Jurnal Manajemen Informatika & Sistem Informasi (MISI)
Volume 8, Nomor 2, Juni 2025.
ISSN 2614-1701 (media cetak)
ISSN 2614-3739 (media online)
DOI : 10.36595/misi.v5i2
http://e-journal.stmiklombok.ac.id/index.php/misi

IMPLEMENTASI PENGGUNAAN HAPROXY LOAD BALANCING DAN FAIL2BAN PADA SERVER MENGGUNAKAN VIRTUAL PRIVATE SERVER

Maulid Putra Perdana[1], Khairan Marzuki[2], Ondi Asroni,[3], Husain[4], Lilik widyawati[5], Lalu Zazuli Azhar Mardedi[6]

[123456]Program Studi Ilmu Komputer, Universitas Bumigora

Jl. Ismail Marzuki No.22, Universitas Bumigora, Cilinaya, Cakranegara, Mataram 83127
maulidefek426@gmail.com[1], khairan.marzuki@universitasbumigora.ac.id[2],
ondiansori@universitasbumigora.ac.id[3], husain@universitasbumigora.ac.id[4],
lilikwidya@universitasbumigora.ac.id[5], zazuli@universitasbumigora.ac.id[6]

Abstract

As the need for web-based services increases, server performance and security become crucial aspects in managing network infrastructure. High server load can cause performance degradation to service failure (server down). Therefore, a Load Balancing system is needed to distribute traffic evenly and an additional security system to protect the server from the threat of attacks. This study analyzes the implementation of *Load Balancing* using *HAProxy* with the *Round Robin algorithm* to improve server performance and the implementation of Fail2Ban as a security system to prevent someone from trying to log in repeatedly on a *Virtual Private Server* (VPS). The research method used is the *Network Development Life Cycle* (NDLC), which includes needs analysis, system design, simulation. Testing was carried out with *Apache JMeter* to measure server performance and simulation of failed login attempts was carried out with *PuTTy* to test the effectiveness of *Fail2Ban*. The results of the study showed that *HAProxy load balancing* successfully distributed the load evenly, increased service availability, and reduced response time. In addition, the implementation of *Fail2Ban* proved effective in detecting and blocking IP addresses that repeatedly failed to log in, thereby increasing server security. This research contributes to the optimization of VPS-based server performance and security, which can be applied to various web services such as e-commerce, blogs, corporate systems, and schools.

Keywords : *Load Balancing, HAProxy, Fail2Ban, Server Security, Virtual Private Server (VPS).*

Abstrak

Seiring meningkatnya kebutuhan layanan berbasis web, performa dan keamanan *server* menjadi aspek krusial dalam pengelolaan infrastruktur jaringan. Beban yang tinggi pada *server* dapat menyebabkan penurunan kinerja hingga kegagalan layanan (*server down*). Oleh karena itu, diperlukan sistem *Load Balancing* untuk mendistribusikan lalu lintas secara merata dan sistem keamanan tambahan untuk melindungi *server* dari ancaman serangan. Penelitian ini menganalisis penerapan *Load Balancing* menggunakan *HAProxy* dengan algoritma *Round Robin* untuk meningkatkan kinerja *server* serta penerapan *Fail2Ban* sebagai sistem keamanan dalam mencegah seseorang yang mencoba *login* berulang kali pada *Virtual Private Server* (VPS). Metode penelitian yang digunakan adalah *Network Development Life Cycle* (NDLC), yang mencakup analisis kebutuhan, desain sistem, simulasi prototype. Pengujian dilakukan dengan *Apache JMeter* untuk mengukur kinerja *server* dan simulasi percobaan *login* gagal dilakukan dengan *PuTTy* untuk menguji efektivitas *Fail2Ban*. Hasil penelitian menunjukkan bahwa *HAProxy load balancing* berhasil mendistribusikan beban secara merata, meningkatkan ketersediaan layanan, dan mengurangi waktu *respons*. Selain itu, penerapan *Fail2Ban* terbukti efektif dalam mendeteksi serta memblokir alamat IP yang berulang kali gagal melakukan *login*, sehingga meningkatkan keamanan *server*.

Jurnal Manajemen Informatika & Sistem Informasi (MISI)
Volume 8, Nomor 2, Juni 2025.
ISSN 2614-1701 (media cetak)
ISSN 2614-3739 (media online)
DOI : 10.36595/misi.v5i2
http://e-journal.stmiklombok.ac.id/index.php/misi

Penelitian ini memberikan kontribusi dalam optimalisasi performa dan keamanan *server* berbasis VPS, yang dapat diterapkan pada berbagai layanan web seperti *e- commerce*, blog, sistem perusahaan, maupun sekolah.

Kata kunci : *Load Balancing, HAProxy, Fail2Ban, Keamanan Server, Virtual Private Server (VPS).*

