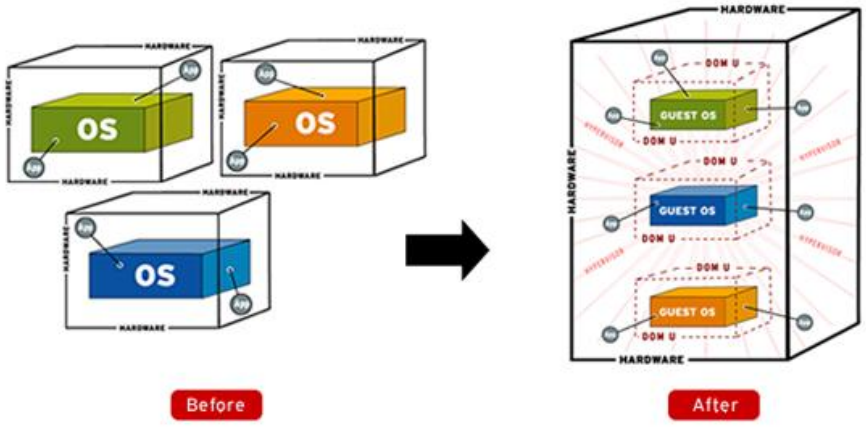
BAB 2
TINJAUAN PUSTAKA

2.1 Landasan Teori

Landasan teori pada penelitian ini berfungsi sebagai kerangka acuan ilmiah untuk membedah permasalahan yang di angkat. Bagian ini memapar sejumlah teori fundamental, konsep dasar yang relevan dengan topik. Penjelasan mendalam mengenai poin-poin tersebut dijabarkan ke dalam sub-bab berikut :

2.1.1 Virtualisasi

Virtualisasi merupakan teknologi yang memungkinkan satu perangkat keras fisik digunakan untuk menjalankan beberapa sistem operasi atau layanan secara bersamaan dalam bentuk *Virtual Machine* (VM). Dengan pendekatan ini, sumber daya seperti CPU, memori, dan penyimpanan dapat dialokasikan secara lebih efisien sehingga pemanfaatan server menjadi optimal. Selain itu, virtualisasi juga menyederhanakan proses administrasi sistem serta mempercepat proses *deployment* aplikasi dan layanan (Soni et al., 2019). Virtualisasi juga dapat dianalogikan sebagai sumber daya perangkat keras dibagikan diantara *client-client* yang mengira mereka sedang berjalan di perangkat keras asli (Hartanto, 2012). Dalam penelitian ini, virtualisasi menjadi pilar utama dalam membangun *cluster* diatas tiga server fisik.



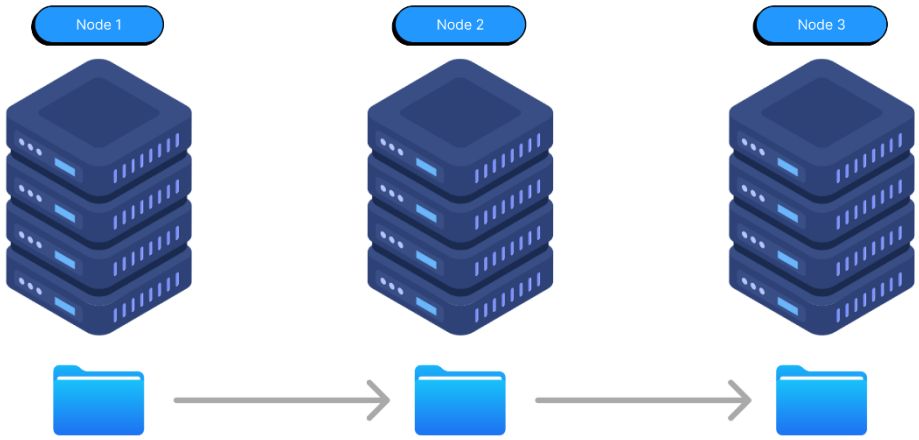
Gambar 2.1 Virtualisasi

2.1.2 Cluster Server

Cluster server didefinisikan sebagai sekumpulan *node* server independen yang saling terhubung dan bekerja secara kolektif sebagai satu sistem yang sama. Arsitektur ini dirancang untuk mencapai beberapa tujuan krusial, seperti peningkatan kerja, skalabilitas dan ketersediaan layanan yang tinggi. Tujuan fundamental dari pembentukan cluster adalah menjamin kontinuitas operasional. Jika salah satu node mengalami kegagalan, node lain harus mampu mengambil alih beban kerja melalui mekanisme *failover* secara otomatis. Berdasarkan studi (Najmi & Zulfariansyah, 2025), penerapan *failover clustering* pada Proxmox VE telah terbukti efektif dalam meningkatkan uptime layanan hingga diatas 99% dengan waktu respons *failover* rata-rata sekitar 1,85 detik. Secara umum, jenis jenis cluster server dapat dibedakan berdasarkan fokus tujuannya :

1. High Availability Cluster, fokus menjaga uptime layanan.
2. Load Balancing Cluster, membagi beban kerja antar server.
3. High Performance Cluster, memaksimalkan daya komputasi paralel.

