Jurnal RAMATEKNO, Vol. 6 No. 1, April 2026, hal. 1 – 14 ISSN 2828-2108
Makalah dikirim 25 Februari 2026; Revisi 2 Maret 2026; Diterima 4 Maret 2026

IMPLEMENTASI PRIVATE CLOUD IaaS UNTUK WEB SERVER DENGAN PENGUATAN KEAMANAN JARINGAN MENGGUNAKAN FIREWALL IPTABLES

[1]Halimil Fathi, [2]Musawarman,[3]Ricak Agus Setiawan,[4]Heti Mulyani, [5]Sukrina Herman, [6]Adit hidayat
[1]Teknologi Rekyasa Perangkat Lunak, Politeknik Enjinering Indorama
[1]e-mail: halimil.fathi@pei.ac.id

Abstrak

*Penerapan layanan web server berbasis cloud semakin meningkat, namun pengelolaan infrastruktur dan keamanan jaringan masih menjadi tantangan, khususnya pada lingkungan private cloud skala institusi yang memiliki keterbatasan sumber daya. Penelitian ini bertujuan membangun infrastruktur web server berbasis private cloud menggunakan model Infrastructure as a Service (IaaS) serta meningkatkan keamanan jaringan melalui penerapan firewall iptables. Metode Network Development Life Cycle (NDLC) digunakan untuk mendukung tahapan analisis, perancangan, implementasi, hingga pengujian sistem secara sistematis. Implementasi dilakukan dengan memanfaatkan virtualisasi server, aaPanel sebagai panel manajemen, serta konfigurasi aturan firewall untuk membatasi koneksi tidak sah dan memitigasi serangan jaringan seperti Distributed Denial of Service (DDoS). Pengujian keamanan difokuskan pada simulasi serangan DDoS menggunakan metode HTTP Flood. Hasil pengujian secara kuantitatif menunjukkan bahwa penerapan aturan iptables terbukti efektif menekan dampak serangan, ditandai dengan penurunan beban vCPU server dari kondisi kritis (**100%**) menjadi stabil di angka **4,3%**, menjaga penggunaan RAM di kisaran **1,03 GB** , serta mempertahankan response time layanan tetap normal di kisaran **120 ms** dengan menolak lalu lintas anomali (packet drop rate mencapai **95,8%**). Penelitian ini memberikan kontribusi berupa model implementasi private cloud web server dengan mekanisme pengamanan jaringan yang andal, efisien, dan terukur untuk diterapkan pada lingkungan pendidikan.*

Kata kunci: *Private Cloud, Infrastructure as a Service (Iaas), Web server, Iptables Firewall, Network Security*

Abstract

*The adoption of cloud-based web server services continues to grow; however, infrastructure management and network security remain significant challenges, particularly in institutional private cloud environments with limited resources. This study aims to develop a private cloud-based web server infrastructure using the Infrastructure as a Service (IaaS) model while enhancing network security through the implementation of an iptables firewall. The Network Development Life Cycle (NDLC) method was employed to systematically support the analysis, design, implementation, and testing phases. Implementation was carried out utilizing server virtualization, aaPanel as the management control panel, and firewall rule configurations to restrict unauthorized connections and mitigate network attacks such as Distributed Denial of Service (DDoS). Security testing focused on simulating DDoS attacks using the HTTP Flood method. Quantitative results indicate that the application of iptables rules effectively mitigated attack impacts, as evidenced by the reduction of the server's vCPU load from a critical state (**100%**) to a stable **4.3%**. Furthermore, RAM usage remained stable at **1.03 GB**, and service response time was maintained at a normal range of **120 ms** by rejecting anomalous traffic, achieving a packet drop rate of **95.8%**. This research contributes an applicable model for implementing secure private cloud web server infrastructure that is reliable, efficient, and measurable for educational or laboratory environments.*

Keywords*: Private Cloud, Infrastructure as a Service (Iaas), Web server, Iptables Firewall, Network Security*

1. PENDAHULUAN

Perkembangan teknologi informasi yang sangat pesat telah menjadikan internet sebagai sarana utama dalam memperoleh dan mendistribusikan informasi secara cepat dan efisien. Pemanfaatan teknologi informasi juga terus berkembang pada berbagai sektor, termasuk bidang pendidikan, sehingga menjadi salah satu aspek penting dalam menunjang aktivitas akademik maupun operasional institusi[1].

