aplikasi FIO.

Tabel 3. 6 Skenario *Bandwidth* dengan Jumlah kerja (1) dan Ukuran Blok (1)

***Bandwidth* (KiB/s)**							
No	**Jumlah *Size* (Gb)**	**Jumlah Kerja**	**Ukuran Blok (k)**	**Ceph *Cluster***		**Raid**	
				Read	***Write***	***Read***	***Write***
1	1	1	1				
2	2	1	1				
3	3	1	1				
4	4	1	1				
5	5	1	1				

Tabel 3.6 menggambarkan pengujian *bandwidth* (KiB/s) untuk ceph *cluster* dan raid dengan variasi ukuran data (1-5 GB), satu th*read* kerja, dan ukuran blok 1 kB. Pengujian dilakukan dalam lima tahap, mulai dari 1 GB dan meningkat hingga 5 GB, dengan parameter lainnya tetap konstan. Hasil *bandwidth* untuk operasi baca dan tulis pada kedua jenis penyimpanan dicatat di setiap tahap. Tujuan pengujian ini adalah membandingkan performa *bandwidth* ceph *cluster*

dan raid seiring peningkatan ukuran data, menunjukkan variasi kinerja dalam menangani berbagai ukuran data dengan parameter yang sama.

Tabel 3. 7 Skenario *Bandwidth* dengan Jumlah kerja (1) dan Ukuran Blok (2)

***Bandwidth* (KiB/s)**							
No	**Jumlah *Size* (Gb)**	**Jumlah Kerja**	**Ukuran Blok (k)**	**Ceph *Cluster***		**Raid**	
				Read	***Write***	***Read***	***Write***
1	1	1	2				
2	2	1	2				
3	3	1	2				
4	4	1	2				
5	5	1	2				

Tabel 3.7 merupakan skenario pengujian *bandwidth* (KiB/s) pada ceph *cluster* dan raid dengan variasi ukuran data (1-5 GB), jumlah kerja 1, dan ukuran blok 2 kB. Pengujian terdiri dari lima tahap, dimulai dengan ukuran data 1 GB dan meningkat hingga 5 GB, sementara parameter lainnya tetap konstan. Setiap tahap mencatat hasil *bandwidth* untuk operasi baca dan tulis pada kedua jenis penyimpanan. Tujuan pengujian ini adalah untuk membandingkan performa *bandwidth* ceph *cluster* dan raid saat ukuran data meningkat, menunjukkan perbedaan kinerja dalam menangani berbagai ukuran data dengan parameter yang sama.

Tabel 3. 8 Skenario *Bandwidth* dengan Jumlah kerja (2) dan Ukuran Blok (2)

***Bandwidth* (KiB/s)**							
No	**Jumlah *Size* (Gb)**	**Jumlah Kerja**	**Ukuran Blok (k)**	**Ceph *Cluster***		**Raid**	
				Read	***Write***	***Read***	***Write***
1	1	2	2				
2	2	2	2				
3	3	2	2				
4	4	2	2				
5	5	2	2				

Tabel 3.8 merupakan skenario pengujian *bandwidth* (KiB/s) pada ceph *cluster* dan raid dengan variasi ukuran data (1-5 GB), jumlah kerja 2, dan ukuran blok 2

kB. Pengujian terdiri dari lima tahap, dimulai dengan ukuran data 1 GB dan meningkat hingga 5 GB, sementara parameter lainnya tetap konstan. Setiap tahap mencatat hasil *bandwidth* untuk operasi baca dan tulis pada kedua jenis penyimpanan. Tujuan pengujian ini adalah untuk membandingkan performa *bandwidth* ceph *cluster* dan raid saat ukuran data meningkat, menunjukkan perbedaan kinerja dalam menangani berbagai ukuran data dengan parameter yang sama. Hasil dari pengujian ini diharapkan dapat memberikan wawasan yang jelas mengenai efisiensi dan kemampuan kedua sistem penyimpanan dalam mengelola beban kerja yang semakin besar, sehingga dapat membantu dalam pengambilan keputusan terkait pilihan sistem penyimpanan yang optimal sesuai dengan kebutuhan.