Penelitian ini berfokus pada HA Cluster yang digunakan untuk menjaga kontinuitas layanan walau terjadi kegagalan fisik.



Gambar 2.2 *Cluster* Server

2.1.3 High Availability

High Availability (HA) adalah konsep fundamental yang bertujuan menjamin layanan tetap beroperasi tanpa interupsi yang signifikan, meskipun terjadi kegagalan

pada salah satu komponen sistem. Pencapaian HA memerlukan kombinasi dari replikasi data, redudansi perangkat dan mekanisme *failover* otomatis.

Gambar 2.3 High Availability

Proxmox VE telah menyertakan manajer HA bawaan (*built-in*) yang berfungsi yang mendeteksi *node* yang gagal dan secara otomatis memindahkan *Virtual Machine*(VM) ke node lain yang sehat dalam cluster. Jika *server down* atau sumber daya berhenti, maka *cluster* HA akan melihat dan memastikan bahwa sumber daya di-restart tempat lain dalam *cluster*, sehingga dapat digunakan kembali setelah gangguan (Vugt, 2014). HA Manager ini beroperasi dengan melakukan pemeriksaan kesehatan (*health check*) secara berkala, dikoordinasikan melalui Proxmox Cluster Engine (pve-cluster) dan Corosync. Berdasarkan studi, implementasi HA pada Proxmox VE yang menggunakan Ceph Storage terbukti mampu mempertahankan ketersediaan layanan VM di atas 97% (Febriansyah & Prapanca, 2024). Untuk menghitung durasi downtime pada saat terjadi kegagalan node, digunakan rumus sebagai berikut :

$$\text{Downtime} = T_{\text{Layanan Online Kembali}} - T_{\text{Layanan Mati}}$$

Failover time merupakan waktu yang dibutuhkan layanan atau VM untuk aktif kembali pada node lain setelah terjadi kegagalan di node asalnya, yang dihitung dengan rumus berikut :

$$\text{Failover Time} = T_{\text{VM Aktif di node baru}} - T_{\text{Node gagal}}$$

Sebagai dasar penilaian terhadap baik atau buruknya nilai downtime dan failover yang dihasilkan pada penelitian, digunakan acuan standar industri mengenai tingkat ketersediaan (*availability*). (O'Hara et al., 2024) dan dokumentasi (PhoenixNAP, 2024) menjelaskan bahwa sistem High Availability umumnya diklasifikasikan berdasarkan tingkat *availability* yang dikenal sebagai *three nines, four nines, dan five nines.* Setiap tingkat memiliki toleransi downtime tertentu yang dapat dijadikan benchmark untuk menilai hasil pengujian pada penelitian ini. Tabel berikut menunjukan standar *availability* dan batas maksimal downtime yang umum digunakan dalam industri :

Tabel 2.1 Standar *High Availability*

Tingkat Availability	Istilah	Downtime per Tahun
99,9%	Three Nines	± 8,76 jam
99,99%	Four Nines	± 52 menit
99,999%	Five Nines	± 5 menit

Sumber : (PhoenixNAP, 2024), (O'Hara et al., 2024)

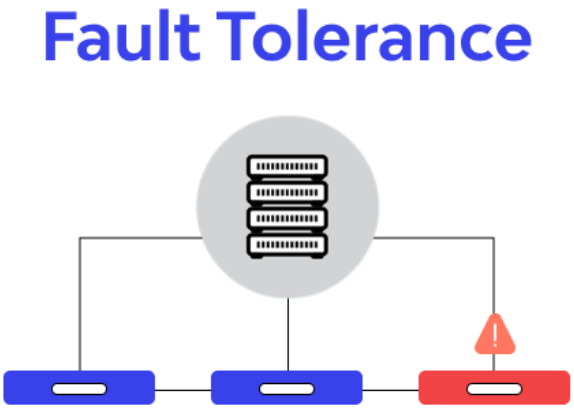
Standar tersebut tidak digunakan sebagai target pencapaian dalam penelitian, namun digunakan sebagai dasar pembanding dalam menilai nilai downtime dan failover yang diperoleh dalam pengujian.

