



PENERAPAN DISASTER RECOVERY PLAN PADA DATABASE MYSQL MENGUNAKAN METODE MASTER TO MASTER REPLICATION BERBASIS DOCKER

Rahmat Suhatman^{1*}, Muhammad Arif Fadhly Rida², Ibnu Surya³ dan Raudatul Tasya⁴
Teknik Informatika (Politeknik Caltex Riau, Pekanbaru, 28265, Indonesia)^{1,2,3,4}
rahmat@pcr.ac.id¹, fadhly@pcr.ac.id², ibnu@pcr.ac.id³, raudatul19ti@mahasiswa.pcr.ac.id⁴

**Penulis Koresponden*

ABSTRAK

Data yang tersimpan pada *database* harus memiliki proteksi terhadap serangan dan gangguan. Serangan dan gangguan pernah dialami oleh Politeknik Caltex Riau (PCR) yang hanya memiliki satu *database*. *Database* yang mengalami serangan atau gangguan mengakibatkan sistem informasi tidak dapat diakses oleh pengguna. Saat ini PCR belum mempunyai rencana pemulihan sistem informasi sehingga pemulihan sistem informasi memerlukan waktu yang lama agar dapat diakses kembali oleh pengguna. Oleh karena itu, PCR memerlukan rencana pemulihan jika *database* mengalami serangan atau gangguan. Pada penelitian ini dilakukan replikasi *database* menggunakan *Disaster Recovery Plan (DRP)* untuk merespons keadaan darurat, menyediakan operasi cadangan selama gangguan terjadi, dan mengelola pemulihan dan menyelamatkan proses sesudahnya. Metode yang dipakai adalah *Master to Master MySQL Replication*. Saat *database* pada master 1 mengalami perubahan maka secara *realtime* akan disalin ke master 2. Berdasarkan hasil pengujian didapatkan proses replikasi untuk query *insert update* dan *delete* berhasil 100%. Begitu juga pada mekanisme replikasi master to master antar server berjalan secara otomatis. Sehingga penerapannya dapat memberikan cadangan yang termutakhirkan secara langsung bagi sistem *database*, saat mengalami serangan atau kerusakan.

Kata kunci: serangan, gangguan, database, Disaster Recovery Plan (DRP), Master to Master MySQL Replication

ABSTRACT

Data stored in the database must have protection against attacks and interference. Attacks and disruptions have been experienced by the Caltex Riau Polytechnic (PCR), which only has one database. A database that experiences an attack or disruption makes the information system inaccessible to users. Currently, PCR does not have an information system recovery plan so information system recovery will take a long time so that users can reaccess it. Therefore, PCR requires a recovery plan if the database experiences an attack or disruption. In this research, database replication was carried out using a Disaster Recovery Plan (DRP) to respond to emergencies, provide backup operations during disruptions, and manage recovery and rescue processes afterward. The method used is Master to Master MySQL Replication. When the database on Master 1 experiences changes, it will be copied to Master 2 in real time. Based on the test results, it was found that the replication process for insert update and delete queries was 100% successful. Likewise, the master-to-master replication mechanism between servers runs automatically. So its implementation can provide an updated backup directly for the database system when it experiences an attack or damage.

Keywords: attacks, disruptions, databases, Disaster Recovery Plan (DRP), Master to Master MySQL Replication

Histori Artikel:

Diserahkan: 18 Desember 2023

Diterima setelah Revisi: 14 Juni 2024

Diterbitkan: 14 Juni 2024

1. PENDAHULUAN

Teknologi adalah penerapan pengetahuan ilmiah untuk tujuan praktis dalam kehidupan manusia atau pada perubahan dan manipulasi lingkungan manusia. Seiring dengan perkembangan teknologi yang semakin pesat, saat ini banyak teknologi yang telah dikembangkan terutama di bidang *database*. *Database* atau basis data merupakan struktur penyimpanan data untuk menambah, mengakses dan memproses data yang disimpan dalam sebuah *database* komputer, diperlukan sistem manajemen *database* seperti MySQL Server.