1. PENDAHULUAN

Seiring berkembangnya teknologi informasi, kebutuhan pengguna pun meningkat, seperti kebutuhan akan permintaan informasi di Internet maupun website. Menjalankan sebuah situs web memerlukan *server* yang aktif selama 24 jam sehari untuk menangani permintaan dari pengguna. Umumnya, *server* web berjalan pada sistem operasi *Linux* dan ditempatkan di institusi, perusahaan, atau sekolah yang memerlukan layanan web. [1]. Ketika permintaan informasi di Internet meningkat, beban *traffic* di *server* akan meningkat, sehingga berpotensi membebani beban kerja layanan *server*. Sehingga harus adanya sebuah *server* dengan kinerja sangat andal yang dapat mengelola ratusan bahkan hingga ribuan permintaan layanan untuk menghindari terjadinya *server down*. *Server* down merupakan sebuah masalah yang paling umum terjadi ketika banyaknya permintaan layanan tersebut. [2]. Selain itu juga, ketika teknologi dengan berbagai fungsi dan kemudahan berkembang semakin pesat, dengan begitu kualitas keamanan *server* perlu ditingkatkan, pengungkapan informasi kepada orang yang tidak berwenang sangat merugikan pemilik informasi. Karena pada saat sekarang banyak sekali terjadinya penyerangan yang dilakukan pada sebuah *server*, dengan menggunakan sebuah *tool hacking* penyerang dapat dengan mudah melakukan pembobolan pada *server* sehingga penyerang dengan leluasa menguasai *server* dan mendapatkan akses penuh dari *server* dan mengambil informasi yang telah disimpan pada *server* [3]. Pada penelitian yang dilakukan oleh [4] menerapkan load balancing untuk mendistribusikan *traffic* pada dua atau lebih *server*, atau sumber daya, untuk mencapai pemanfaatan sumber daya yang optimal, memaksimalkan *bandwidth*, mengurangi waktu *respons*, dan menghindari kemacetan proses pendistribusian beban *traffic* secara merata antar *server*. Kemudian penelitian yang dilakukan oleh [1] sistem *load balancing* atau *clustering* menggunakan *HAProxy* dengan algoritma *round robin* dapat berjalan dengan baik ketika diakses oleh 500 *user* secara bersamaan. Pada penelitian tersebut menjelaskan bahwa harus ada pengembangan dari segi keamanan *server* perusahaan atau organisasi. Pada penelitian yang dilakukan oleh [3] penerapan fail2ban pada *server* terbukti dapat mengatasi masalah penyerangan yang dilakukaan pada sebuah *server* dengan melakukan pemblokiran IP penyerang sehingga *server* dapat dipastikan aman. Kemudian penelitian yang dilakukan oleh [5] penerapan *Firewall Sophos XG430* tidak dapat bekerja dengan secara maksimal sehingga tidak dapat memblokir serangan justru memblokir jaringan untuk akses aplikasi akibat dari itu *file-file* pegawai dan database *server* pada Diskominfo kota Sumedang tidak dapat diakses. Selanjutnya penelitian yang dilakukan oleh [6] menjelaskan bahwa penerapan *Instrument Detection System* (IDS) dapat mendeteksi jika terjadinya sebuah serangan yang dilakukan pada *server* dan salah satu perangkat lunak IDS itu sendiri yaitu *fail2ban* yang digunakan untuk mengenali upaya *login* yang gagal dan melakukan pemblokiran pada alamat IP sumber serangan. *Fail2ban* menjadi sebuah solusi untuk menigkatan keamanan *server* guna untuk menghindari terjadinya penyerangan pada *server*, dengan kemampuan *fail2ban* diharapkan dapat menjaga dan membatasi penyerangan yang dilakukan pada *server* agar tetap aman tanpa adanya gangguan penyerang dari orang yang tidak bertanggung jawab. Tujuan dari penelitian ini, Menganalisis performa haproxy load balancing algoritma Round Robin dalam mendistribusikan beban merata ke setiap backend *server* dengan mengukur beberapa parameter metrik performa seperti, presentase penggunaan *CPU*, presentase penggunaan *RAM*, *throughput*, dan *response time* kemudian menganalisis fail2ban yang dapat melindungi *server* dari percobaan login berulang kali di IDcloudhost. Hasil dari penelitian ini diharapkan dapat dijadikan referensi bagi mahasiswa tentang bagaimana meningkatkan ketersediaan dan kinerja website, efisiensi

Jurnal Manajemen Informatika & Sistem Informasi (MISI)
Volume 8, Nomor 2, Juni 2025.
ISSN 2614-1701 (media cetak)
ISSN 2614-3739 (media online)
DOI : 10.36595/misi.v5i2
http://e-journal.stmiklombok.ac.id/index.php/misi

pembagian beban kerja *server*, mengurangi resiko terjadinya *server* down dan keamanan pada *server*.

2. TINJAUAN PUSTAKA

2.1 Jaringan Komputer

Jaringan komputer adalah kumpulan koneksi berbagai komputer dan perangkat, seperti *router*, *switch*, dll, yang dihubungkan melalui kabel atau media nirkabel [7]. Komputer, printer, atau perangkat periferal yang terhubung ke jaringan disebut node. Jaringan komputer terdiri dari dua atau lebih unit komputer yang dapat saling bertukar data/informasi atau berbagi sumber daya seperti file, printer, media penyimpanan, dan lain- lain [8].

2.2 *Server*

Server adalah suatu perangkat yang tujuannya untuk memberikan layanan kepada beberapa perangkat lainnya. Beberapa *server* bekerja keras sementara yang lain bekerja dengan ringan. Tidak semua kebutuhan *Server* perlu dimaksimalkan sebaik mungkin, Beberapa persyaratan dapat diminimalkan untuk menghemat biaya. *Server* yang lemah, tidak memadai, atau tidak dapat diandalkan dapat mengakibatkan kerugian yang signifikan, oleh karena itu sebelum membuat *server*, ada baiknya mempertimbangkan beberapa faktor *server* harus disesuaikan dengan persyaratan kompatibilitasnya dengan perangkat keras jaringan lain, aktivitas bisnis yang dilakukan, dan jumlah pengguna yang dilayaninya, serta kinerja perangkat keras yang optimal, andal, dan terjamin juga harus dilengkapi dengan perangkat lunak pendukung [9].

2.3 *HAProxy*

Teknologi yang saat ini banyak digunakan untuk *load balancing* atau penyeimbangan pada web *server* adalah *haproxy*. *Haproxy* atau *High Availability Proxy* adalah penyeimbang beban TCP dan HTTP open source cepat dan perangkat lunak *server proxy* yang digunakan oleh situs web terkenal seperti *Github, StackOverflow, Reddit, Tumblr*, dan *Twitter* [10]. Oleh karena itu, *Haproxy* adalah solusi populer untuk mendukung penyeimbangan beban dan kebutuhan *failover server* web [11]. Selain penyeimbangan beban, *Haproxy* dapat menangani kegagalan yang disebabkan oleh salah satu *server* tidak dapat menggunakan *Haproxy*, yang mendukung banyak algoritma [12]. Sistem *Haproxy* bekerja dengan membagi permintaan masuk ke beberapa *server backend* yang tersedia, ini mendistribusikan beban *server* secara merata di antara *server*. *HAProxy* dirancang khusus untuk digunakan di lingkungan jaringan yang padat dan kompleks. Tujuannya adalah untuk menjaga ketersediaan suatu aplikasi atau layanan dan memastikan akses cepat dengan memastikan bahwa semua permintaan dari klien diproses secara efisien dan dialihkan ke *server backend* yang paling tepat [12].

2.4 *Load Balancing*

Load balancing merupakas proses mendistribusikan beban layanan ke sekelompok *server* atau perangkat jaringan sebagai *respons* terhadap permintaan dari pengguna [10] tujuannya adalah untuk mencapai pemanfaatan sumber daya yang optimal, memaksimalkan *throughput*, mengurangi waktu respons, dan menghindari kelebihan beban [4]. Dengan bantuan layanan *load balancing*, sumber daya jaringan dapat didistribusikan ke beberapa host lain untuk menjaga kinerja seluruh jaringan komputer tetap stabil. Ketika pengguna mengakses suatu *server*, hal tersebut justru memberikan beban pada *server* karena harus memproses permintaan yang ditujukan kepada pengguna. Semakin banyak pengguna, semakin banyak pemrosesan yang dilakukan, *server* membuka sesi komunikasi sehingga pengguna dapat menerima layanan dari *server*. Sekalipun hanya satu *server* yang kelebihan beban, *server* tersebut tidak dapat melayani banyak pengguna karena daya komputasinya yang terbatas [12].