2.1.4 Fault Tolerance

Fault Tolerance adalah kemampuan sistem untuk terus beroperasi meskipun terjadi kerusakan pada sebagian komponennya. Tujuan bukan hanya menjaga ketersediaan, tetapi juga untuk memastikan data tetap konsisten. Menurut (Yahya, 2023), sistem cluster Proxmox VE dengan mekanisme replikasi data mampu meminimalisir kehilangan data saat node mengalami kegagalan. Perbedaan mendasar antara FT dan HA adalah :

a. HA fokus pada pemulihan cepat setelah kegagalan.
b. FT fokus pada mencegah kegagalan mempengaruhi sistem melalui replikasi langsung (real-time replication)

Dalam sistem Proxmox VE, kombinasi HA dan FT memungkinkan sistem tetap berjalan stabil bahkan saat dilakukan simulasi kerusakan perangkat keras (hardware failure).



Gambar 2.4 *Fault Tolerance*

Menurut dokumentasi resmi (Red Hat., 2024) ,proses recovery tidak memiliki standar waktu baku karena durasinya sangat dipengaruhi oleh faktor-faktor teknis seperti jumlah OSD, kapasitas disk, tipe media penyimpanan, bandwith jaringan, tingkat replikasi dan beban cluster saat terjadi kegagalan. Oleh karena itu, dalam penelitian ini waktu recovery data diukur menggunakan log Ceph pada saat pengujian dilakukan. Perhitungan recovery time dilakukan dengan rumus :

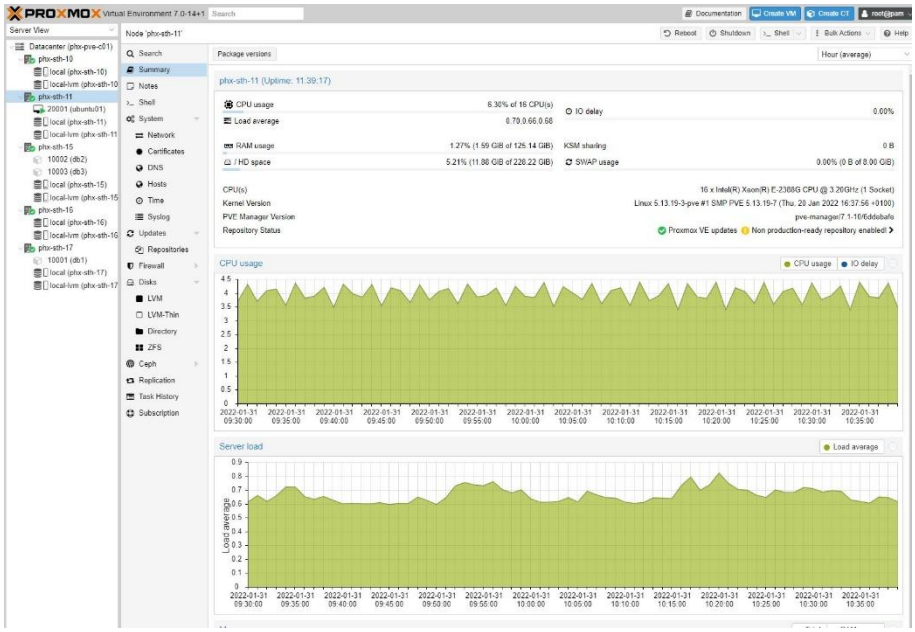
$$\text{Recovery Time} = T_{\text{Ceph Healthy}} - T_{\text{Failure}}$$

Rumus ini digunakan untuk menentukan durasi yang dibutuhkan Ceph agar kembali berada pada status HEALTH OK setelah terjadi kegagalan node atau disk.

2.1.5 Proxmox Virtual Environment (VE)

Proxmox VE merupakan platform virtualisasi open-source berbasis *Debian Linux* yang menggabungkan dua teknologi utama, yaitu *KVM (Kernel-based Virtual Machine)* dan *LXC* (*Linux Container*), serta menyediakan antarmuka web yang memudahkan manajemen cluster dan konfigurasi High Availability (HA). Dokumentasi resmi Proxmox menjelaskan bahwa sistem ini didukung oleh beberapa komponen inti, termasuk Proxmox Cluster File System (pmxcfs) yang menyinkronkan konfigurasi antar node, *Corosync* sebagai komunikasi cluster untuk menjaga

konsistensi dan deteksi kegagalan, serta HA Manager yang mengelola proses *failover* otomatis bila node mengalami kegagalan (GmbH, 2025).