Meningkatkan proses kerja agar lebih efektif dan efisien, diperlukan kinerja sistem *database* yang optimal. Namun, dalam penggunaan *database*, kita akan dihadapkan dengan resiko gangguan pada *database*. Seperti terjadinya kehilangan data dan juga *database* tidak dapat diakses (*downtime*). Saat server *database* gagal, semua layanan akan berhenti dan semua proses bisnis tidak akan berfungsi tepat. Kerusakan seperti itu dapat mengakibatkan kerugian yang sangat besar, baik material dan kerusakan moral bagi perusahaan.

Projects.co.id, sebuah website yang menyediakan layanan yang menghubungkan *freelancer* dan klien mengalami kegagalan perangkat keras pada tahun 2015. Situs ini tidak dapat diakses dan mereka tidak memiliki data cadangan terbaru. Menggunakan data cadangan dari beberapa minggu yang lalu, mereka hanya dapat mengembalikan 70-80% dari keseluruhan data yang mereka miliki. Untungnya, situs web ini aktif kembali setelah 4 hari mencoba untuk memulihkan data dari hard drive (Hardy, 2015).

Politeknik Caltex Riau (PCR) merupakan Perguruan Tinggi Vokasi Swasta terbaik di Indonesia, yang memiliki jumlah mahasiswa sebanyak lebih kurang 2000 mahasiswa dan mahasiswi. Politeknik Caltex Riau pernah mengalami musibah (*disaster*) pada Sistem Informasi, sehingga tidak dapat diakses oleh siapapun, dikarenakan PCR hanya memiliki satu *Database* saja, sehingga ketika server *down* atau mengalami bencana lainnya maka sistem yang ada tidak dapat diakses. Oleh karena itu maka penulis melakukan proses penyusunan rencana pemulihan dari bencana, khususnya pada layanan basis data. Proses penyusunan rencana pemulihan ini disebut Disaster Recovery Planning (DRP) atau Disaster Recovery Plan, untuk memastikan tercapainya suatu kondisi pulih dalam waktu yang ditentukan sehingga institusi mampu melanjutkan fungsinya dengan kerugian minimal.

Data yang tersimpan pada *database* harus memiliki cadangan terhadap ancaman dan gangguan. Untuk mengatasi kehilangan data perlu dilakukan replikasi terhadap sistem *database*. Mekanisme yang dipakai ialah *Master to Master MySQL Replication*. Ini adalah teknik yang digunakan oleh perangkat lunak manajemen *database* sehingga memiliki salinan dari satu *database* yang disimpan di komputer yang berbeda dapat digunakan dan diperbarui oleh banyak pengguna dengan cara yang tidak terpusat. Ketika perubahan dibuat ke *database* yang menggunakan replikasi multi-master, perubahan yang dibuat ditransmisikan ke semua komputer lain di jaringan sehingga semua salinan *database* yang digunakan diperbarui.

Dengan metode ini, salah satu komputer berfungsi sebagai master server pertama dan yang lainnya berfungsi sebagai master server kedua. Pada prosesnya, setiap komputer akan dapat *write* dan *read* data didalam *database*. Apabila kita melakukan perubahan data pada master Server 1, maka otomatis data pada master Server 2 akan berubah. Begitu juga jika kita melakukan perubahan data pada master Server 2, *database* pada master Server 1 akan berubah. Artinya setiap master Server 1 dan master Server 2 akan dapat mengubah dan menambah data pada *database* yang akan didistribusikan. Diharapkan dengan adanya metode Master to Master menggunakan skenario Disaster Recovery Plan dapat diterapkan agar *database* memiliki cadangan pada server master yang berbeda.