Seiring meningkatnya penggunaan komputer dan jaringan internet, kebutuhan pengelolaan data dan layanan digital menjadi semakin kompleks. Data dan layanan harus dapat diakses secara cepat, aman, serta mampu melayani pengguna dalam jumlah besar melalui jaringan telekomunikasi yang handal[2]. Salah satu teknologi yang banyak digunakan untuk memenuhi kebutuhan tersebut adalah *Cloud Computing*, yang memungkinkan penyediaan sumber daya komputasi secara fleksibel melalui jaringan internet sesuai kebutuhan pengguna[3].

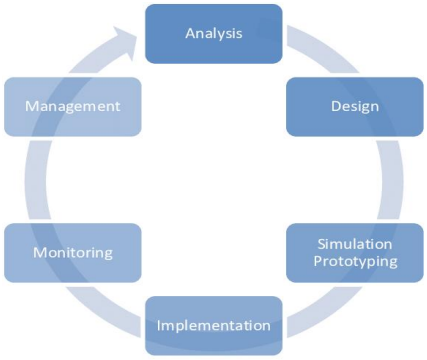
Salah satu model layanan dalam *Cloud Computing* adalah *Infrastructure as a Service (IaaS)*[4], yang memberikan keleluasaan bagi pengguna dalam mengelola infrastruktur server secara mandiri, termasuk pengelolaan *web server* sebagai media penyedia layanan aplikasi berbasis web[5]. Namun demikian, pengelolaan *web server* berbasis *cloud* masih menghadapi tantangan pada aspek keamanan jaringan[6], terutama ketika menggunakan panel manajemen server berbasis web seperti aaPanel. Ancaman keamanan seperti serangan *Distributed Denial of Service* (DDoS) berpotensi mengganggu stabilitas layanan dan membahayakan data pengguna[7].

Oleh karena itu, diperlukan mekanisme keamanan tambahan untuk melindungi web *server* berbasis *cloud*[8]. Salah satu solusi yang umum digunakan adalah *iptables*, yaitu *firewall* berbasis Linux yang berfungsi sebagai penyaring paket data dan pengendali lalu lintas jaringan berdasarkan aturan tertentu[9]. Penerapan *iptables* pada *server Linux*, khususnya Ubuntu Server, dapat membantu membatasi akses tidak sah, mengurangi potensi serangan *DDoS*, serta meningkatkan keamanan dan stabilitas layanan yang dikelola melalui aaPanel[7].

Berbeda dengan penelitian sebelumnya yang umumnya hanya membahas implementasi web server atau penerapan *firewall* secara terpisah, penelitian ini mengintegrasikan pembangunan infrastruktur *private cloud* berbasis *Infrastructure as a Service* (IaaS), manajemen layanan server menggunakan aaPanel, serta mekanisme pengamanan jaringan melalui konfigurasi *firewall iptables* dalam satu lingkungan implementasi yang terpadu. Selain itu, penelitian ini juga melakukan pengujian keamanan melalui simulasi serangan jaringan untuk mengevaluasi efektivitas perlindungan sistem terhadap stabilitas layanan server. Kontribusi penelitian ini berupa model implementasi web server berbasis *private cloud yang* tidak hanya mudah dikelola tetapi juga memiliki mekanisme pengamanan jaringan yang dapat diterapkan pada lingkungan laboratorium maupun institusi pendidikan dengan keterbatasan sumber daya infrastruktur.

2. METODE PENELITIAN

Penelitian ini menggunakan metode Network Development Life Cycle (NDLC) sebagai pendekatan dalam perancangan, implementasi, dan pengujian sistem *web server* berbasi *cloud*. Metode Metode NDLC dipilih karena lebih sesuai untuk infrastruktur *private cloud* berbasis teknologi *open-source* (Ubuntu, aaPanel, *iptables*), berbeda dengan metode PPDIOO yang cenderung terikat pada ekosistem perangkat keras tertentu (seperti Cisco). Selain itu, dibandingkan SDLC (*Software Development Life Cycle*) yang murni berfokus pada sikllus pengembangan kode perangkat lunak, NDLC menyediakan kerangka kerja yang lebih spesifik dalam menangani arsitektur jaringan. Kehadiran fase *Monitoring* dan *Management* pada NDLC menjadikannya metode yang paling ideal untuk mengevaluasi efektivitas kebijakan keamanan *firewall* secara berkelanjutan pasca-implementasi. NDLC terdiri dari beberapa tahapan utama yaitu Analysis, Design, Simulation/Prototype, Implementation, Monitoring dan Management[10] terdapat pada gambar 1 berikut ini.



