**DEVELOPMENT OF PRIVATE STORAGE ARCHITECTURE BASED ON OBJECT STORAGE WITH THE IMPLEMENTATION OF ERASURE CODING METHOD TO IMPROVE DATA RELIABILITY AND EFFICIENCY**

**Ainun Maarif, I Putu Hariyadi2**

Universitas Bumigora

|  |  |
| --- | --- |
| **Artikel Info**  ***Kata-kata kunci***  *kk* | ***ABSTRAK***  kk. |
| **Article Info**  ***Keywords***  *kk* | ***ABSTRAC***  kk |

# Introduction.

The development of information technology, which is taking place at an incredible pace, has become a dominant force involving every aspect of our lives. (Nazwa Salsabila Lubis1, Muhammad Irwan Padli Nasution2). In recent decades, technology has developed very rapidly, changing the way humans live and interact. The emergence of a new era known as the Fourth Industrial Revolution or Industrial Revolution 4.0 is one of the most significant impacts (Muhamad Waqqor Bukhori1\*, Muhammad Giyaatsusshidqi2, NabilaAgustina3, Yumna Sabilal Huda).

Especially related to data management and storage. In today's digital era, the amount of data generated by companies, organizations, and individuals is increasing exponentially. This data is not only large, but also important, so that storage reliability and efficiency are primary needs. As a result, various more modern and efficient data storage solutions have been developed, one of which is object storage.

Object storage is a scalable and robust data storage architecture. Object storage has common characteristics such as object-based storage, horizontal scalability, and support for unstructured data, this technology is also widely used in cloud storage, such as Amazon Simple Storage Service (S3), Google Cloud Storage, and OpenStack Swift. (Prakhar Pandey1, Arpit2, Umashankar Sharma3).

However, issues of reliability and storage efficiency arise along with the increase in data volume. Distributed storage often faces the risk of hardware failure or data loss due to system failure.

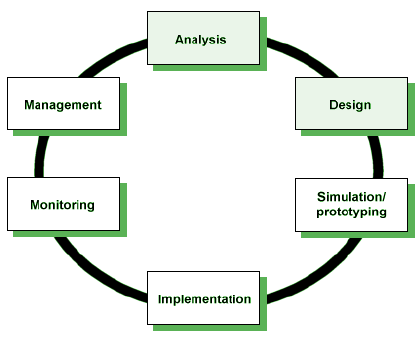
Reliability and efficiency of data storage are two very important components. If data is lost due to hardware failure or other problems, it can greatly affect the company's operations. On the other hand, given the high cost of managing data storage, storage capacity efficiency must also be considered. Therefore, a mechanism is needed to ensure that the stored data remains available and protected even if a device fails. Erasure coding is one solution that has proven successful.

Errasure code is one strategy applied in distributed data storage systems, and is applied to network coding, the code used in this technique is based on an algebraic structure in the form of a vector. (Agus Maman Abadi\*, Musthofa, and Emut). Erasure Coding (EC) is a data protection method in which data is broken into fragments, expanded and encoded with redundant data pieces and stored in a set of various locations or storage media. (Meutia Zamieyus).

Despite much progress, the development of object-based private storage architectures using erasure coding methods remains a challenge. The gap between how erasure coding can be used in the real world to improve storage reliability and efficiency theoretically is a major challenge. This gap is often exacerbated by the lack of interdisciplinary collaboration, especially between practitioners in the field and storage architecture designers. Nevertheless, closer collaboration is essential for successfully utilizing object storage in various situations. Integrating this storage technology into existing infrastructures also requires changes in methodology and techniques.

This is especially necessary to fully exploit the benefits of erasure coding, especially in terms of optimizing data efficiency and resilience at scale. This paper reviews the relevant literature and reviews the development of object-based private storage architectures using erasure coding methods, with a special emphasis on improving data reliability and efficiency. The goal of this research is to fill the gap related to the lack of a holistic framework that combines various storage technologies into a more solid model.