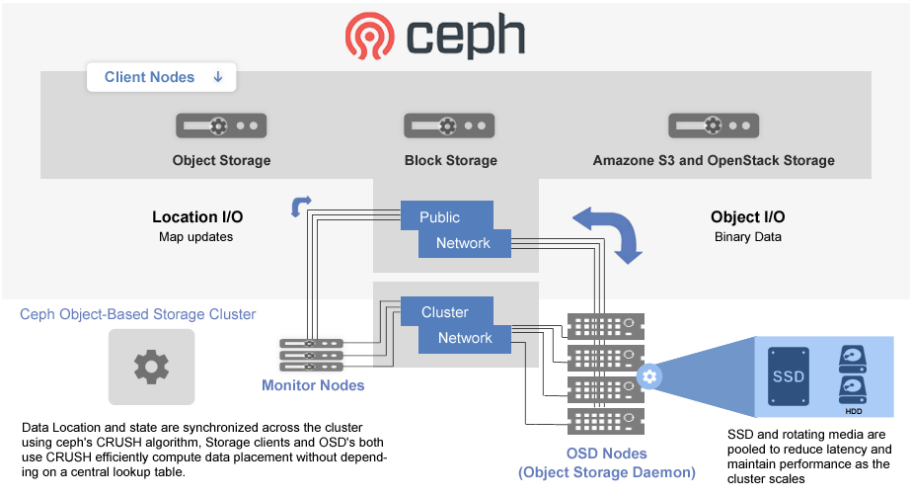
Gambar 2.5 Dashboard Proxmox VE

2.1.6 Ceph Storage

Ceph adalah sistem penyimpanan terdistribusi yang bersifat *open-source* dan dirancang untuk menyediakan *scalability*, *fault tolerance* dan *high performance*. Dalam implementasi *High Availability*, *ceph* berperan penting sebagai media penyimpanan bersama (*shared storage*) antar node. Ceph terdiri dari beberapa komponen utama seperti :

a. OSD (*Object Storage Daemon*) : Menyimpan dan mereplikasi data antar node.
b. *Monitor* (MON) : Mengawasi kesehatan cluster.
c. *Manager* (MGR) : Menyediakan data statistik dan kontrol administratif.

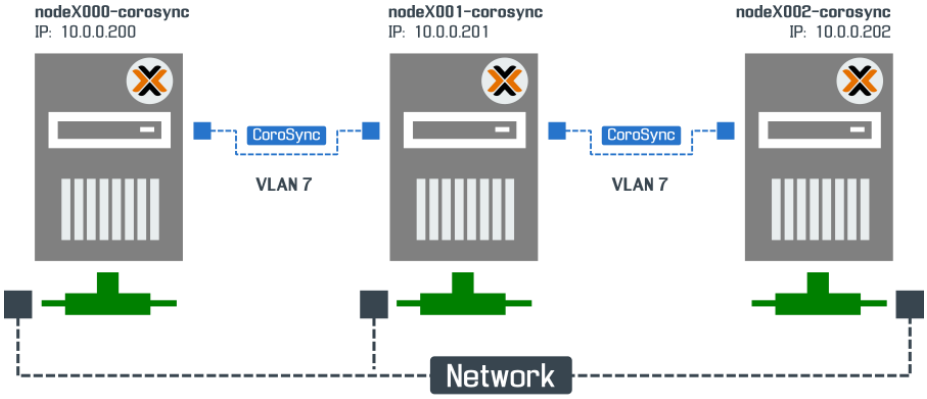
Keunggulan utama Ceph adalah kemampuan *self-healing* dan *self-managing*, sehingga cluster dapat tetap beroperasi meskipun terjadi kegagalan pada salah satu node penyimpanan. Hal ini ditegaskan dalam dokumentasi resmi Ceph yang menjelaskan bahwa mekanisme replikasi serta proses pemulihan otomatis memungkinkan Ceph menjaga ketersediaan dan integritas data pada kondisi kegagalan perangkat keras (T. C. Community, 2025).



Gambar 2.6 Ceph *Storage*

2.1.7 Corosync dan Pacemaker

Corosync merupakan layanan komunikasi yang digunakan dalam sistem cluster untuk memastikan sinkronisasi status antar node. *Pacemaker* berperan sebagai *cluster resource manager* yang mengatur prioritas layanan dan proses *failover*. Pada Proxmox VE, Corosync digunakan secara bawaan (*default service*), sementara *Pacemaker* dapat diintegrasikan secara opsional. Kombinasi keduanya dapat meningkatkan stabilitas dan efisiensi dalam pengelolaan HA (GmbH, 2025).