2. LANDASAN TEORI

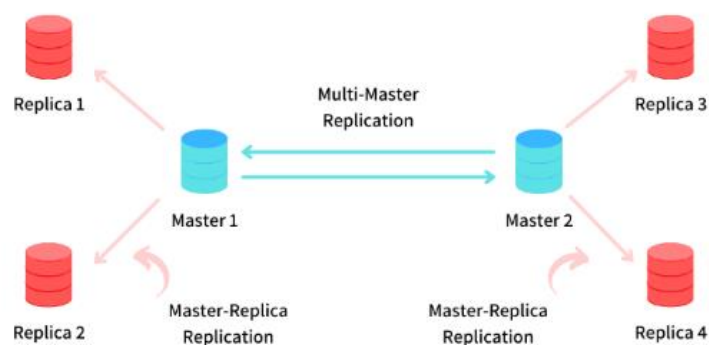
Database adalah sebuah tempat penyimpanan yang besar dimana terdapat kumpulan data, yang tidak hanya berisi data operasional tetapi juga deskripsi data. Seperti yang disampaikan oleh Connolly dan Begg (2010, p.65), bahwa database adalah kumpulan data yang saling terhubung secara logis dan deskripsi dari data tersebut, dirancang untuk menemukan informasi yang dibutuhkan oleh sebuah organisasi. Database ini digunakan tidak hanya oleh satu orang maupun satu departemen, database dapat digunakan oleh seluruh departemen dalam perusahaan. Database ini akan menjadi sumber data yang digunakan secara bersama dalam perusahaan. Hal ini kembali ditegaskan oleh Connolly dan Begg (2010, p.65), database tidak lagi dimiliki oleh satu departemen tetapi sumber perusahaan yang saling berbagi.

Disaster Recovery Planning adalah prosedur untuk merespons suatu keadaan darurat, menyediakan cadangan operasi selama gangguan terjadi, dan mengelola pemulihan dan menyelamatkan proses sesudahnya. Sasaran pokok DRP adalah untuk menyediakan kemampuan dalam menerapkan proses kritis di lokasi lain dan mengembalikannya ke lokasi dan kondisi semula dalam suatu batasan waktu yang memperkecil kerugian kepada organisasi, dengan pelaksanaan prosedur *recovery* yang cepat.

Docker adalah salah satu proyek FLOSS (*free/library/open/source software*) yang mengembangkan *platform* teknologi virtualisasi berbasis container. Docker menggunakan fitur isolasi sumber daya dari kernel linux seperti cgroups dan namespaces untuk memberi akses container secara independen untuk berjalan di dalam sebuah system operasi seperti Linux, menghindari *overhead* ketika menjalankan mesin virtual (Apridayanti, Isnawaty, & Saputra, 2018).

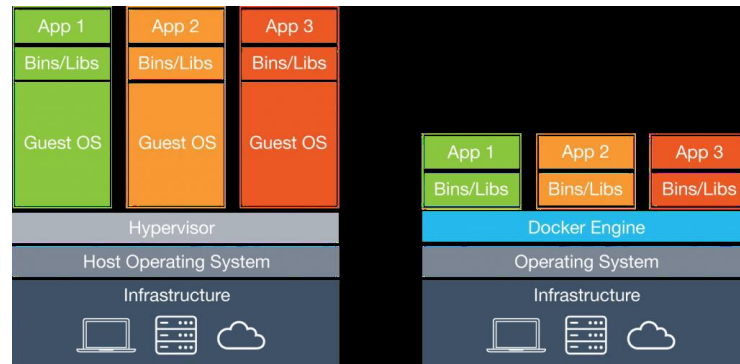
Replikasi adalah suatu teknik untuk melakukan penggandaan dan pendistribusian data dan objek-objek dari satu database ke database lain dan melaksanakan sinkronisasi antara database sehingga konsistensi data dapat terjamin. Dengan menggunakan teknik replikasi ini, data dapat didistribusikan ke lokasi yang berbeda melalui koneksi jaringan lokal maupun internet.

Terdapat dua teknik replikasi pada database yaitu Single Master Replicated (Master-to-Slave) dan Multi Master Replicated (Master-to-Master).