2.5 *Virtual Private Server* (VPS)

Virtual Private Server dikenal dengan singkatan VPS. Ini adalah metode membagi komputer *server* menjadi beberapa *server* yang terlihat seperti *server* independen namun bertindak seperti satu komputer. Meskipun memiliki beberapa akun VPS di satu komputer, satu akun tidak berinteraksi dengan akun lainnya [13]. VPS mengurangi biaya dengan menyediakan spesifikasi *server* yang kuat dan fleksibel, dari rendering video hingga *server game* dan *hosting* situs web, VPS menawarkan berbagai kegunaan dan digunakan oleh banyak pengguna berbeda untuk memenuhi kebutuhan pemrosesan, komputasi, dan penyimpanan pribadi dan

Jurnal Manajemen Informatika & Sistem Informasi (MISI)
Volume 8, Nomor 2, Juni 2025.
ISSN 2614-1701 (media cetak)
ISSN 2614-3739 (media online)
DOI : 10.36595/misi.v5i2
http://e-journal.stmiklombok.ac.id/index.php/misi

profesional mereka yang semakin meningkat [14].

2.6 *Fail2ban*

Fail2ban adalah aplikasi yang mencegah alamat IP login ke SmartVPS Linux setelah terlalu banyak upaya login yang gagal [15]. *Fail2ban* mendeteksi upaya login yang gagal dan memblokir alamat IP penyerang [6]. *Fail2ban* saat ini tersedia di hampir semua repositori distribusi. Karena ini *tool fail2ban* telah mengimplementasikan metode IDPS (*Intrusion Detection & Prevention System*), *tool* ini dapat digunakan sebagai sistem deteksi intrusi dan menolak serangan dengan memblokir penyerang. IDPS (*Intrusion Detection & Prevention System*) sendiri merupakan pengembangan lebih lanjut dari IDS (*Intrusion Detection System*). Secara default, *fail2ban* hanya dapat digunakan pada satu *server* dan tidak dapat terhubung ke *server* lain [3].

2.7 *Apache Jmeter*

Apache JMeter adalah perangkat lunak sumber terbuka berbasis Java yang dirancang untuk menguji kinerja dan fungsionalitas berbagai sistem. Awalnya dikembangkan untuk menguji aplikasi web, namun kini telah berkembang untuk mendukung pengujian lainnya, seperti FTP, server basis data, dan objek Java. *JMeter* mampu mensimulasikan jumlah pengguna yang mengakses suatu sistem serta menyediakan laporan analisis hasil pengujian secara mendetail [16].

3. METODOLOGI PENELITIAN

3.1. Tahapan Penelitian

Peneitian ini menggunakan metodologi **NDLC** (*Network Development Life Cycle*) dengan tahapan *analysis, disign, simulasi prototype*, memiliki beberapa alasan yang dapat mendukung efektivitas dan keberhasilan pengembangan sistem, terutama dalam konteks sistem berbasis teknologi informasi atau jaringan. Penelitian ini bertujuan untuk mengurangi beban *server* menggunakan *HAProxy Load Balancing* serta mendeteksi dan mencegah *user* yang berusaha *login* dengan tujuan ingin mengambil atau memodifikasi *file server* melalui *ssh* menggunakan *Fail2ban*. Metodologi ini dipilih karena kerangka kerja sistematisnya dalam mengidentifikasi, mendesain, dan menguji mengurangi beban *server* serta keamanan *server*, serta memantau secara *realtime* presentase penggunaan *CPU*, penggunaan *RAM*, *throughput*, dan *response time*.

3.2. Pengumpulan Data

Pada tahapan pengumpulan data yang dilakukan penulis untuk mendapatkan informasi mengenai skripsi yang dikerjakan menggunakan metode studi literatur, yaitu mempelajarin serta memahami artikel ilmiah yang membahas *haprxoy load balancing* algoritma round robin dan *fail2ban*. Selain itu penulis juga menggunakan data dan informasi dari berbagai sumber antara lain internet, buku, paper, e-book, dan artikel ilmiah.

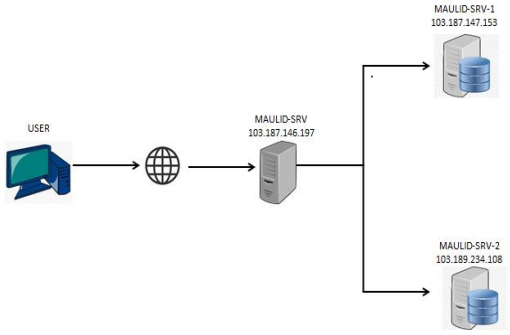
3.3. Tahap Desain

Penelitian ini menggunakan metode eksperimen untuk menguji efektifitas dari penggunaan haporxy load balancing dan fail2ban.

3.4. Simulasi Prototype

Pada tahap simulasi prototype, ada beberapa simulasi pengujian yang akan dilakukan pada penelitian ini yaitu simulasi uji coba pada *haproxy load balancing* dan uji coba pada *fail2ban*.

1. Rancang Uji Coba Haproxy load balancing



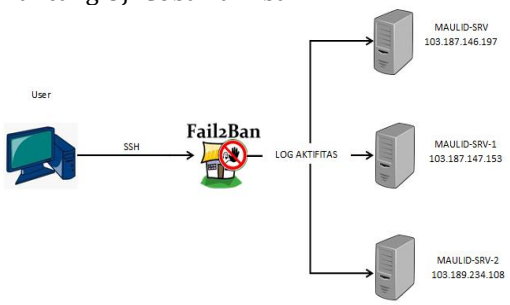
Gambar 1. Topologi Haproxy

Terdapat tiga server dinamai MAULID-SRV, MAULID-SRV-1, dan MAULID-SRV-2, yang dimana MAULID-SRV berfungsi sebagai server HAProxy load balancing, MAULID-SRV-1, dan MAULID-SRV-2 berfungsi sebagai *backend server*. Setiap permintaan dari user melalui internet yang mengakses IP dari

Jurnal Manajemen Informatika & Sistem Informasi (MISI)
Volume 8, Nomor 2, Juni 2025.
ISSN 2614-1701 (media cetak)
ISSN 2614-3739 (media online)
DOI : 10.36595/misi.v5i2
http://e-journal.stmiklombok.ac.id/index.php/misi

MAULID-SRV sebagai server Haproxy telah dikonfigurasikan load balancing menggunakan algoritma round robin yang dapat mendistribusikan permintaan dari user secara kepada MAULID-SRV-1 dan MAULID-SRV-2 sebagai backend server. Ketika Load Balancer menerima permintaan dari pengguna, permintaan pertama akan dikirim ke MAULID-SRV-1, sedangkan permintaan kedua akan diteruskan ke MAULID-SRV-2, dan permintaan ketiga akan kembali ke MAULID-SRV-1. Begitu seterusnya secara bergantian antara MAULID-SRV-1 dan MAULID-SRV-2 kemudian memproses permintaan yang diterima dan mengirimkan respons kembali ke user melalui jalur yang sama.