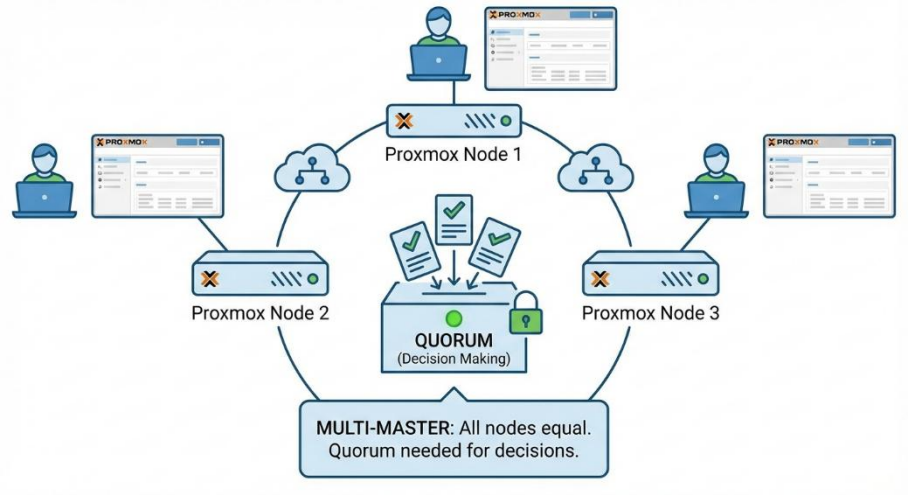


Gambar 2.7 *Corosync*

2.1.8 Quorum dan Voting System

Quorum merupakan mekanisme penentuan keputusan dalam *cluster*, yaitu jumlah minimum node aktif yang dibutuhkan agar sistem dapat berfungsi dengan benar. Jika jumlah node aktif kurang dari quorum, maka cluster dianggap tidak stabil. Dalam sistem tiga node, quorum tercapai ketika minimal dua node aktif dan saling

berkomunikasi sehingga mayoritas (lebih dari 50%) suara tercapai. *Corosync* menangani proses voting dan memastikan hanya node yang valid yang boleh menjalankan layanan dan menghindari terjadinya kondisi *split brain* (SUSE, 2024).



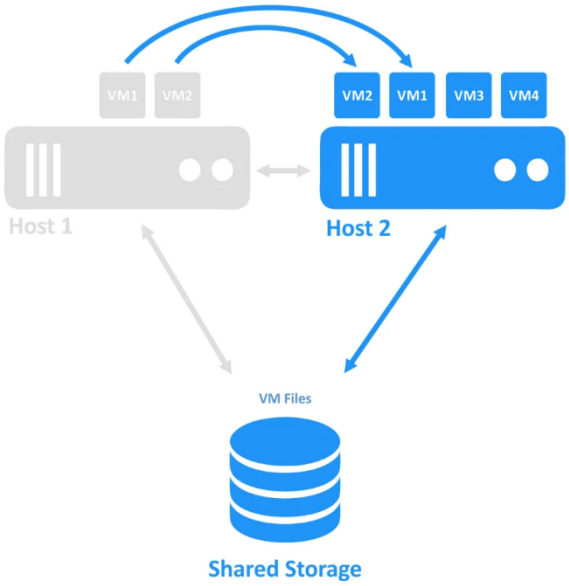
Gambar 2.8 *Quorum*

2.1.9 Mekanisme Failover dan Recovery

Mekanisme *failover* merupakan proses otomatis ketika layanan berpindah dari node yang gagal ke node lain yang masih aktif. Proses ini terdiri dari :

a. Deteksi kegagalan (*failure detection*) oleh *Corosync.*
b. Pengalihan layanan (*service migration)* ke node lain.
c. Pemulihan (*recovery*) setelah node yang gagal kembali aktif.

Waktu yang dibutuhkan dalam proses ini disebut *failover time*, salah satu parameter utama yang akan diuji dalam penelitian ini. Dokumentasi resmi Proxmox menjelaskan bahwa HA manager bekerja bersama *Corosync* dan PMXCFS untuk memastikan deteksi kegagalan yang cepat serta pemindahan layanan secara otomatis guna menjaga kontinuitas operasional (GmbH, 2025).



Gambar 2.9 *Failover*

2.1.10 Jenis-jenis Kegagalan Sistem & Parameter Evaluasi Sistem

Dalam pengujian ketahanan High Availability (HA) pada sistem *cluster* Proxmox VE, simulasi *hardware failure* dilakukan untuk menilai respon dan perilaku sistem. Jenis kegagalan yang umum disimulasikan dan relevan dalam penelitian ini meliputi *Power Failure* (terputusnya daya listrik yang menyebabkan server mati mendadak) dan *Disk Failure* (kerusakan media penyimpanan yang mengganggu akses data). Untuk menggambarkan respons sistem secara kuantitatif, beberapa parameter evaluasi kunci diukur selama simulasi kegagalan :

1. Rata-rata waktu *downtime* : durasi waktu layanan tidak aktif.
2. Rata-rata *Failover* Time : waktu yang dibutuhkan layanan untuk pulih pada node.
3. *Recovery* data pada *Ceph Storage* : waktu yang dibutuhkan Ceph untuk kembali ke status HEALTH OK setelah kegagalan *node* atau disk.