Gambar 1 Ilustrasi Replikasi Master-to-Master

Container adalah virtualisasi pada level sistem operasi dimana tiap proses atau aplikasi yang dijalankan tiap *container* memiliki kernel yang sama. *Container* adalah sebuah abstraksi *layer* aplikasi yang membungkus *package* diperlukan oleh aplikasi. *Container* merupakan suatu sistem operasi virtualisasi yang ringan yang dan menjadi alternatif dari teknik virtualisasi berbasis *hypervisor*. *Container* berbeda dengan virtualisasi yang memerlukan sistem operasi tersendiri untuk masing-masing mesin virtual, sedangkan banyak *container* dapat berbagi sistem operasi yang sama.

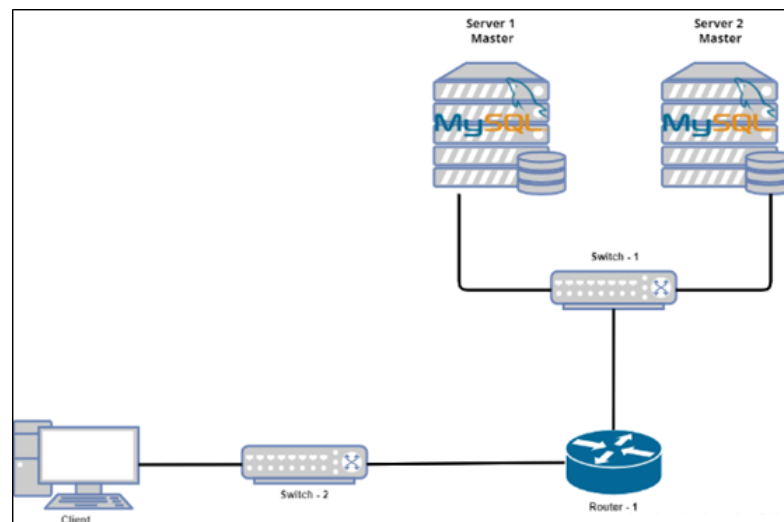


Gambar 2 Perbandingan Container dan Virtual Machine

Ubuntu adalah salah satu distribusi Linux yang berbasis Debian dan didistribusikan menjadi perangkat lunak sistem operasi yang bebas atau disebut juga sistem operasi yang berbasis Linux Debian (Husen & Surbakti, 2020). Ubuntu adalah sistem operasi pertama yang berkomitmen pada rilis terjadwal dengan irama yang dapat diprediksi, setiap enam bulan yang dimulai Oktober 2004.

3. PERANCANGAN DAN HASIL

3.1 Topologi



Gambar 3 Skema Topologi

3.2 IP Address

Tabel 1 IP Address

Perangka t	Interfac e	IP Address	Subnet Mask
SRV-1	Eth0	103.19.x.y1	255.255.255.z
SRV-2	Eth0	103.19.x.y2	255.255.255.z
Router - 1	Eth1	IP Publik PCR	IP Publik PCR
Switch - 1	Eth0	-	-
Switch - 2	Eth0	-	-
Client	Eth0	192.168.100.10	255.255.255.0

3.3 Spesifikasi Perangkat Server

Tabel 2 Spesifikasi Server Fisik

Uraian	Spesifikasi
Processor	Intel(R) Xeon(R) CPU E5-2620 v4 32 Core @ 2.10GHz
Hard Disk	2 TB
Memory	256 GB
Operating System	Proxmox 7
IP Address	IP Public PCR

Tabel 3 Spesifikasi Server Master 1 dan Master 2

Uraian	Spesifikasi
Processor	Intel(R) Xeon(R) CPU E5-2620 v4 32 Core @ 2.10GHz
Hard Disk	1 TB
Memory	8 GB
Operating System	Ubuntu 20.04
IP Address	IP Public PCR
Software	Docker + KVM

3.4 Pengujian Replikasi dan Query MySQL

Skenario pengujian replikasi yang dilakukan adalah dengan membuat table pada server database dan memastikan bahwa ter-replikasi, lalu melakukan pengujian replikasi terhadap *query insert*, *update* dan *delete* pada table yang dibuat, yang pemeriksaannya dilakukan pada setiap server master.