# METODOLOGY



Gambar 1. 1 Metode NDLC

Metode penelitian yang digunakan adalah *Network Development Life Cycle* (NDLC). *NDLC* terdiri dari 6 (enam) tahapan yaitu *Analysis, Design, Simulation Prototyping, Implementation, Monitoring* dan *Management*, seperti terlihat pada gambar dibawah. Peneliti hanya menggunakan 3 (tiga) dari 6 (enam) tahapan yang terdapat pada *NDLC* yaitu *Analysis, Design*, dan *Simulation* Prototypi*ng* (Suharto & Irfan, 2019). Tahapan-tahapan yang terdapat pada NDLC. Dimana tahapannya yang terlihat pada gambar 2.1 :

Dari 6 tahapan yang ada, penulis hanya menggunakan 3 tahapan yaitu *analysis*, *design* dan simulation *prototyping*. Sehingga dari 3 tahapan tersebut penulis dapat mengimplementasikan dalam pembuatan sistem.

1. Analysis

Tahap awal ini dilakukan analisa kebutuhan, analisa permasalah yang mucul, analisa keinginan pengguna, analisa desain *arsitectur private storage*.

1. Design

Dari data-data yang didapatkan sebelumnya, tahap design ini akan membuat gambar desain arsitectur penyimpanan privat berbasis object storage, merancang scema penerapan metode erasure coding.

1. Simulation and prototyping

mempresentasikan tinjauan ulang yang menyeluruh terhadap spesifikasi dan desain.Hal ini dimaksudkan untuk melihat kinerja awal dari rancangan pengembangan arsitekture penyimpanan privat berbasis object storage dengan penerapan erasure coding.

* 1. **Analisis Data**

Adapun langkah-langkah yang akan dilakukan penulis pada tahapan ini yaitu melakukan pengumpulan dan penulis melakukan analisis data dari permasalahan yang muncul dan melakukan analisis kebutuhan pada pengembangan arsitektur penyimpanan privat berbasis object storage dengan penerapan metode erasure coding untuk meningkatkan keandalan dan efisiensi data.