Pengukuran nilai-nilai ini bertujuan untuk mengevaluasi ketahanan dan efisiensi *failover* cluster Proxmox VE Community Edition saat menghadapi kegagalan fisik.

2.1.11 Fencing (STONITH)

Fencing adalah mekanisme mengeluarkan node yang bermasalah dari *cluster* agar tidak mengganggu node lain. Pada Proxmox, fencing biasanya dilakukan melalui *power management* dengan tujuan mencegah *split brain* dan menjaga agar hanya node sehat yang menjalankan *cluster*. Dokumentasi resmi Proxmox juga menegaskan bahwa mekanisme *fencing* merupakan bagian penting dari arsitektur *High Availability* untuk menjamin integritas dan keandalan operasi cluster (GmbH, 2025).

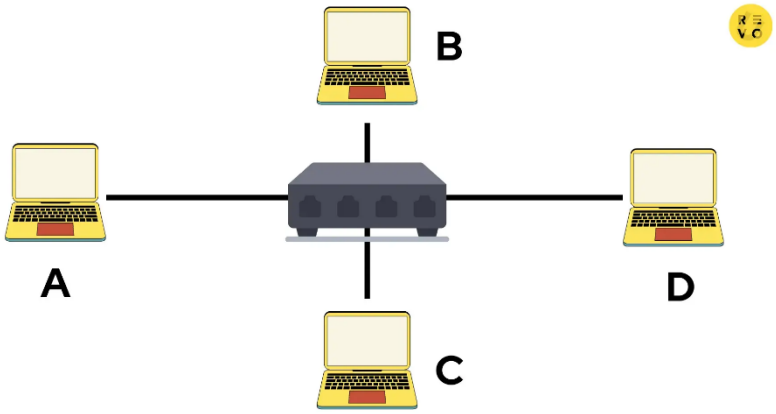
Gambar 2.10 *Fencing*

2.1.12 Heartbeat

Heartbeat adalah mekanisme pengiriman sinyal antar-node secara periodik untuk memastikan setiap node masih hidup/*online.* Jika *heartbeat* berhenti, maka *cluster* menganggap node tersebut mengalami kegagalan dan memicu proses *failover* atau *fencing*. Pada Proxmox VE, mekanisme *heartbeat* dikelola melalui *Corosync* yang menggunakan protokol komunikasi *multicast* atau *unicast* untuk menjaga sinkronisasi status antar-node dalam *cluster*. Dokumentasi *Corosync* dan Proxmox menjelaskan bahwa *heartbeat* adalah komponen utama dalam proses deteksi kegagalan yang cepat dan akurat dalam sistem High Availability (GmbH, 2025).

2.1.13 Switch External

Switch External merupakan perangkat jaringan yang digunakan sebagai jalur utama untuk menghubungkan node-node Proxmox VE dengan jaringan publik, termasuk akses internet, IP Publik, serta koneksi dari perangkat *client*. Switch ini menangani lalu lintas manajemen seperti akses antarmuka web Proxmox, komunikasi SSH, serta konektivitas layanan virtual machine. Menurut (Oppenheimer, 2010), pemisahan lalu lintas manajemen dari jalur penyimpanan sangat penting untuk menjaga stabilitas jaringan dan mengurangi resiko *bottleneck.*