Pemeriksaan query pada basis data dengan Langkah-langkah sebagai berikut :

1. Jalankan query *insert* data pada master 1 lalu lakukan pemeriksaan pada master 2, apakah ada data yang sama atau tidak. Kemudian lakukan sebaliknya. Jika ada, maka pengujian sudah bisa dikatakan berhasil.
2. Begitu juga dengan *query update*. Jalankan query *update* data pada master 1 lalu melakukan pemeriksaan pada master 2, apakah ada data yang sama atau tidak. Lakukan sebaliknya, jika ada, maka pengujian *update* sudah bisa dikatakan berhasil.
3. Selanjutnya lakukan *query delete*. Pada pengujian ini akan melakukan penghapusan data pada master 1 lalu periksa kembali pada master 1, apakah ada data yang telah dihapus tadi tidak muncul lagi. Kemudian juga melakukan pengecekan di master 2, apakah data yang telah dihapus pada master 1 telah berhasil dihapus juga pada master 2. Jika data tidak ada, maka pengujian *delete* sudah bisa dikatakan berhasil, lakukan juga sebaliknya.

3.4.1 Hasil Pengujian Replikasi Pembuatan Table dan Query MySQL

Hasil pengujian replikasi pembuatan tabel pada salah satu master dan tereplikasi secara otomatis pada master yang lain adalah seperti gambar 4 .

```

root@mysql2: /home/mysql
mysql> CREATE TABLE Student (
->   PersonID int,
->   LastName varchar(255),
->   FirstName varchar(255),
->   Address varchar(255),
->   City varchar(255)
-> );
Query OK, 0 rows affected (0.05 sec)

mysql> show tables;
+-----+
| Tables_in_mydata |
+-----+
| Student          |
| article          |
| auth_assignment  |
| auth_item        |
| auth_item_child  |
| auth_rule        |
| kasus           |
| lab              |
| migration        |
| pinjamlab        |
| user             |
+-----+
11 rows in set (0.01 sec)

root@mysql1: /home/mysql
mysql> show tables;
+-----+
| Tables_in_mydata |
+-----+
| Student          |
| article          |
| auth_assignment  |
| auth_item        |
| auth_item_child  |
| auth_rule        |
| kasus           |
| lab              |
| migration        |
| pinjamlab        |
| user             |
+-----+
11 rows in set (0.01 sec)

mysql>
    
```

Gambar 4 Pembuatan tabel student pada Master 2 dan tereplikasi pada Master 1

Hasil pengujian replikasi query insert update dan delete pada master1 dan master dua serta kebalikannya dapat dilihat pada tabel 4, 5 dan 6.

Tabel 4 Replikasi Query Insert pada Master 1 dan Master 2

Percobaan	Master 1	Master 2	Status
1	Insert Data 1	Data 1 Replicated	Berhasil
2	Insert Data 2	Data 2 Replicated	Berhasil
3	Insert Data 3	Data 3 Replicated	Berhasil
4	Insert Data 4	Data 4 Replicated	Berhasil
5	Insert Data 5	Data 5 Replicated	Berhasil
6	Insert Data 6	Data 6 Replicated	Berhasil
7	Insert Data 7	Data 7 Replicated	Berhasil
8	Insert Data 8	Data 8 Replicated	Berhasil
9	Insert Data 9	Data 9 Replicated	Berhasil
10	Insert Data 10	Data 10 Replicated	Berhasil

Percobaan	Master 2	Master 1	Status
11	Insert Data 11	Data 11 Replicated	Berhasil
12	Insert Data 12	Data 12 Replicated	Berhasil
13	Insert Data 13	Data 13 Replicated	Berhasil
14	Insert Data 14	Data 14 Replicated	Berhasil
15	Insert Data 15	Data 15 Replicated	Berhasil
16	Insert Data 16	Data 16 Replicated	Berhasil
17	Insert Data 17	Data 17 Replicated	Berhasil
18	Insert Data 18	Data 18 Replicated	Berhasil
19	Insert Data 19	Data 19 Replicated	Berhasil
20	Insert Data 20	Data 20 Replicated	Berhasil