2. Rancang Uji Coba Fail2ban



Gambar 2. Topologi Fail2ban

Simulasi skenario *login* secara berulang kali oleh *user* melalui *SSH* akan dipantau melalui penggunaan *fail2ban* sebagai IPS (*Intrusion Prevention System*). Skenario ini terdapat tiga *server*, yaitu Maulid-srv, Maulid-srv-1 dan Maulid-srv-2 telah dilengkapi dengan *fail2ban* yang berfungsi untuk mencatat *log* aktivitas *server* secara detail, seperti percobaan *login* pada *server* beserta IP dari *user*. *Log* aktvitas yang dihasilkan dari *fail2ban* akan disimpan pada setiap *server* kemudian *fail2ban* yang telah disetting akan menganalisis lebih lanjut upaya percobaan *login* melalui *SSH* dari *user* tersebut apakah melebihi lima kali percobaan atau tidak, jika *user* melakukan percobaan *login* sebanyak lima kali dengan hasil gagal maka *fail2ban* akan otomatis melakukan ban IP *user* tersebut sehingga *user* tersebut tidak dapat mengakses *server* dalam jangka waktu yang telah ditentukan.

3.5. Rancangan Pengalamatan IP

Dalam sistem yang dirancang, setiap perangkat dan layanan diberi alamat IP tertentu agar bisa saling berkomunikasi dengan lancar selama uji coba. IP publik digunakan untuk setiap *server*, memungkinkan mereka dikenali secara unik dalam jaringan internal tanpa bentrok dengan perangkat lain. Ini memastikan koneksi tetap stabil dan aman dalam lingkungan yang dikendalikan. Pengalamatan IP telah diberikan secara otomatis oleh layanan penyedia VPS itu sendiri yaitu *IDcloudhost* sehingga memudahkan dalam melakukan pengujian melalui domain yang terdaftar pada Tabel 1.

Tabel 1. Pengalamatan IP

No	Perangkat	IP Public
1	MAULID-SRV	103.217.144.186
2	MAULID-SRV-1	103.217.144.137
3	MAULID-SRV-2	103.217.144.197

3.6. Skenario Uji Coba

Dalam skenario uji coba ini, tujuan utamanya adalah pemanfaatan kemampuan *haproxy load balancing* algoritma *roundrobin* untuk mendukung dalam menyeimbangkan beban *server* dan *fail2ban* untuk mendeteksi IP serta melakukan *ban* IP dari *user*. Lingkungan uji coba untuk *haproxy* dengan algoritma *roundrobin* terdiri dari 3 *server*, yaitu Maulid- srv, Maulid-srv-1, dan Maulid-srv-2. Maulid-srv berfungi sebagai *load balancer* yang dimana akan mendistribusikan *request* dari *user* dan membagi beban secara merata pada setiap *server backend* yaitu Maulid-srv- 1 dan Mauid-srv-2. *Server backend* akan berperan sebagai penyedia konten yang berisi wordpress. Algoritma *roundrobin* berfungsi mendistribusikan beban secara merata pada *backend server* dan pada saat melakukan *refresh* pada *wordpress* maka secara bergantian akan menampilkan konten dari Maulid-srv-1 dan Maulid-srv-2. Analisis pembagian beban secara merata kepada setiap *server* akan dilakukan menggunakan *Jmeter*, dimana *Jmeter* akan menampilakn pembagian setiap *user* secara merata kepada setiap *backend server*, dan juga *jmeter* digunakan untuk melihat *respons time*, *througput*, dan melakukan *trigger* pada metrik presentase penggunaan *CPU*, presentase

Jurnal Manajemen Informatika & Sistem Informasi (MISI)
Volume 8, Nomor 2, Juni 2025.
ISSN 2614-1701 (media cetak)
ISSN 2614-3739 (media online)
DOI : 10.36595/misi.v5i2
http://e-journal.stmiklombok.ac.id/index.php/misi

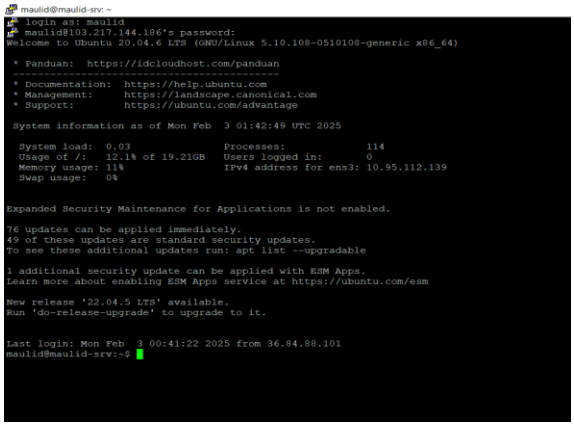
penggunaan *RAM* secara *realtime* pada IDCloudhost.

Penggunaan *Fail2ban* yang telah terinstall pada setiap *server* diharapkan dapat berperan dalam menjaga keamanan *server* dari *user* yang mencoba melakukan *login* melalui *ssh*, *fail2ban* akan mencatat dan merekam IP dari *user* melalui *log* aktivitas. *Fail2ban* yang telah disetting akan merekam *log* aktivitas *user* pada setiap *server*, dimana *user* yang telah mencoba melakukan *login* secara paksa pada *server* sebanyak lima kali memicu konfigurasi *fail2ban* untuk melakukan *ban* IP *user* tersebut sehingga pada akhirnya *user* tersebut tidak lagi dapat mengakses *server* dalam jangka waktu yang telah ditentukan.

4. HASIL DAN PEMBAHASAN

4.1. Hasil Instalasi dan Konfigurasi

Pembahasan hasil dari instalasi dan konfigurasi pada Maulid-srv, Maulid-srv-1, dan Maulid-srv-2 menggunakan Sistem Operasi Ubuntu Server 20.04 lts, dimana Maulid-srv berperan sebagai *HAProxy Load Balancing server* membagi *request* dari user secara merata pada setiap *server backend* yaitu, Maulid-srv-1 dan Maulid-srv-2. *Fail2ban* sebagai alat untuk menerapkan rule agar mendeteksi ip yang mencoba *login* pada setiap *ssh server* kemudian memblokir ip yang mencoba *login* secara berulang kali pada setiap *ssh server* secara otomatis. Selain itu juga memuat tentang pengujian rancang keamanan *server* berdasarkan skenario ujicoba.

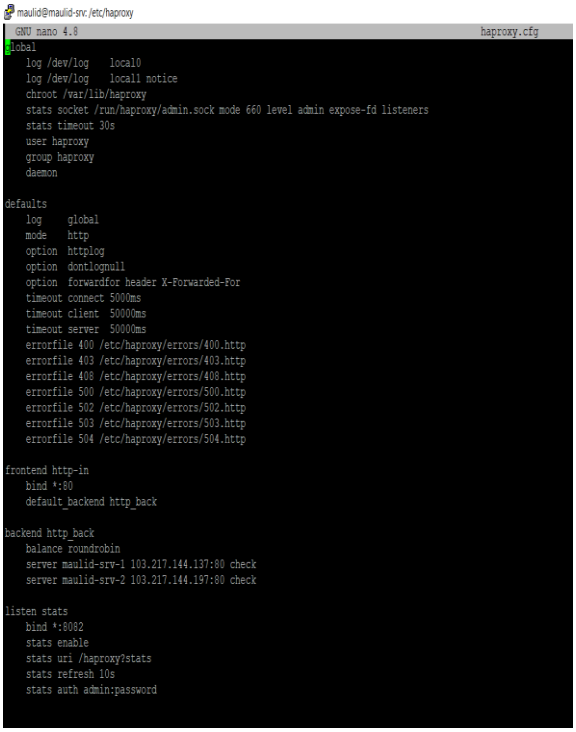


Gambar 3. Instalasi Ubuntu 20.04 lts

Pada Gambar 3 menampilkan hasil dari instalasi menggunakan sistem operasi Ubuntu 20.04 lts yang telah diinstal pada setiap server.