* Pengumpulan Data

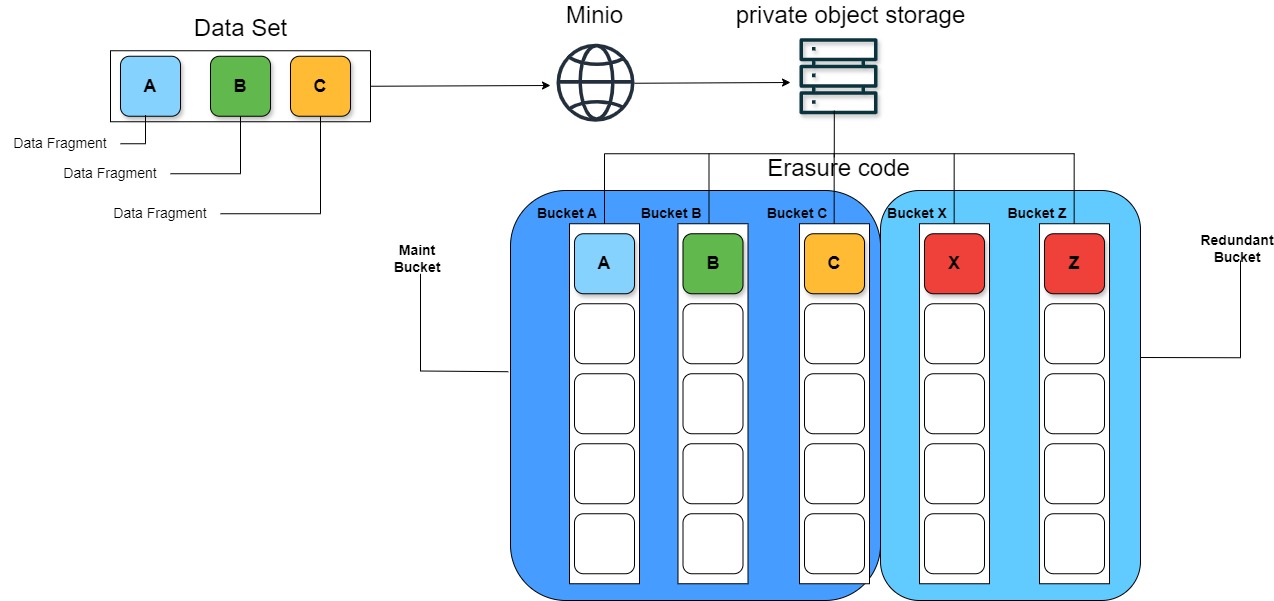
**Tabel 2.1 Penelitian Sebelumnya**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| No | Penulis | Tujuan | Pendekatan/Metode Penelitian | Hasil Penelitian | Aspek Kritis Penelitian |
| 1. | Jeong Joon Kim (2021) | mengusulkan metode penyimpanan dan pemulihan berkas yang efisien | Metode yang digunakan adalah teknik pemulihan parallel HDFS berbasis erasure code | HDFS, salah satu sistem berkas terdistribusi berbasis EC, dan mengusulkan metode penyimpanan dan pemulihan berkas yang efisien. Teknik buffering dan pooling digunakan dalam penyimpanan berkas, yang meningkatkan kinerja sekitar 2,5 kali lipat dibandingkan dengan HDFS yang ada. Untuk pemulihan berkas, kinerja ditingkatkan sekitar 2 kali lipat dengan menggunakan distribusi beban I/O disk, penempatan blok acak, dan teknik daur ulang matriks. | Penelitian ini memang menggunakan erasure code tapi hanya sebatas memulihkan file system |
| 2 | Aatish ChiniahNomor telepon 1\*dan Avinash Mungur2(2021) | untuk menyajikan solusi yang tersedia sedemikian rupa sehingga pengembang aplikasi cloud dapat dengan mudah mengidentifikasi solusi yang sesuai yang sesuai dengan persyaratan aplikasi mereka sendiri. Jadi pertama-tama, kami memberikan pengantar singkat tentang berbagai jenis sistem penyimpanan terdistribusi. | Menggunakan teorema CAP (Consistency, Availability and Partition Tolerance) | Penerapan erasure code sangat cocok untuk dikombinasikan dengan layanan cloud storage | Penelitian ini memang menggunakan erasure code namun penerapanya masih pada cloud storage. |
| 3. | Anindita Sarkar Mondal ID 1\*, Madhupa Sanyal2† , Ari Kusumastuti3† , Hrishav Bakul Barua3† , Kartick Chandra Mondal( 2024) | untuk menunjukkan arsitektur berbagai sistem penyimpanan cloud komersial yang akan menciptakan kesadaran bagi konsumen layanan penyimpanan juga |  |  |  |
| 4. | Prakhar Pandey1 , Arpit2 , Umashankar Sharma3(2024) | menyelidiki secara komprehensif prinsip desain, arsitektur, dan karakteristik kinerja sistem penyimpanan objek terdistribusi, selain itu bertujuan untuk meliputi skalabilitas, toleransi kesalahan, dan efisiensi keseluruhan penyimpanan object storage. | Menggunakan metode campuran untuk menyelidiki secara komprehensif prinsip desain dan karakteristik kinerja sistem penyimpanan objek terdistribusi. Desain penelitian menggabungkan tinjauan pustaka sistematis, analisis arsitektur, dan evaluasi kinerja empiris untuk mencapai pemahaman holistik tentang pokok bahasan. | Hasilnya adalah object storage mampu menangani peningkatan jumlah node secara efisien, menunjukan throughput yang kuat dalam operasi baca. | Penelitian ini memerlukan optimasi lebih lanjut untuk menjaga throughput dan latensi yang ideal di bawah beban kerja tinggi. |
| 5. | AndrewHanushevsky1Michal KamilSimon2, DanKamiYang (2024) | a) membuktikan fungsionalitas system  b) memahami overhead dari mode akses yang dipilih,  c) mengukur kinerja mode akses yang dipilih,  d) melakukan operasi administratif. | Metode yang digunakan adalah metode pemulihan data, dan administrasi | Pengintegrasian kemampuan erasure code dalam cluster xrood storage berfungsi, dan mendukung semua autentikasi/otorisasi WLCC, akses data, dan persyaratan transfer data | Penelitian ini memang menerapkan erasure code akan tetapi penelitian ini hanya menerapkan erasure code hanya dalam mengetahui throughput dari upload file dari user kedalam xrootd storage, dan terjadi penurunan kinerja dalam pembacaan ketika mengakses data melalui proxy xrootd |
| 6. | Vaneet anggarawai 2020 | Tujuan dari monograf ini adalah untuk memberikan tinjauan tentang kemajuan terkini (baik teoritis maupun praktis) pada sistem yang menggunakan kode penghapusan untuk penyimpanan terdistribusi. | Metode menggunakan erasure code. | Menerapkan erasure code dalam system storage untu memperbaiki ukuran file dan memvariasikan tingkat permintaan file dengan tingkat permintaan rata-rata | Penelitian ini dilakukan dengan menggunakan system storage. |
| 7. |  |  |  |  |  |
| 8. |  |  |  |  |  |
| 9. |  |  |  |  |  |
| 10. |  |  |  |  |  |