3.4.1.3 Skenario Pengujian Latensi Ceph *Cluster* dan Raid

Tabel Skenario penelitian di bawah ini merupakan tabel perbandingan latensi untuk ceph *cluster* dan raid yang akan di gunakan serta menjadi rujukan untuk proses pengujian latensi menggunakan aplikasi FIO.

Tabel 3. 9 Skenario Latensi dengan Jumlah kerja (1) dan Ukuran Blok (1)

Latensi (ms)							
No	**Jumlah *Size* (Gb)**	**Jumlah Kerja**	**Ukuran Blok (k)**	**Ceph *Cluster***		**Raid**	
				Read	***Write***	***Read***	***Write***
1	1	1	1				
2	2	1	1				
3	3	1	1				
4	4	1	1				
5	5	1	1				

Tabel 3.9 menampilkan hasil pengujian latensi (ms) pada ceph *cluster* dan raid dengan variasi ukuran data (1-5 GB), jumlah kerja 1, dan ukuran blok 1 kB. Pengujian terdiri dari lima tahap, dimulai dengan ukuran data 1 GB dan meningkat hingga 5 GB, sementara parameter lainnya tetap konstan. Setiap tahap mencatat hasil latensi untuk operasi baca dan tulis pada kedua jenis penyimpanan. Tujuan pengujian ini adalah untuk membandingkan performa latensi ceph *cluster*

dan raid saat ukuran data meningkat, menunjukkan perbedaan kinerja dalam menangani berbagai ukuran data dengan parameter yang sama.

Tabel 3. 10 Skenario Latensi dengan Jumlah kerja (1) dan Ukuran Blok (2)

Laatensi (ms)							
No	**Jumlah *Size* (Gb)**	**Jumlah Kerja**	**Ukuran Blok (k)**	**Ceph *Cluster***		**Raid**	
				Read	***Write***	***Read***	***Write***
1	1	1	2				
2	2	1	2				
3	3	1	2				
4	4	1	2				
5	5	1	2				

Tabel 3.10 menampilkan hasil pengujian latensi (ms) pada ceph *cluster* dan raid dengan variasi ukuran data (1-5 GB), jumlah kerja 1, dan ukuran blok 2 kB. Pengujian terdiri dari lima tahap, dimulai dengan ukuran data 1 GB dan meningkat hingga 5 GB, sementara parameter lainnya tetap konstan. Setiap tahap mencatat hasil latensi untuk operasi baca dan tulis pada kedua jenis penyimpanan. Tujuan pengujian ini adalah untuk membandingkan performa latensi ceph *cluster* dan raid saat ukuran data meningkat, menunjukkan perbedaan kinerja dalam menangani berbagai ukuran data dengan parameter yang sama.

Tabel 3. 11 Skenario Latensi dengan Jumlah kerja (2) dan Ukuran Blok (2)

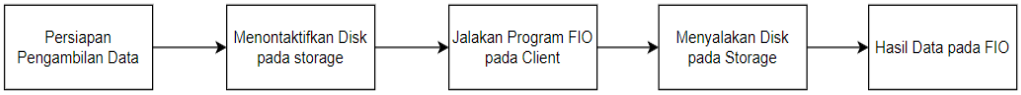
Laatensi (ms)							
No	**Jumlah *Size* (Gb)**	**Jumlah Kerja**	**Ukuran Blok (k)**	**Ceph *Cluster***		**Raid**	
				Read	***Write***	***Read***	***Write***
1	1	2	2				
2	2	2	2				
3	3	2	2				
4	4	2	2				
5	5	2	2				

Tabel 3.11 menampilkan hasil pengujian latensi (ms) pada ceph *cluster* dan raid dengan variasi ukuran data (1-5 GB), jumlah kerja 2, dan ukuran blok 2 kB. Pengujian terdiri dari lima tahap, dimulai dengan ukuran data 1 GB dan

meningkat hingga 5 GB, sementara parameter lainnya tetap konstan. Setiap tahap mencatat hasil latensi untuk operasi baca dan tulis pada kedua jenis penyimpanan. Tujuan pengujian ini adalah untuk membandingkan performa latensi ceph *cluster* dan raid saat ukuran data meningkat, menunjukkan perbedaan kinerja dalam menangani berbagai ukuran data dengan parameter yang sama.