Gambar 4. Status Haproxy

Gambar 4 menunjukkan *haproxy.service t*elah berhasil diinstalasi dan dijalankan pada Maulid-srv.Untuk *menjalankan HAProxy* diperlukan beberapa konfigurasi agar *HAProxy* dapat berjalan sebagai alat untuk penyeimbang pada maulid-srv-1 dan maulid-srv-2

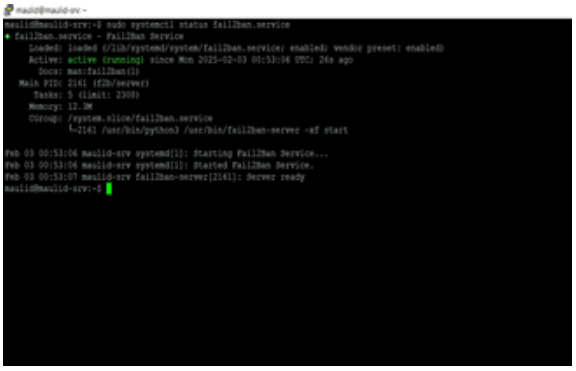


Gambar 5. Konfigurasi Haproxy

Jurnal Manajemen Informatika & Sistem Informasi (MISI)
Volume 8, Nomor 2, Juni 2025.
ISSN 2614-1701 (media cetak)
ISSN 2614-3739 (media online)
DOI : 10.36595/misi.v5i2
http://e-journal.stmiklombok.ac.id/index.php/misi

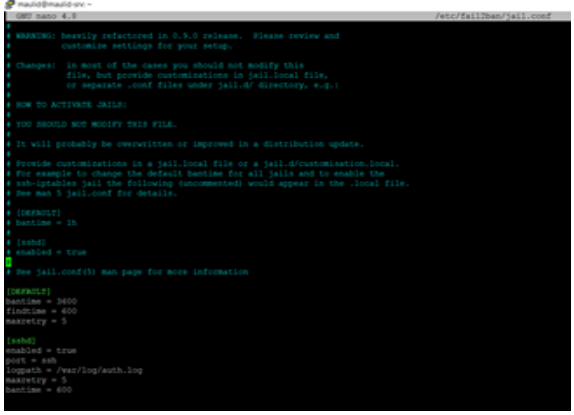
Gambar 5 menjelaskan Konfigurasi manual melibatkan penambahan *frontend* dan *backend*. Frontend mengatur bagaimana sebuah permintaan diteruskan ke *backend*, dan *backend* adalah sekumpulan *server* yang menerima permintaan. *Backend* dapat memiliki beberapa *server*, dan apabila Potensi *load* akan meningkat dapat menambahkan lebih banyak *server* ke *backend*. kapasitas dengan membagi beban ke sejumlah *server.*

Instalasi dan konfigurasi *Fail2ban* pada setiap server digunakan untuk mencegah seseorang yang mencoba login secara berulang kali pada server.



Gambar 6. Hasil Instalasi Fail2ban

Pada Gambar 6 menunjukkan hasil instalasi *fail2ban* pada mauid- srv, *dengan* begitu maulid-srv sudah dapat dijalankan seperti pada gambar yang menunjukkan status dari *fail2ban* yang terlihat *active* (*running*). Untuk menjalankan *fail2ban* diperlukan beberapa konfigurasi agar *Fail2ban* dapat berjalan sebagai alat untuk memblokir IP pengguna yang mencoba mengakses *server* dalam beberapa kali percobaan dan mengirim *log* tersebut pada *fail2ban.log*



Gambar 7. Konfigurasi Fail2ban

Gambar 7 menampilkan Menampilkan konfigurasi menunjukan rules yang akan ditetapkan untuk melindungi server dari percobaan login beruang kali yang dilakukan oleh orang lain secara berulang kali pada server.

4.2. Hasil Uji Coba Load Balancing

Uji coba *HAProxy Load Balancing* disini dilakukan untuk melihat bagaimana *algoritma Round Robin* mendistribusikan beban secara merata kepada setiap *server backend*. Untuk melihat hasil *load balancing* telah berjalan yaitu dengan akses browser menggunakan alamat IP *server HAProxy* sehingga akan menampilkan halaman beranda maulid-srv-1 kemudian *refresh* halaman beranda sehingga akan menampilkan halaman beranda maulid-srv-2 yang menunjukan *HAProxy Load Balancing algoritima Round Robin* berfungsi. Untuk memverifikasi hal tersebut akan ditunjukan pada Gambar 8

Gambar 8. Tampilan web Maulid-srv-1

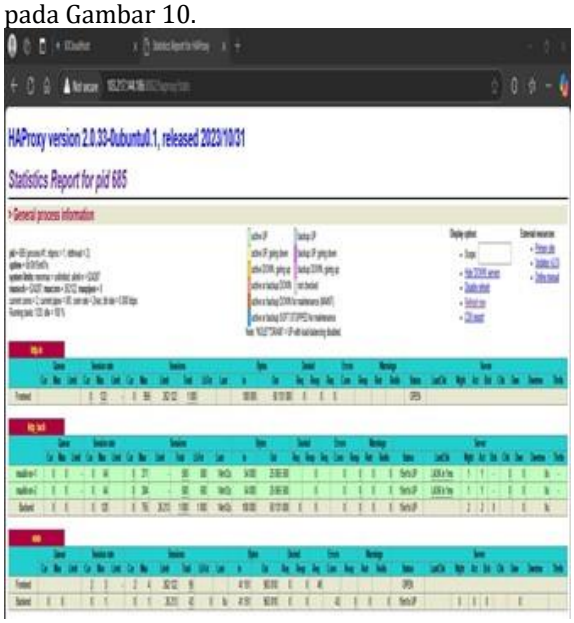
Gambar 8 menunjukkan halaman beranda maulid-srv, yang menampilkan halaman maulid-srv-1 menandakan bahwa *HAProxy load balancing* berjalan dan akan bergati ke halaman beranda maulid-srv-2 setelah dilakukan *refresh*, ditunjukkan pada Gambar 9.