Gambar 1. *Network Development Life Cycle.*

2.1. ANALYSIS

Tahapan analysis bertujuan untuk mengidentifikasi kebutuhan sistem sebelum proses perancangan dan implementasi dilakukan. Pada tahap ini dilakukan pengumpulan dan analisis kebutuhan yang meliputi:

1. Kebutuhan Perangkat keras (*Hardware*)
   Meliputi pemilihan spesifikasi seperti prosesor, kapasistas memori, media penyimpananan dan sumber daya lain yang disesuaikan dengan kebutuhan layanan web server.
2. Kebutuhan Perangkat Lunak (*Software*)
   Mencakup pemilihan sistem operasi server (*ubuntu server*), Panel Manajemen server (aaPanel), layanan *web server* (Nginx, MySQL, PHP), serta perangkat keamanan jaringan berupa Firewall iptables.
3. Kebutuhan keamanan Jaringan
   Tahapan ini mencakup identifikasi potensi ancaman jaringan seperti brute force attack, port scanning, *Distributed Denial of Service* (DDoS) dan *Brute Force*, serta kebutuhan perancangan aturan firewall untuk mengurangi risiko serangan.

Hasil analis menjadi dasar dalam perancangan topologi jaringan dan konfigurasi keamanan pada tahap berikutnya.

2.2. DESGIN

Tahapan design berfokus pada perancangan arsitektur dan topologi jaringan yang akan digunakan dalam penelitian. Hasil desain meliputi.

a. Perancangan Topologi Jaringan
   Menunjukan hubungan antara server, *firewall*, perangkat jaringan, serta koneksi jaringan yang digunakan dalam implementasi sistem.
b. Perencanaan Kebutuhan Sistem
   Meliputi estimasi kebutuhan perangkat keras, perangkat lunak, kapasistas server, dan kebutuhan bandwidth sesuai dengan layanan yang akan dijalankan.

2.3. SIMULATION AND PROTOTYPE

Tahapan ini dilakukan untuk memastikan sistem dapat berjalan dengan baik sebelum diterapkan pada lingkungan implementasi sebenarnya. Simulasi dilakukan menggunakan *VMWare Workstation pro* sebagai lingkungan virtualisasi server. Kegiatan pada tahap ini meliputi:

1. instalasi sistem operasi Ubuntu server sebagai sistem utama server.
2. Instalasi dan kofigurasi aaPanel serta layanan web server berabasis *LNMP* (*Linux, Nginx, MySQL, PHP*)
3. Penerapan konfigurasi awal *firewall Iptables* berdasarkan kebutuhan keamanan yang telah dianalisis.
4. Sismulasi serangan jaringan menggunakan aplikasi **LOIC** (*Low Orbit Ion cannon*) dengan metode *HTTP Flood* untuk menguji efektivitas konfigurasi *Firewall* dalam mitigasi serangan *DDoS* pada Lingkungan jaringan lokal.

Hasil simulasi digunakan sebagai bahan evaluasi sebelum sistem diterapkan pada tahap implementasi.

2.4. IMPLEMENTATION

Tahapan implentasi dilakukan dengan menerapkan seluruh hasil perancangan dan simulasi ke dalam lingkungan sistem yang digunakan secara nyata. Kegiatan implementasi meliputi:

1. Instalasi dan konfigurasi server local serta integrasi layanan aaPanel dengan aturan *firewall Iptables* sesuai desain sistem.
2. Konfigurasi domain, database, dan pengujian akses layanan *web server* menggunakan *protocol HTTP/HTTPS* melalui jaringan local.
3. Pengujian keamanan firewall melalui simulasi serangan jaringan untuk memastikan aturan iptables bekerja secara efektif dalam membatasi akses tidak sah.

Tahapan ini menghasilkan sistem web server yang telah terkonfigurasi dengan mekanisme keamanan jaringan dan siap digunakan serta dipantau pada tahap selanjutnya.