Gambar 2.11 Switch

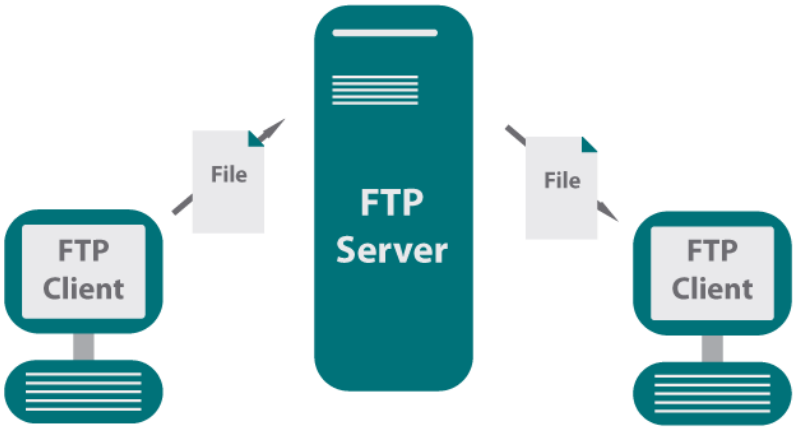
2.1.14 Mail Server

Mail Server adalah sistem yang bertugas mengelola proses pengiriman, penerimaan dan penyimpanan pesan email dalam suatu jaringan komputer. Proses pengiriman email dilakukan menggunakan protokol SMTP (*Simple Mail Transfer Protocol*), sementara proses penerimaan dan pembacaan email dilakukan melalui protokol POP3 (*Post Office Protocol*) atau IMAP (*Internet Message Access Protocol*) yang memungkinkan sinkronisasi antara klien dan server (Tanenbaum & Wetherall, 2011). Dalam penelitian terkait ketersediaan layanan, mail server digunakan sebagai indikator untuk mengukur stabilitas sistem dan durasi *downtime* saat terjadi kegagalan pada node cluster.

Gambar 2.12 Mail Server

2.1.15 FTP Server

FTP (*File Transfer Protocol*) merupakan protokol standar yang digunakan untuk melakukan transfer file antara klien dan server melalui jaringan TCP/IP. FTP menyediakan dua saluran komunikasi, yaitu *control connection* untuk perintah dan otentikasi, serta *data connection* untuk proses transfer data aktual (Stallings, 2017). Dalam konteks evaluasi performa sistem, FTP sering dijadikan layanan pengujian untuk mengamati keberlangsungan akses file, waktu respons dan durasi *downtime* ketika terjadi kegagalan node dalam *cluster*.



Gambar 2.13 FTP Server

2.2 Penelitian Terdahulu

Pada topik penelitian pertama dilakukan oleh Alfido Adha Febriansyah dan Aditya Prapanca (2024) dari *Journal of Informatics and Computer Science* (*JINACS*) yang berjudul "Simulasi IMplementasi High Availability Server Menggunakan Ceph pada Proxmox". Penelitian ini menggunakan platform Proxmox VE dengan penerapan Ceph Storage untuk mendukung sistem High Availability. Pada penelitian ini dilakukan simulasi untuk menguji keandalan server ketika salah satu node mengalami gangguan. Hasil penelitian menunjukan kestabilan server dalam menangani kegagalan sehingga waktu *downtime* dapat diminimalisir. Namun penelitian ini hanya melakukan sebatas simulasi *software* bukan pada kegagalan fisik.

Penelitian selanjutnya dilakukan oleh Khayreen Najmi dan Muhammad Zulfariansyah (2025) dalam Jurnal Teknologi Informasi (JURTI) dengan judul "Perancangan dan penerapan High Availability Web Server dengan Failover Clustering di Lingkungan pendidikan Tinggi". Pada penelitian ini digunakan metode *Network Development Life Cycle* (NDLC) untuk membangun sistem *failover clustering* berbasis Proxmox VE. Sistem ini diuji di lingkungan pendidikan untuk memastikan layanan web tetap berjalan meskipun salah satu server mengalami gangguan. Hasil penelitian menunjukan bahwa penerapan *failover cluster* berhasil menurunkan waktu *downtime* dan meningkatkan ketersediaan sistem. Namun pada penelitian ini tidak menggunakan Ceph Storage pada pengujiannya.

Penelitian selanjutnya dilakukan oleh Didik Darmadi dan Rafles Susandi (2023) dalam Journal of Analytical research, Statistic and Computation (GARUDA) dengan judul "Pengujian efektivitas Sistem Cluster dengan Penerapan High Availability pada Server Virtual". Penelitian ini membahas penerapan cluster berbasis Proxmox VE untuk mengukur efektivitas dan kestabilan server virtual dalam menangani kegagalan. Hasil pengujian menunjukan sistem mampu bertahan dari kegagalan pada salah satu node dengan waktu pemulihan kurang dari 1 menit. Namun pada penelitian tidak melakukan pengujian dengan Ceph Storage pada pengujiannya.

Penelitian berikutnya dilakukan oleh Gede Arna Jude Saskara et al. (2025) dari *Jurnal Algoritme* dengan judul *"Implementasi Proxmox untuk High Availability* dan *Load Balancing Sistem SIAK Undiksha."* Penelitian ini menggunakan metode *Network Development Life Cycle* (NDLC) dalam membangun *cluster* Proxmox dengan dukungan Ceph Storage. Hasil penelitian menunjukan bahwa sistem memiliki tingkat ketersediaan sebesar 99,89% dan dapat menjaga kestabilan layanan akademik. Namun, pada kondisi beban tinggi lebih dari 20.000 permintaan, performa sistem menurun sehingga diperlukan optimasi *load balancing*. Pada penelitian ini tidak melakukan pengujian hardware failure dan tidak mengukur recovery data pada penelitiannya.