Gambar 9. Tampilan web Maulid-srv-2

Jurnal Manajemen Informatika & Sistem Informasi (MISI)
Volume 8, Nomor 2, Juni 2025.
ISSN 2614-1701 (media cetak)
ISSN 2614-3739 (media online)
DOI : 10.36595/misi.v5i2
http://e-journal.stmiklombok.ac.id/index.php/misi

4.3. Hasil Uji Coba Haproxy Load Balancing Beban 100-1000

Pengujian *HAProxy Load Balancing* dilakukan untuk melihat bagaimana algoritma *Round Robin* mendistribusikan beban secara merata saat jumlah permintaan meningkat secara bertahap. Dalam *algoritma* ini, permintaan didistribusikan secara bergantian ke setiap *backend server*. Pengujian ini akan dilakukan dengan jumlah user yang meningkat secara bertahap mulai dari 100, 200, 300 hingga 1000 *user*. Fokus dari penilitian ini untuk mengukur persentase penggunaan *CPU*, penggunaan *RAM*, *throughput*, dan *Respon Time*, dengan tujuan untuk mengukur cara kerja dari *algoritma Round Robin* yang mampu mendistribusikan beban secara merata pada setiap *backend server*. Pembagian beban secara merata dapat dilihat secara langsung pada web statik *HAProxy* sepeti pada Gambar 10.



Gambar 10. Statistik Haproxy

Pada Gambar 10 dapat dilihat *HAProxy Load Balancing* dengan *Algoritma Round Robin* dapat mendistribusikan beban secara merata kepada setiap *backend server*, yang dimana pengujian dengan 1000 ditunjukkan pada tabel *Session*, *Total* masing-masing 500 kepada setiap *backend server*, dan LbTot (*Load Balancing Total*) masing-masing 500 kepada setiap *backend server*. Hal ini menunjukkan dengan jelas bagaimana *Algoritma Round Robin* dapat mendistribusikan beban secara merata kepada setiap *backend server*. Data dari Tabel 2, Tabel 3, Tabel 4, dan Tabel 5 akan menyajikan rangkaian uji coba yang telah dilakukan secara bertahap mulai dari 100 hingga 1000 *user* dengan menggunakan *Apache JMeter*.

Tabel 2. Presentase Penggunaan CPU

User	Load Balancing	Backend
100	4.46 %	12.15 %
200	8.23 %	9.08 %
300	12.69 %	23.51 %
400	16.61 %	31.09 %
500	19.74 %	35.11 %
600	24.55 %	42.53 %
700	25.57 %	35.9 %
800	26.41 %	37.53 %
900	26.66 %	46.49 %
1000	40.24 %	64.46 %

Penjelasan dari Tabel 2 yaitu, pada saat pengujian dilakukan dengan 100 *user* presentase penggunaan *CPU* dari penggunaan *Load Balancing* dan *Backend server* meningkat menjadi 4.46 % dan 12.15%, selanjutnya pada saat pengujian dilakukan dengan 500 *user* presentase penggunaan *CPU Load Balancing* meningkat menjadi 19.74% dan 35.11% pada *backend server*. Sehingga pada pengujian dengan 1000 *user* terdapat peningkatan yang sangat signifikan presentase penggunaan *CPU Load Balancing* menjadi 40.24% dan *backend server* 64.46%, dengan demikian presentase penggunaan *CPU Load Balancing* tidak terlalu meningkat seiring dengan bertambahnya *user* dengan kisaran 4.46% hingga 40.24%, sedangkan pada *backend server* menunjukkan tidak stabil dimana presentase penggunaan *CPU* berada dikisaran 12.15% hingga 64.46%. Secara keseluhuran penggunaan *load balancing algoritma Round Robin* berhasil karena mampu mendistribusikan beban secara merata pada *backend server* dengan presentase penggunaan *CPU* dibawah 45%. Pengujian *Penggunaan RAM* akan disajikan pada Tabel 3

Jurnal Manajemen Informatika & Sistem Informasi (MISI)
Volume 8, Nomor 2, Juni 2025.
ISSN 2614-1701 (media cetak)
ISSN 2614-3739 (media online)
DOI : 10.36595/misi.v5i2
http://e-journal.stmiklombok.ac.id/index.php/misi

Tabel 3. Presentase Penggunaan RAM

User	Load Balancing	Backend
100	9.5 req/sec	5.9 req/sec
200	12.6 req/sec	11.1 req/sec
300	12 req/sec	8.7 req/sec
400	6.1 req/sec	16 req/sec
500	8.3 req/sec	20 req/sec
600	4.8 req/sec	19.9 req/sec
700	5.7 req/sec	8.6 req/sec
800	11.6 req/sec	5.7 req/sec
900	10.4 req/sec	13.7 req/sec
1000	16.2 req/sec	16 req/sec

Penjelasan dari Tabel 3 yaitu, pada saat pengujian dilakukan dengan 100 *user* penggunaan *RAM* dari penggunaan *Load Balancing* dan *Backend server* meningkat menjadi 648 MB dan 812 MB selanjutnya pada saat pengujian dilakukan dengan 500 *user* penggunaan *RAM Load Balancing* meningkat menjadi 1.333 MB dan 1.423 MB pada *backend server*. Sehingga pada pengujian dengan 1000 *user* terdapat peningkatan yang sangat signifikan penggunaan *CPU Load Balanci*ng menjadi 1.436 MB dan *backend server* 1.854 MB, dengan demikian penggunaan *RAM Load Balancing* tidak terlalu meningkat seiring dengan bertambahnya *user* dengan kisaran 648 MB hingga 1.436 MB, sedangkan pada *backend server* menunjukkan tidak stabil dimana penggunaan *RAM* berada dikisaran 812 MB hingga 1.854 MB. Secara keseluhuran penggunaan *load balancing algoritma Round Robin* berhasil karena mampu mendistribusikan beban secara merata pada *backend server* dengan penggunaan *RAM* dibawah 1.500 MB. Pengujian *Throughput* akan disajikan pada Tabel 4

Tabel 4. Throughput

User	Load Balancing	Backend
100	0.67 sec/user	3.6 sec/user
200	5 sec/user	7.5 sec/user
300	2.8 sec/user	12.1 sec/user
400	22.2 sec/user	8.7 sec/user
500	2.6 sec/user	8.4 sec/user
600	15.3 sec/user	8.5 sec/user
700	45 sec/user	14.7 sec/user
800	32.5 sec/user	7.8 sec/user
900	3.22 sec/user	16.7 sec/user
1000	31.9 sec/user	6 sec/user