Tabel 5 Replikasi Query Update pada Master 1 dan Master 2

Percobaan	Master 1	Master 2	Status
1	Update Data 1	Data 1 Updated	Berhasil
2	Update Data 2	Data 2 Updated	Berhasil
3	Update Data 3	Data 3 Updated	Berhasil
4	Update Data 4	Data 4 Updated	Berhasil

Percobaan	Master 2	Master 1	Status
11	Update Data 11	Data 11 Updated	Berhasil
12	Update Data 12	Data 12 Updated	Berhasil
13	Update Data 13	Data 13 Updated	Berhasil
14	Update Data 14	Data 14 Updated	Berhasil

5	Update Data 5	Data 5 Updated	Berhasil
6	Update Data 6	Data 6 Updated	Berhasil
7	Update Data 7	Data 7 Updated	Berhasil
8	Update Data 8	Data 8 Updated	Berhasil
9	Update Data 9	Data 9 Updated	Berhasil
10	Update Data 10	Data 10 Updated	Berhasil

15	Update Data 15	Data 15 Updated	Berhasil
16	Update Data 16	Data 16 Updated	Berhasil
17	Update Data 17	Data 17 Updated	Berhasil
18	Update Data 18	Data 18 Updated	Berhasil
19	Update Data 19	Data 19 Updated	Berhasil
20	Update Data 20	Data 20 Updated	Berhasil

Tabel 6 Replikasi Query Delete pada Master 1 dan Master 2

Percobaan	Master 1	Master 2	Status
1	Delete Data 1	Data 1 Deleted	Berhasil
2	Delete Data 2	Data 2 Deleted	Berhasil
3	Delete Data 3	Data 3 Deleted	Berhasil
4	Delete Data 4	Data 4 Deleted	Berhasil
5	Delete Data 5	Data 5 Deleted	Berhasil
6	Delete Data 6	Data 6 Deleted	Berhasil
7	Delete Data 7	Data 7 Deleted	Berhasil
8	Delete Data 8	Data 8 Deleted	Berhasil
9	Delete Data 9	Data 9 Deleted	Berhasil
10	Delete Data 10	Data 10 Deleted	Berhasil

Percobaan	Master 2	Master 1	Status
11	Delete Data 11	Data 11 Deleted	Berhasil
12	Delete Data 12	Data 12 Deleted	Berhasil
13	Delete Data 13	Data 13 Deleted	Berhasil
14	Delete Data 14	Data 14 Deleted	Berhasil
15	Delete Data 15	Data 15 Deleted	Berhasil
16	Delete Data 16	Data 16 Deleted	Berhasil
17	Delete Data 17	Data 17 Deleted	Berhasil
18	Delete Data 18	Data 18 Deleted	Berhasil
19	Delete Data 19	Data 19 Deleted	Berhasil
20	Delete Data 20	Data 20 Deleted	Berhasil

3.5 Pengujian DRP

Dalam konteks pengujian disaster recovery plan, metode master- to-master dapat digunakan untuk menguji ketersediaan data di kedua server. Untuk menguji *disaster recovery plan* digunakan metode master-to-master, dengan Langkah sebagai berikut:

1. Mematikan server utama (master 1), sebagai contoh dari skenario disaster yang dihadapi oleh server. Pengujian dilakukan dengan melakukan beberapa aktifitas query saat master 1 mati, dan master 2 dapat menerima aktifitas query tersebut, sebagai server yang aktif, mengambil alih fungsi dari master 1 yang mati.
2. Mengaktifkan Kembali master 1, lalu memeriksa apakah terjadi perubahan data berdasarkan replikasi yang dilakukan oleh master 2 ke master 1. Hasilnya data pada master 1 termutakhirkan dengan baik secara otomatis dari data yang pada master 2.