Pada tahapan pengumpulan data ini penulis melakukan studi *literature* yaitu dengan mempelajari beberapa artikel ilmiah yang membahas tentang *Virtual Extensible Lan , Vxlan Over IPSec, Ansible Playbook,* dan *Vlan*. Setelah membaca jurnal ilmiah dan skripsi sebelumnya diperoleh informasi tentang beberapa jurnal ilmiah dan skripsi sebelumnya sebagai berikut.

* 1. **Design**

Pada tahapan ini penulis mendesain sebuah rancangan berupa rancangan uji coba atau rancangan skenario uji coba dari sistem yang dibuat, dan membuat rancangan sistem Rancangan jaringan uji coba yang digunakan terlihat seperti gambar dibawah ini.



**Gambar 2.1 Rancangan Topologi Uji Coba**

Rancangan system diatas terlihat 1 (satu) server private object storage dimana dari rancangan tersebut terdapat minio yang difungsikan sebagai object storage dan didalam object storage tersebut diterapkan erasure code sebagai teknik yang digunakan untuk menyimpan dan melindungi data ke beberapa bucket didalam minio object storage. Dari gambar diatas terlihat 3 (tiga) bucket utama yaitu bucket A, bucket B, dan Bucket C, yang akan difungsikan sebagai tempat menyimpan pecahan data dari data set yang dikirim kedalam minio object storage, dan terlihat 2 (dua) bucket redundant yaitu bucket X, dan Bucket Z yang akan difungsikan sebagai restoring data ketika terjadi kerusakan pada salah satu bucket utama.

* Rancangan system

Pada perancangan sistem ini dilakukanya pengalamatan *IP* yang akan digunakan pada penelitian ini dan menggunakan 7 (tujuh) alamat *IP* kelas *C*, yaitu *192.168.100.X/24* untuk alamat *IP* setiap perangkat, *192.168.10.X/24, 192.168.20.X/24, 192.168.30.X/24* untuk setiap *VXLAN* *Over IPSec* di setiap *server PVE*, dan *192.168.11.X/24, 192.168.22.X/24,* dan *192.168.33.X/24* untuk alamat *IP VLAN* pada setiap *server PVE* A lokasi pengalamatan *IP* pada masing masing perangkat, *server* dan masing masing router dan client di setiap *vlan*.

**Rancangan Pengalamatan IP:**

**Tabel 2.2 Pengalamatan IP**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **NO** | **Nama Perangkat** | **Interface** | **Alamat IP** | **Subnetmask** |
| 1 | Windows 11 | Vmnet 3 | 192.168.100.252/24 | 255.255.255.0 |
| 2 | Mikrotik chr | Ether 1 Ether 2 | 192.168.100.254/24 | DHCP 255.255.255.0 |
| 3 | Server ansible | Ens 33 | 192.168.100.253 | 255.255.255.0 |
| 4 | Server PVE1 | Vmbr0 | 192.168.100.1/24 | 255.255.255.0 |
| 5 | Vxlan 10 | Vmbr2 | 192.168.10.254/24 | 255.255.255.0 |
| 6 | Vlan 11 | Vmbr2 | 192.168.11.254/24 | 255.255.255.0 |
| 7 | Container 11 | Vlan11 | 192.168.11.11/24 | 255.255.255.0 |
| 8 | Container 12 | Vlan11 | 192.168.11.12/24 | 255.255.255.0 |
| 9 | Vxlan 20 | Vmbr3 | 192.168.20.254/24 | 255.255.255.0 |
| 10 | Vlan 22 | Vmbr3 | 192.168.22.254/24 | 255.255.255.0 |
| 11 | Container 21 | Vlan 22 | 192.168.22.21/24 | 255.255.255.0 |
| 12 | Container 22 | Vlan 22 | 192.168.22.22/24 | 255.255.255.0 |
| 13 | Vxlan 30 | Vmbr4 | 192.168.30.254/24 | 255.255.255.0 |
| 14 | Vlan 33 | Vmbr4 | 192.168.33.254/24 | 255.255.255.0 |
| 15 | Container 31 | Vlan 33 | 192.168.33.31/24 | 255.255.255.0 |
| 16 | Container 32 | Vlan 33 | 192.168.33.32/24 | 255.255.255.0 |
| 17 | Server PVE2 | Vmbr0 | 192.168.100.2/24 | 255.255.255.0 |
| 18 | Vxlan 10 | Vmbr2 | 192.168.10.253/24 | 255.255.255.0 |
| 19 | Vlan 11 | Vmbr2 | 192.168.11.253/24 | 255.255.255.0 |
| 20 | Container 11 | Vlan11 | 192.168.11.13/24 | 255.255.255.0 |
| 21 | Container 12 | Vlan11 | 192.168.11.14/24 | 255.255.255.0 |
| 22 | Vxlan 20 | Vmbr3 | 192.168.20.253/24 | 255.255.255.0 |
| 23 | Vlan 22 | Vmbr3 | 192.168.22.253/24 | 255.255.255.0 |
| 24 | Container 21 | Vlan 22 | 192.168.22.23/24 | 255.255.255.0 |
| 25 | Container 22 | Vlan 22 | 192.168.22.24/24 | 255.255.255.0 |
| 26 | Vxlan 30 | Vmbr4 | 192.168.30.253/24 | 255.255.255.0 |
| 27 | Vlan 33 | Vmbr4 | 192.168.33.253/24 | 255.255.255.0 |
| 28 | Container 31 | Vlan 33 | 192.168.33.33/24 | 255.255.255.0 |
| 29 | Container 32 | Vlan 33 | 192.168.33.34/24 | 255.255.255.0 |