3.4.2 *Disaster recovery*

Pengujian disaster *recovery* diterapkan pada kedua sistem penyimpanan, yaitu ceph *cluster* dan raid, dengan porsi pengujian yang sama untuk memastikan konsistensi hasil. Langkah-langkah pengujian dapat dilihat pada diagram blok di bawah ini agar lebih memudahkan pemahaman tentang proses pengujian disaster *recovery*. Diagram blok tersebut akan menjelaskan setiap tahapan pengujian secara detail, mulai dari simulasi kegagalan, langkah-langkah pemulihan, hingga evaluasi hasil pemulihan. Proses ini mencakup pengaturan ulang parameter, pemantauan kinerja sistem selama pemulihan, dan analisis data untuk menilai efektivitas strategi disaster *recovery* yang digunakan oleh masing-masing sistem penyimpanan.



Gambar 3. 2 Diagram Blok Disaster *Recovery*.

Pada gambar 3.2 memberikan penjelasan gambaran terinci mengenai langkah-langkah dalam proses pengujian *disaster recovery*. Tahapan awal melibatkan persiapan, dimana semua sistem diperiksa untuk memastikan bahwa sistem beroperasi dengan sempurna. Jika terdapat kesalahan atau kegagalan pada tahap ini, pengujian tidak dapat dilanjutkan hingga semua masalah diperbaiki. Setelah persiapan selesai, langkah selanjutnya adalah meninjau proses pengujian. Proses ini melibatkan simulasi menonaktifkan beberapa *disk* pada kedua sistem penyimpanan, baik itu ceph maupun raid. Tahapan selanjutnya adalah dengan menjalankan program fio pada *client* masing – masing sistem, setalah program fio di jalankan tahapan selanjutnya yaitu dengan menyalakan kembali *disk* pada sistem.

Pada tahap ini sistem penyimpanan secara otomatis memulai proses pemulihan untuk mengatasi kehilangan data atau kegagalan *disk*, proses pemulihan ini di pantau dan di catat kecepatan *transfer* dari *recovery*-nya oleh *tool* fio yang di akhir akan mengeluarkan nilai dari kecepatan *transfer* di masing – masing sistem. Setelah semua tahapan selesai data akan di kumpulkan dan data tersebut akan di proses serta di analisis untuk hasil yang di dapatkan. Keseluruhan, diagram blok tersebut membantu memvisualisasikan dan menjelaskan secara sistematis setiap tahapan yang dilibatkan dalam pengujian *disaster recovery* untuk memastikan kesiapan dan keandalan sistem dalam menghadapi situasi darurat.

3.4.2.1 Skenario Pengujian *Disaster Recovery* Ceph *Cluster*

Tabel skenario di bawah ini merupakan parameter pengujian ceph *cluster* yang akan di gunakan untuk proses pengujian pada *disaster recovery*, Berikut tabel skenario untuk *disaster recovery*.

Tabel 3. 12 Skenario Pengujian Disaster *Recovery* Ceph *Cluster*

No	OSD	Kondisi Sistem	Parameter			
			Read		*Write*	
			Kecepatan *Recovery* (IOPS)	*Bandwidth* (KiB/s)	Kecepatan *Recovery* (IOPS)	*Bandwidth* (KiB/s)
1	0	*Health*				
2	1					
3	2					
4	3					
5	4					
6	5					
7	0 dan 3					
8	1 dan 4					
9	2 dan 5					