4. KESIMPULAN

Berdasarkan hasil pengujian dapat disimpulkan bahwa,

1. Replikasi Database Master to Master menggunakan docker berhasil diterapkan pada database MySQL
2. Replikasi untuk query insert, update dan delete 100% berhasil dilakukan dari master 1 to master 2 dan sebaliknya.
3. Replikasi Master to Master pada skenario DRP (master 1 mati dan master 2 aktif) dapat secara otomatis melakukan replikasi system user pada database MySQL.

5. DAFTAR PUSTAKA

- [1] Connolly dan Begg. (2020). Implementasi virtualisasi server berbasis docker container. *PROSISKO: Jurnal Pengembangan Riset dan Observasi Sistem Komputer*, 7(2), 165-175
- [2] Apridayanti, Isnawati, & Sputra, (2018). Desain dan Implementasi Virtualisasi Berbasis Docker untuk Deployment Aplikasi Web. *Jurnal Semantik Vol.4 No.2* 2018.
- [3] Husen, Z., & Surbakti, M. S. (2020). *Membangun Server dan Jaringan Komputer dengan Linux Ubuntu*. Syiah Kuala University Press.
- [4] Jasweni, (2018)“Implementation of MariaDB Database Replication by Using Cluster Technology”, <https://opac.pcr.ac.id>.
- [5] Kurniadi, W., Djusmin, V. B., & Yasir, F. N. (2021). Implementasi Sistem Replikasi Database Terdistribusi pada Server Sistem Informasi Data Wisuda Universitas Cokroaminoto Palopo. *PROSIDING SEMANTIK*, 3(1), 258-262.
- [6] Purwanto, E, (2012). Perbandingan Strategi Replikasi Pada Sistem Basis Data Terdistribusi. *JURNAL MAHASISWA TI S1*,1-8.
- [7] Sipahutar, C. (2019). Implementasi Disaster Recovery Plan Menggunakan Cloud Computing Pada Web Server, <https://opac.lib.pcr.ac.id>.
- [8] Yuliadi,B, (2016). Rancangan Disaster Recovery Pada Instansi Pendidikan Studi Kasus Universitas Mercu Buana, *JURNAL TEKNIK INFORMATIKAVOL. 9 NO. 1, APRIL 2016*
- [9] Huang, J., Fitriasia, Y., & Fadhly Ridha, M. A. (2018). Implementasi Multi Server Data Storage Pada Cloud Computing. *Jurnal Komputer Terapan*, 4(2), 1–9.
- [10] Edy Suranta G.S, S. P. (2017). Distributed Replicated Block Device (DRBD) sebagai Alternatif High Availability Data Replication pada Cloud Computing. *Jurnal Komputer Terapan*, 3(1), 59–68.
- [11] Ridha, M. A. F., & Suhatman, R. . (2022). Kubernetes Cluster Performance Comparison with KVM, Vagrant and LXD Virtualization. *Jurnal Komputer Terapan*, 8(1), 151–157.
- [12] D. Dhining, Y. Rokhayati, and D. Kurniawan, (2017). Penerapan Replikasi Data pada Aplikasi Ticketing Menggunakan Slony PostgreSQL, *JAIC*, vol. 1, no. 2, pp. 9-18, Dec. 2017.
- [13] M. H. Alifian dan D. Priharsari, (2021). Penyusunan Disaster Recovery Plan (DRP) menggunakan framework NIST SP 800-34 (Studi Kasus pada Perusahaan IT Nasional), *J-PTIHK*, vol. 5, no. 10, hlm. 4673–4679, Okt 2021.
- [14] Suprianto, adi. (2019). Penyusunan Disaster Recovery Plan (DRP) berdasarkan framework NSIT SP 800-34 studi kasus departemen teknologi informasi PT Pupuk Kalimantan Timur. S1. Universitas Brawijaya