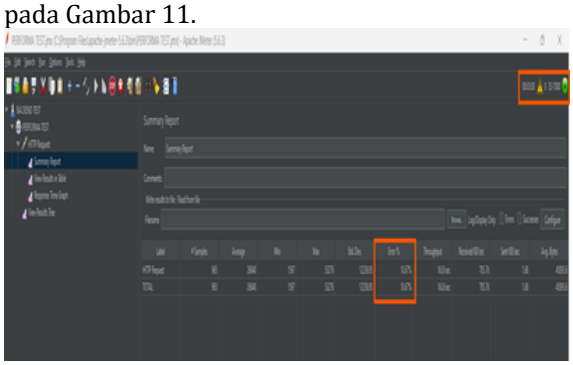
Penjelasan dari Tabel 4 yaitu, pada saat pengujian dilakukan dengan 100 *user Thoughput* dari penggunaan *Load Balancing* dan *Backend server* menjadi 9.5 req/sec dan 5.9 req/sec, selanjutnya pada saat pengujian dilakukan dengan 500 *user Thoughput Load Balancing* meningkat menjadi 8.3 req/sec dan 20 req/sec pada *backend server*. Sehingga pada pengujian dengan 1000 *user* terdapat peningkatan yang sangat signifikan *Thoughput Load Balanci*ng menjadi 16.2 req/sec dan *backend server* 16 req/sec, dengan demikian *Thoughput Load Balancing* meningkat seiring dengan bertambahnya *user* dengan kisaran 4.8 req/sec hingga 16.2 req/sec, sedangkan pada *backend server* menunjukkan *Thoughput* berada dikisaran 5.7 req/sec hingga 20 req/sec. Secara keseluhuran penggunaan *load balancing algoritma Round Robin* berhasil karena mampu mendistribusikan beban secara merata. Pengujian *Response Time* akan disajikan pada Tabel 5.

Tabel 5. Respons Time

User	Load Balancing	Backend
100	648 MB	812 MB
200	889 MB	1.044 MB
300	936 MB	1.390 MB
400	1.076 MB	1.407 MB
500	1.333 MB	1.423 MB
600	1.344 MB	1.602 MB
700	1.258 MB	1.768 MB
800	1.300 MB	1.843 MB
900	1.283 MB	1.708 MB
1000	1.436 MB	1.854 MB

Jurnal Manajemen Informatika & Sistem Informasi (MISI)
Volume 8, Nomor 2, Juni 2025.
ISSN 2614-1701 (media cetak)
ISSN 2614-3739 (media online)
DOI : 10.36595/misi.v5i2
http://e-journal.stmiklombok.ac.id/index.php/misi

Penjelasan dari Tabel 4.4 yaitu, pada saat pengujian dilakukan dengan 100 *user* Response *Time* dari penggunaan *Load Balancing* dan *Backend server* menghasilkan angka 0.67 sec/user dan 3.6 sec/user, selanjutnya pada saat pengujian dilakukan dengan 500 *user Response Time* meningkat menjadi 2.6 sec/user dan 8.4 sec/user pada *backend server*. Sehingga pada pengujian dengan 1000 *user* terdapat peningkatan yang sangat signifikan *Response Time* pada *Load Balancin*g menjadi 32.5 sec/user sedangkan *backend server* 6 sec/user, dengan demikian *Response Time Load Balancing* meningkat seiring dengan bertambahnya *user* dengan kisaran 0.67 sec/user hingga 31.9 sec/user, sedangkan pada *backend server* menunjukkan stabil dimana *Response Time* berada dikisaran 3.6 sec/user hingga 16.7 sec/user. Namun dibalik *Response Time* yang ditunjukkan oleh backend server terdapat banyak sekali persentase *Error* seperti yang ditunjukkan pada Gambar 11.



Gambar 11. Hasil Uji Coba 1000 User Backend Server

Pada Gambar 11 menunjukkan hasil uji coba yang dilakukan di *Apache JMeter* pada *backend server* dengan 1000 *user*, terlihat pada saat melakukan uji coba *backend server* tidak mampu untuk menangani 1000 *user* dengan waktu pengujian selama 3 menit *backend server* hanya mampu menangani 965 *user* dengan 35 *user* tidak mampu ditangani dan dapat diindikasikan *eror*, dari 965 *user* yang berhasil ditangani oleh *backend* terdapat presentase jumlah *eror* yang cukup banyak yaitu 10.67%

4.3. Hasil Uji Coba Fail2ban

Uji coba *Fail2ban* dilakukan untuk perlindungan terhadap *server* menjadi hal yang sangat penting, *Fail2Ban* dapat mendeteksi IP dari seseorang yang *login* secara berulnag kemudian secara otomatis memblokir alamat IP yang melakukan percobaan berulang, dengan menguji *Fail2Ban*, dapat mengevaluasi apakah alat ini mampu melindungi *server* dari ancaman yang sering terjadi, seperti *login* yang gagal berkali-kali. Uji coba ini menggunakan *PuTTy* sebagai alat untuk memantau log aktivitas dan hasil deteksi yang dilakukan *Fail2ban*, seperti yang ditunjukkan pada Gambar 12.

Gambar 12. Hasil Uji Coba Fail2ban

Pada Gambar 12 menunjukkan log aktivitas pada *fail2ban* yang dimana dilakukan uji coba melakukan *login* secara berulang kali dengan jumlah 5 kali percobaan *login* sesuai dengan konfigurasi pada *jail.conf*, *fail2ban* akan secara otomatis akan memblokir IP yang mencoba *login* pada *server* sehingga tidak dapat mencoba *login* pada *server* dalam beberapa waktu, dengan demikian pengujian terhadap penggunaan *fail2ban* dalam menjaga keamanan *server* berhasil.

5. KESIMPULAN DAN SARAN

Berdasarkan hasil pengujian dan analisis yang telah dilakukan terhadap haproxy load balancing *algoritma Round Robin* menjaga *server* agar tidak overload dan *fail2ban* melindungi *server* dari seseorang yang mencoba login secara berulang kali pada *server*. Penelitian ini mengevaluasi kinerja dari *haproxy load balancing algoritma round robin* dan *fail2ban* pada lingkungan *Virtual Private Server* (VPS) di IDCloudhost. Hasil pengujian menunjukkan bahwa penggunaan *algoritma round robin* efektif untuk beban pada *server* yang memiliki *resource* yang terbatas, dengan uji coba jumlah *user* hingga 1000 secara bersamaan mengakses *server*, *algoritma round robin* mampu membagi beban

Jurnal Manajemen Informatika & Sistem Informasi (MISI)
Volume 8, Nomor 2, Juni 2025.
ISSN 2614-1701 (media cetak)
ISSN 2614-3739 (media online)
DOI : 10.36595/misi.v5i2
http://e-journal.stmiklombok.ac.id/index.php/misi

dari 1000 user di setiap backend *server*, sehingga tidak terjadinya *overload* pada *server*. Pengujian menggunakan *fail2ban* sangat efektik dalam melindungi *server* dari seseorang yang mencoba *login* secara berulang kali, sehingga secara otomatis langsung memblokir alamat IP tersebut, dengan adanya *log* pada *fail2ban* juga memudahkan administrator untuk memantau aktivitas yang terjadi pada *server*. Penelitian ini memberikan panduan dalam memilih penggunaan *algoritma round robin* pada *haproxy load balancing* sesuai jika memliki *resource* yang sangat terbatas pada *server* untuk memastikan *server* dapat menangani berbagai kondisi beban, serta *fail2ban* yang ampuh dalam melindungi *server* dari ancaman dari luar.