Pada tabel 3.12, terdapat beberapa aspek yang dijelaskan untuk memahami proses pengujian disaster *recovery*. osd (*Object Storage Daemon*) merupakan sebutan untuk *disk* yang ada pada ceph *cluster*. Pada tabel 3.12, terdapat dua

variasi pengujian yang dilakukan, yaitu dengan *single disk* dan *double disk*. Pengujian *single disk* merupakan penonaktifan satu *disk*, seperti yang ada pada tabel 3.12 dari nomor 1 sampai 6. Sementara itu, pengujian *double disk* merupakan penonaktifan dua *disk* dalam satu kali percobaan, seperti pada nomor 7 sampai 9. Kondisi sistem perlu diketahui untuk memastikan bahwa semua *disk* atau osd yang ada pada sistem berjalan dengan baik setelah melakukan pengujian. Parameter yang diambil pada pengujian ini, yang dapat dilihat pada tabel 3.12, mencakup dua kondisi: ketika *disk* melakukan *input* (*write*) dan *output* (*read*). Parameter yang sama digunakan untuk kedua kondisi tersebut, yaitu kecepatan *transfer* (iops) dan *bandwidth*. Proses pengujian ini bertujuan untuk mengevaluasi ketahanan dan performa sistem ceph *cluster* dalam menghadapi kegagalan *disk*. Dengan melakukan penonaktifan *disk* secara bertahap, baik *single* maupun *double*, dapat diukur seberapa cepat sistem dapat pulih dan mempertahankan performa baca-tulis data.

##mematikan *disk* (osd)
systemctl stop ceph-osd@0.*service*
systemctl stop ceph-osd@1.*service*
systemctl stop ceph-osd@2.*service*
systemctl stop ceph-osd@3.*service*
systemctl stop ceph-osd@4.*service*
systemctl stop ceph-osd@5.*service*
systemctl stop ceph-osd@0.*service*
systemctl stop ceph-osd@3.*service*
systemctl stop ceph-osd@1.*service*
systemctl stop ceph-osd@4.*service*
systemctl stop ceph-osd@2.*service*
systemctl stop ceph-osd@5.*service*

Menyalakan *Disk*
systemctl start ceph-osd@0.*service*
systemctl start ceph-osd@1.*service*
systemctl start ceph-osd@2.*service*
systemctl start ceph-osd@3.*service*
systemctl start ceph-osd@4.*service*
systemctl start ceph-osd@5.*service*
systemctl start ceph-osd@0.*service*
systemctl start ceph-osd@3.*service*
systemctl start ceph-osd@1.*service*
systemctl start ceph-osd@4.*service*
systemctl start ceph-osd@2.*service*
systemctl start ceph-osd@5.*service*

Untuk melakukan pengujian seperti yang sudah dijelaskan diagram blok pada gambar 3.2, *disk* (OSD) pada sistem ceph *cluster* harus dinonaktifkan agar dapat mensimulasikan kondisi ketika terjadi bencana pada *disk*. Untuk menonaktifkan serta menyalakan kembali *disk* (OSD), diperlukan perintah tertentu agar simulasi dapat berjalan sesuai dengan skenario yang diinginkan. Perintah ini bisa dilihat

pada bagian perintah di atas. Proses penonaktifan osd memungkinkan kita untuk mengukur dampak langsung pada kinerja sistem, termasuk kecepatan *transfer* data (iops) dan *bandwidth*. Selain itu, pengujian ini membantu kita untuk memahami bagaimana sistem dapat pulih dari kegagalan *disk* dan mengembalikan fungsionalitas normal setelah osd dinyalakan kembali. Melalui pengujian ini, kita juga dapat mengevaluasi efisiensi dari prosedur disaster *recovery* yang telah diterapkan, memastikan bahwa sistem tetap dapat berjalan dengan baik dan data tetap aman selama dan setelah terjadi gangguan pada *disk*.

3.4.2.2 Skenario Pengujian *Disaster Recovery* Sistem Raid

Tabel skenario 3.13 merupakan parameter pengujian raid yang akan di gunakan untuk proses pengujian pada *disaster recovery*, proses pengujian akan di lakukan dengan metode yang sudah di jelaskan dengan mengambil dua parameter yaitu iops dan *bandwidth* beserta dengan *read*/*write* pada sistem raid, Berikut tabel skenario untuk *disaster recovery*.