* Kebutuhan Perangkat Keras dan Perangkat Lunak

Pada tahapn analisa kebutuhan perangkat keras dan perangkat lunak dari penulisan skripsi dan pembuatan sistem adalah sebagai berikut:

1. Dengan mengunakan 1 (satu) unit laptop/*notebook* yang menggunakan sistem operasi *windows* dan didalam sistem operasi *windows* tersebut diinstalkan sebuah *vmware workstation.* Kemudian didalam *vmware workstation* tersebut diinstallkan beberapa sistem operasi untuk membangun sebuah sistem virtualisasi seperti *proxmox, CentOs7* sebagai *server* *ansible* atau sebagai mesin virtualisasi untuk melakukan otomatisasi ke pada *server proxmox ve,* dan *mikrotik chr* sebagai *internet gateway* .

* 1 laptop/*notebook* dengan spesifikasi berikut:
* Processor : Intel Core I5
* Hardisk :1 TB
* Memory :12 GB
* Sistem Operasi : Windows 11, 64bit
* 1 server Ansible dengan spesifikasi berikut:
* Processor : 1 Core
* Hardisk : 20 GB
* Memory : 1 GB
* Sistem Operasi : CentOs 7
* 2 server proxmox 7 dengan spesifikasi berikut:
* Processor : 1 core
* Hardisk : 20 GB
* Memory : 1 GB
* Sistem Operasi : Mikrotik RouterOs versi 7
* 1 sistem Operasi mikrotik chr dengan spesifikasi berikut:
* Processor : 1 Core
* Memory : 254 MB

1. Kebutuhan perangkat lunak (*Software)*
2. *Vmware workstation* perangkat lunak atau *software*  *virtual machine* (*VM*) yang akan digunakan untuk menjalankan lebih dari 1 (satu) sistem operasi (*OS*) yang ada didalam perangkat keras (*hardware)* contohnya seperti personal komputer (*PC*) dan *laptop*/*notebook*.
3. *Proxmox ve* salah satu dari sekian banyak varian *linux* yang berbasis *debian* yang akan difungsikan sebagai *server virtualisasi*.
4. *Mikrotik cloud hosted router (CHR)* salah satu produk dari *mikrotik* yang dikhususkan untuk teknologi virtualisasi. *Chr* digunakan pada aplikasi virtualisasi diantara lain *virtualBox, vmware, kvm* dan lainya.
5. *Community enterprise operating system (centos7) distro linux* yang digunakan dalam skala perusahaan, dan bebas biaya atau gratis*. Centos* sendiri di *code* dari *source code red hat enterprise (RHEL)* yang dikembangkan dalam sebuah komunitas yang disebut *centos project*.

Pada tahapan ini membahas tentang konfigurasi yang ada pada perangkat jaringan dan pengkodean sistem otomatisasi yang akan dibuat.