Penelitian berikutnya dilakukan oleh Dadan Irwan, Heru Sukoco, dan Sri Wahjuni (2020) dalam *Jurnal Komputer Indonesia (JKI)* yang berjudul *"Service High* Availability *pada* Native *Server dan Virtual Server Menggunakan Proxmox VE."* Penelitian ini membandingkan penerapan HA pada server fisik (*native*) dan server virtual berbasis Proxmox VE. Hasil penelitian menunjukkan bahwa server virtual memiliki keunggulan pada efisiensi sumber daya dan kemudahan pengelolaan, meskipun waktu *failover* sedikit lebih lama pada beban tinggi. Pada penelitian hanya terfokus pada penerapan HA pada server native dan virtual.

Tabel 2.2 Penelitian Terdahulu

Uraian	**(Saskara et al., 2025)**	**(Najmi & Zulfariansyah, 2025)**	**(Febriansyah & Prapanca, 2024)**	**(Darmadi & Susandi, 2022)**	**(Irwan et al., 2020)**
Teknik Virtualisasi	Cluster Computing & Load Balancing	Failover Clustering & High Availability	High Availability & Automation	Virtualization (VM & Container) & Clustering	Virtualization (Hypervisor Type 1) & Failover
Platform / Virtual Machine	Proxmox VE + Ceph	Proxmox VE (Failover Cluster)	Proxmox VE (Cluster HA)	Proxmox VE (Cluster HA)	Proxmox VE (Native vs Virtual)
Tools / Software	Proxmox VE, Ceph (Storage), Apache JMeter (Pengujian performa/load test),	Proxmox VE, Ceph (Storage Terdistribusi), Stopwatch (untuk mengukur *response time* manual).	Proxmox VE, Ceph, Python Scripts (untuk otomasi deteksi *downtime*,	Proxmox VE, Fping (untuk ukur downtime), NAS Server (NFS) untuk	Proxmox VE, Heartbeat (Failover), DRBD (Replikasi Data), Httperf

Uraian	(Saskara et al., 2025)	(Najmi & Zulfariansyah, 2025)	(Febriansyah & Prapanca, 2024)	(Darmadi & Susandi, 2022)	(Irwan et al., 2020)
	VMWare (untuk simulasi awal).		pemulihan, dan monitoring ping)	penyimpanan bersama.	(Generator beban/request), Apache2, MySQL
Tujuan Penelitian	Menganalisis kinerja Proxmox Ceph Cluster dalam menghadapi peningkatan trafik (*Load Balancing*) dan memastikan layanan tetap optimal untuk menggantikan sistem server monolitik di UPA TIK Undiksha	Merancang sistem *failover clustering* dan *high availability* untuk meningkatkan stabilitas, ketersediaan, dan performa layanan web server di lingkungan pendidikan tinggi (UNU Kaltim).	Mengurangi *downtime* seminimal mungkin dan meminimalkan intervensi manusia dalam proses pemulihan sistem melalui skrip otomatisasi pada lingkungan Proxmox dan Ceph.	Mengetahui efektivitas pemanfaatan server melalui virtualisasi sistem cluster dan mengukur *downtime* pada fitur *High Availability* dibandingkan server yang berjalan sendiri-sendiri.	Menganalisis kemampuan *service high availability* dengan sistem *failover* dan *failback* pada dua arsitektur berbeda (*native* vs *virtual*) dan mengukur kinerjanya.

Berdasarkan kajian terhadap beberapa penelitian terdahulu pada Tabel 2.2 (*Politeknik Caltex Riau*, 2026), diketahui bahwa penelitian-penelitian tersebut telah membahas topik yang relevan dengan penelitian ini, namun memiliki keterbatasan dari sisi skenario pengujian kegagalan fisik secara langsung pada sistem High Availability (HA), dengan berfokus pada implementasi *High Availability* dan *Fault Tolerance* pada cluster Proxmox VE dengan mengkonfigurasi tiga server fisik menjadi satu cluster. Penyimpanan terdistribusi dibangun menggunakan Ceph untuk replikasi data dan menjaga konsistensi storage antar-node. Penelitian ini melakukan pengujian ketahanan sistem melalui dua skenario kegagalan fisik secara langsung, yaitu pemutusan daya listrik (*power failure*) dan pelepasan hard disk (*disk failure*) pada setiap node. Parameter yang diukur mencakup rata-rata waktu *downtime* layanan, rata-rata waktu *failover* dan rata-rata waktu *recovery* data untuk menilai kemampuan cluster dalam mempertahankan ketersediaan layanan serta

konsistensi data saat terjadi kegagalan perangkat keras nyata (*real hardware failure*).