Tabel 3. 13 Skenario Pengujian *Disaster Recovery* Raid

No	*Array Device* (*Disk*)	Kondisi Sistem	Parameter			
			Read		*Write*	
			Kecepatan *Recovery* (IOPS)	*Bandwidth* (KiB/s)	Kecepatan *Recovery* (IOPS)	*Bandwidth* (KiB/s)
1	0	*Health*				
2	1					
3	2					
4	3					
5	4					
6	0 dan 1					
7	1 dan 2					
8	2 dan 3					
9	3 dan 4					

Pada tabel 3.13 merupakan tabel untuk menjadi acuan pengambilan data pada sistem raid yang hampir sama dengan tabel pengujian ceph *cluster*, hanya saja pada tabel pengujian sistem raid istilah yang digunakan untuk *disk* adalah "*array*

disk," yang mengatur *disk* berdasarkan logika *array*. Penerapan pengujian pada sistem raid terdapat dua variasi yang dilakukan, yaitu dengan *single disk* dan *double disk*, kombinasi ini merupakan bagian dari simulasi pengujian untuk mendapatkan nilai yang diinginkan. Pengujian ini bertujuan untuk mengevaluasi performa dan kehandalan sistem raid dalam menghadapi berbagai skenario kegagalan *disk*, sebagaimana yang dilakukan pada ceph *cluster*, namun dengan fokus pada struktur dan karakteristik khusus yang dimiliki oleh sistem raid.

#menonaktifkan array (*disk*)
mdadm --manage /dev/md0 --set-faulty /dev/vdb
mdadm --manage /dev/md0 --set-faulty /dev/vdc
mdadm --manage /dev/md0 --set-faulty /dev/vdd
mdadm --manage /dev/md0 –-set-faulty /dev/vde
mdadm --manage /dev/md0 --set-faulty /dev/vdf
mdadm --manage /dev/md0 --set-faulty /dev/vdb &&
mdadm --manage /dev/md0 --set-faulty /dev/vdc
mdadm --manage /dev/md0 --set-faulty /dev/vdc &&
mdadm --manage /dev/md0 --set-faulty /dev/vdd
mdadm --manage /dev/md0 --set-faulty /dev/vdd &&
mdadm --manage /dev/md0 --set-faulty /dev/vde
mdadm --manage /dev/md0 --set-faulty /dev/vde &&
mdadm --manage /dev/md0 --set-faulty /dev/vdf

#menyalakan array (*disk*)
mdadm --manage /dev/md0 --re-add /dev/vdb
mdadm --manage /dev/md0 --re-add /dev/vdc
mdadm --manage /dev/md0 --re-add /dev/vdd
mdadm --manage /dev/md0 --re-add /dev/vde
mdadm --manage /dev/md0 --re-add /dev/vdf
mdadm --manage /dev/md0 --re-add /dev/vdb &&
mdadm --manage /dev/md0 --re-add /dev/vdc
mdadm --manage /dev/md0 --re-add /dev/vdc &&
mdadm --manage /dev/md0 --re-add /dev/vdd
mdadm --manage /dev/md0 --re-add /dev/vdd &&
mdadm --manage /dev/md0 --re-add /dev/vde
mdadm --manage /dev/md0 --re-add /dev/vde &&
mdadm --manage /dev/md0 --re-add /dev/vdf

Metode atau skenario yang di lakukan pada pengujian *disaster recovery* pada sistem raid ini dengan menonaktifkan *disk* dengan mengikuti dari tabel 3.13, menonaktifkan *disk* ini di perlukan agar memberikan simulasi *disaster recovery* pada sistem yang nantinya akan di lihat dari seberapa cepat sistem ini melakukan *disaster recovery* dalam satuan iops. Berikut perintah yang digunakan untuk menonaktifkan dan menyalakan *disk* yang ada di atas.