1. Instalasi dan konfigurasi

Pada tahap instalasi dan konfigurasi pada masing masing perangkat pada rancangan sistem yang akan diuji coba sebagai berikut.

1. Installasi CentOs 7 sebagai server *ansible*

* Konfigurasi dan pengalamatan IP server *ansible*
* Installasi *epel-release*
* Intallasi *Ansible*
* Installasi *Nano*
* Installasi sshd

1. Installasi mikrotik chr sebagai internet gateway

Pada langkah ini melakukan pengalamatan IP pada *mikrotik* *chr* dan melakukan konfigurasi *mikrotik* *chr* sebagai *internet gateway* sehingga dapat tehubung ke Internet.

1. Installasi dan konfigurasi *server PVE1* dan *PVE2*

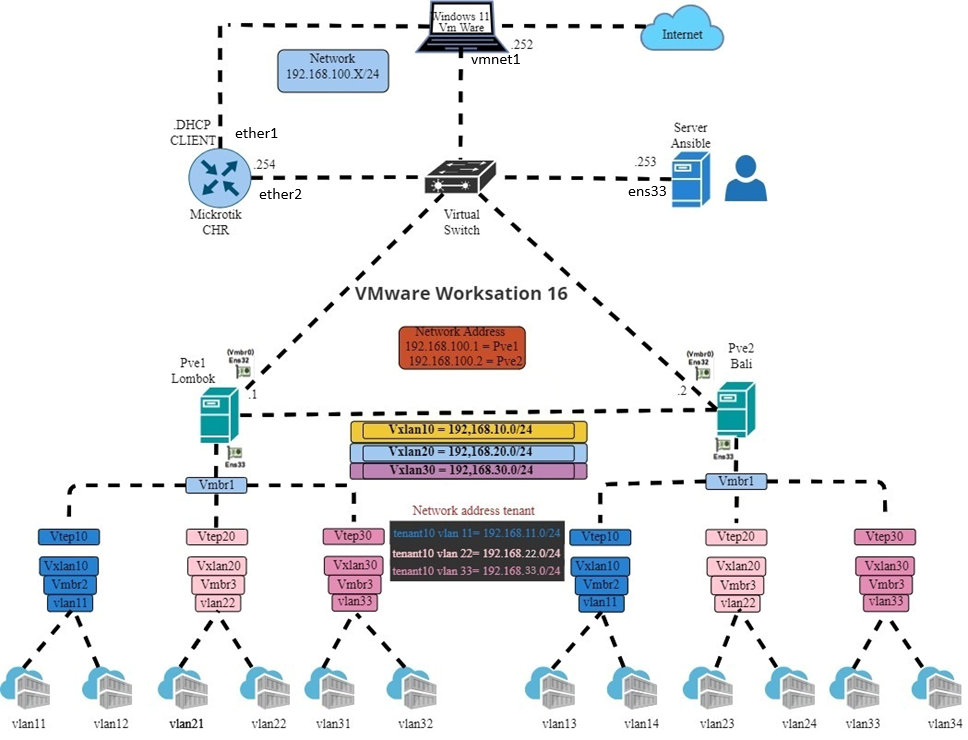
* Konfigurasi pengalamtan *IP* *Server* *PVE1* dan *PVE2*
* Installasi *OpenVswitch*
* Installasi *ifupdown2*
* Installasi *python*-*pip*
* Installasi *Proxmoxer*
* Installasi *IPerf*
* Installasi *VXLAN 10,20,30*
* Installasi *VXLAN Over IPSec 10,20,30*
* Installasi *VLAN 11,22,33*

1. Pada tahap ini dilakukan konfigurasi pembuatan *vxlan, vxlan Over IPSec* dan vlan di masing masing *server PVE 1* dan  *server PVE2* sehingga dapat saling terhubung*.*Pembuatan kode program

Adapun pembuatan kode dari program otomatisasi yang akan dibuat yaitu:

1. Membuat *file create-open-vswitch-pve1* dengan format file .yml
2. Membuat *file create-open-vswitch-pve2* dengan format file .yml
3. Membuat *file create-vxlan-pve1* dengan format file .yml
4. Membuat *file create-vxlan-pve2* dengan format file.yml
5. Membuat *file create-vxlan-over-ipsec-pve1* dengan format file.yml
6. Membuat *file create-vxlan-over-ipsec-pve2* dengan format file.yml
7. Membuat *file create-vlan-pve1* dengan format file .yml
8. Membuat *file create-vlan—pve2* dengan format file .yml
   1. **Simulasi *Prototyping***

Pada tahapan ini melakukan protoryping dari sistem yang sudah dirancang dan dari rancangan yang sudah dibuat akan dilakukan pengujian sistem, berikut topologi prototyping dari sistem terlihat pada gambar 3.4 berikut.