Adapun saran-saran untuk pengembangan skripsi ini lebih lanjut adalah sebagai berikut:

1. Selain Round Robin, penelitian selanjutnya dapat mengeksplorasi algoritma load balancing lain, seperti Least Connection atau Dynamic Load Balancing, yang dapat menyesuaikan distribusi beban berdasarkan kapasitas dan performa masing-masing backend server.
2. Integrasi *HAProxy* dengan *Auto Scaling*, penelitian selanjutnya dapat mengevaluasi bagaimana *HAProxy* dapat bekerja dengan sistem *Auto Scaling* pada *cloud* VPS, di mana jumlah *backend server* dapat bertambah atau berkurang sesuai dengan kebutuhan lalu lintas pengguna.
3. Penerapan sistem monitoring, penelitian dapat menambahkan sistem monitoring *real-time* dengan *Grafana* atau *Prometheus* untuk memantau kinerja *HAProxy* secara lebih visual.
4. Pengujian dapat diperluas dengan menggunakan jumlah *backend server* yang lebih banyak, misalnya 3-5 *server backend*, untuk melihat seberapa efektif *HAProxy* dalam menangani beban pengguna yang lebih tinggi.
5. Penelitian selanjutnya dapat menguji efektivitas *Fail2Ban* dalam menangani berbagai jenis serangan, seperti *DDoS*, *brute force* dengan IP dinamis, atau eksploitasi *API endpoint*.
6. Selain menggunakan *iptables*, penelitian selanjutnya dapat mengeksplorasi kombinasi *Fail2Ban* dengan layanan keamanan berbasis *cloud*, seperti *Cloudflare* atau WAF (*Web Application Firewall*).

DAFTAR PUSTAKA:

[1] S. D. Y. K. Ahmad Riyan Sofyan, "Implementasi Load Balancing Web Server menggunakan Haproxy pada Virtual Server Direktorat SMK Kemendikbudristek," J. Pendidik. Tambusai, vol. 6, pp. 9669–9682, 2022.

[2] M. Z. Asiari, "Analisis Kinerja Sistem Auto Scaling Pada Sistem Web Server Berbasis Clustering Menggunakan Sistem Virtual," Univ. Hasanuddin, pp. 3–35, 2021.

[3] K. A. Prasetyo, M. Idhom, and H. E. Wahanani, "Sistem Pencegahan Serangan Bruteforce Pada Multiple Server Dengan Menggunakan Fail2Ban," J. Inform. Dan Sist. Inf., vol. 1, no. 3, pp. 789–796, 2020, [Online]. Available: http://jifosi.upnjatim.ac.id/index.php/jifosi/article/view/208

[4] R. A. Aprilliandi and R. Efendi, "Perancangan Dan Implementasi Load Balancing Web Server Menggunakan Haproxy (High Availability Proxy) Studi Kasus di SMK Telekomunikasi Tunas Harapan Kab. Semarang," Fak. Teknol. Inf. Univ. Kristen Satya Wacana, no. 672015222, 2019.

[5] Dicky Setiawan, "MEMBANGUN SISTEM MONITORING MALICIOUS MENGGUNAKAN MALTRAIL DAN FAIl2BAN PADA JARINGAN SERVER DISKOMINFO SUMEDANG," Univ. Sebel. April, no. 8.5.2017, pp. 2003–2005, 2022.

[6] A. D. Batistuta, A. H. Hendrawan, and Ritzkal, "Analisis Keamanan Jaringan Server Terhadap Serangan Dictionary Menggunakan Tools Fail2Ban Dengan Notifikasi Telegram," INFOTECH J., vol. 10, no. 1, pp. 64–73, 2024, doi: 10.31949/infotech.v10i1.8730.

[7] H. Setiawan et al., "Penggunaan Metode Signatured Based Use of Signature Based Method," J. Teknol. Inf. dan Ilmu Komput., vol. 8, no. 3, pp. 517–524, 2021, doi: 10.25126/jtiik.202184200.

[8] M. Syafrizal, "Pengantar Jaringan Komputer," Andi Publisher. Accessed: Aug. 10, 2024. [Online]. Available: https://books.google.co.id/books?hl=id&lr=&id=UKNyejI7H0IC&oi=fnd&pg=PA1

&dq=jaringan+komputer&ots=qIcQWnF42h&sig=Fzx6U1twjXsnQjy-bFGexW0JpWs&redir_esc=y#v=onepage&q=jaringan komputer&f=false

[9] C. Rizal, S. Supiyandi, M. Zen, and M. Eka, "Perancangan Server Kantor Desa Tomuan Holbung Berbasis Client Server," Bull. Inf. Technol., vol. 3, no. 1, pp. 27–33, 2022, doi: 10.47065/bit.v3i1.255.

[10] A. R. Sofyan, "Implementasi Load Balancing Web Server menggunakan Haproxy pada Virtual Server Direktorat SMK Kemendikbudristek," J. Pendidik. Tambusai, vol. 6, pp. 9669–9682, 2022, [Online]. Available: https://jptam.org/index.php/jptam/article/view/3954%0Ahttps://jptam.org/index.php/jptam/article/download/3954/3294

[11] R. Riska and H. Alamsyah, "Analisa Dan Perancangan Load Balancing Web Server Mengunakan HAProxy," Techno.Com, vol. 20, no. 4, pp. 552–565, 2021, doi: 10.33633/tc.v20i4.5225.

[12] H. Setiawan, "Instalasi Serta Konfigurasi HAproxy Sebagai Load Balancing Web Server," NetPLG J. Netw. Comput. Appl., vol. 2, no. 1, pp. 23–35, 2023, [Online]. Available: https://jurnal.netplg.com/

[13] R. MOH ZUHDHI MALIK, "OPTIMASI APACHE WEB SERVER SISTEM INFORMASI SASARAN KINERJA PEGAWAI PEMERINTAH KABUPATEN KULON PROGO BERBASIS WEB MENGGUNAKAN VARNISH WEB CACHE," UNIVERSITAS TEKNOLOGI DIGITAL INDONESIA YOGYAKARTA, 2022.

[14] M. N. A. Rizqi and I. K. Dwi Nuryana, "Analisis Perbandingan Kinerja Algoritma Weighted Round Robin dan Weighted Least Connection Menggunakan Load Balancing Nginx Pada Virtual Private Server(VPS)," J. Informatics Comput. Sci., vol. 4, no. 01, pp. 67–75, 2022, doi: 10.26740/jinacs.v4n01.p67-75.

[15] A. WICAKSONO, "PERANCANGAN DAN IMPLEMENTASI IDS SURICATA, SNORT, DAN FAIL2BAN PADA RASPBERRY PI," SEKOLAH TINGGI TEKNOLOGI TERPADU NURUL FIKRI, 2022.

[16] F. Ainuddin, "Implementasi Performance Testing Pada Website E-Logistik Dengan Menggunakan Apache Jmeter," Politek. Negeri Lampung, 2023.