Gambar 3. 2 Rancangan Prototyping Sistem

Pada tahap ujicoba atau prototyping ini memuat semua tentang hasil installasi dan konfigurasi menggunakan beberapa ujicoba sebagai berikut:

Dalam scenario ujicoba ini dilakukan pengujian menjadi 2 (dua) bagian yaitu scenario ujicoba pengujian sebelum diterapkanya *VXLAN Over IPSec* dan sesudah diterapkanya *VXLAN Over IPSec* pada masing masing server *PVE* *1* dan *server PVE2* beirkut:

1. Otomatisasi Konfigurasi pembuatan *OpenVswitch PVE1* dan *PVE2*.
2. Otomatisasi konfigurasi pembuatan *VXLAN PVE1* dan *PVE2.*
3. Otomatisasi *VLAN PVE1* dan *PVE2*.
4. *Sniffing* paket sebelum diterapkanya *VXLAN Over IPSec PVE1* dan *PVE2*.
5. Otomatisasi Konfigurasi pembuatan *VXLAN Over IPSec PVE1* dan *PVE2*.
6. Installasi paket *VXLAN Over IPSec PVE1* dan *PVE2*.
7. *Sniffing* paket sesudah diterapkanya *VXLAN Over IPSec PVE1* dan *PVE2*.

# HASIL DAN PEMBAHASAN

## Penerapan konfigurasi Vxlan Over IPSec pada server pve bali dan Lombok secara automatisasi.

ff

# KESIMPULAN DAN SARAN

f

1. Desain dan implementasi arsitektur virtualisasi SDN *Controller*: Lakukan desain dan implementasi arsitektur yang menggabungkan virtualisasi SDN *Controller* dengan *platform* virtualisasi yang dipilih. Jelaskan secara rinci bagaimana virtualisasi tersebut dapat memberikan manfaat dalam manajemen jaringan, seperti isolasi yang lebih baik, penggunaan sumber daya yang efisien, dan fleksibilitas konfigurasi jaringan.
2. Evaluasi skalabilitas virtualisasi SDN *Controller*: Lakukan evaluasi terhadap skalabilitas dari virtualisasi SDN *Controller* yang diimplementasikan. Uji kinerja jaringan dengan jumlah VM SDN *Controller* yang berbeda-beda dan amati pengaruhnya terhadap kinerja jaringan secara keseluruhan. Evaluasi ini dapat memberikan wawasan tentang batasan dan kelebihan dari solusi virtualisasi yang digunakan.
3. Analisis keamanan pada virtualisasi SDN *Controller*: Fokus pada analisis keamanan pada virtualisasi SDN Controller. Tinjau berbagai ancaman keamanan yang mungkin timbul dalam lingkungan virtualisasi, serta desain dan implementasi mekanisme keamanan yang dapat melindungi SDN *Controller* dan jaringan yang dikelolanya. Lakukan pengujian dan evaluasi terhadap keamanan solusi yang diimplementasikan.
4. Pengembangan algoritma pengalokasian sumber daya otomatis: Lakukan penelitian tentang pengembangan algoritma yang dapat secara otomatis mengalokasikan sumber daya jaringan untuk VM SDN *Controller* berdasarkan permintaan dan kondisi jaringan. Evaluasi algoritma tersebut dengan menguji kinerja jaringan dalam situasi yang berbeda dan bandingkan dengan metode alokasi sumber daya tradisional.
5. Analisis dampak virtualisasi SDN Controller terhadap aplikasi jaringan: Fokus pada analisis dampak dari penggunaan virtualisasi SDN Controller terhadap aplikasi jaringan yang berjalan di atasnya. Tinjau performa aplikasi, latensi, dan faktor-faktor lain yang dapat memengaruhi kinerja aplikasi dalam lingkungan virtualisasi SDN Controller. Bandingkan dengan lingkungan jaringan tradisional.