2.5. MONITORING

Tahap monitoring dilakukan untuk memantau performa server setelah sistem diimplentasikan. Pemantauan dilakukan terhadap kestabilan layanan, penggunaan sumber daya server, serta aktivitas lalau lintas jaringan untuk memastikan sistem berjalan sesuai kebutuhan layanan.

2.6. MANAGEMENT

Tahapan management dilakukan untuk menjaga keberlangsungan operasional sistem melalui pengelolaan konfigurasi server, pembaruan sistem keamanan, serta perbaikan konfigurasi apabila ditemukan gangguan atau potensi kerentaan pada sistem.

3. HASIL DAN PEMBAHASAN

3.1. TAHAPAN ANALISIS

Pada tahapan ini bertujuan untuk menentukan spesifikasi sistem yang diperlukan dalam membangun *web server* berbasis *cloud* di Laboratorium Komputer Teknologi Rekayasa Perangkat Lunak, dengan manajemen layanan menggunakan aaPanel dengan sistem keamanan jaringan berbasis *firewall iptables*. Secara fungsional, sistem yang dirancang harus mampu menjaga performa layanan tetap stabil sekaligus memiliki pertahanan yang kuat dalam membatasi akses tidak sah melalui konfigurasi keamanan yang terukur.

Pemilihan iptables dalam penelitian ini didasarkan pada efisensi komputasi dan *control granular* di tingkat kernel. Dibandingkan dengan *UFW* yang sekerdar antarmuka (*frontend*), *iptables* memberikan fleksibilitas lebih untuk menyusun aturan *packet filtering* yang kompleks. Secara operasional, *iptables* lebih proaktif dalam memblokir *anomaly* secara *real-time*, berbeda dengan *Fail2Ban* yang bersifat reaktif dan memiliki jeda waktu (*latency*) karena bergantung pada pembacaan *log* sistem. Selain itu, jika dibandingkan *Intrusion Detection/Prevention System* (IDS/IPS) seperti *snort* atau *suricata* yang membebani *CPU* dan *RAM* akibat proses *Deep Packet Inspection*, *iptables* beroperasi jauh lebih ringan. Karakteristik ini menjadikannya garda pertahanan pertama (*first line of defense*) yang paling optimal untuk arsitektur *private cloud* berskala institusi yang memiliki keterbatasan sumber daya keras.

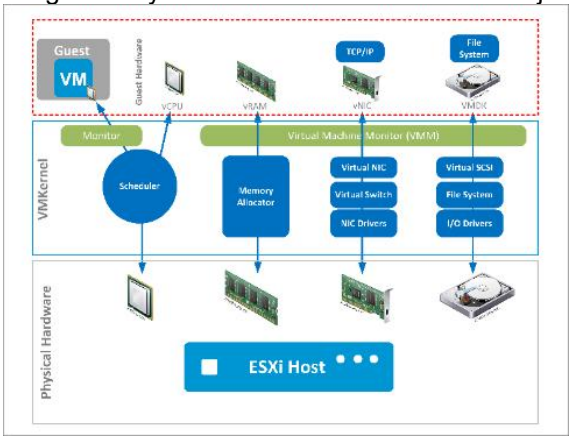
Analisis kebutuhan juga menunjukkan bahwa penggunaan *virtualisasi* server menjadi solusi yang efisien dalam menyediakan infrastruktur server tanpa memerlukan perangkat keras tambahan, sehingga cocok diterapkan pada lingkungan laboratorium dengan keterbatasan sumber daya.

3.2. ANALISIS ARSTITEKTUR JARINGAN CLOUD COMPUTING

Berdasarkan hasil analisis, dirancang arsitektur cloud computing yang digunakan untuk implementasi web server dengan memanfaatkan teknologi virtualisasi menggunakan VMware Workstation. Arsitektur ini memungkinkan server berjalan dalam lingkungan virtual sehingga memudahkan proses konfigurasi, pengujian, serta pengelolaan layanan server.

Komponen utama yang digunakan meliputi sistem operasi Ubuntu Server sebagai server utama, aaPanel sebagai panel manajemen layanan web server, serta firewall iptables yang berfungsi

mengontrol lalu lintas jaringan. Integrasi komponen tersebut memungkinkan sistem menjalankan layanan web secara stabil sekaligus menyediakan mekanisme keamanan jaringan.



Gambar 2. Architecture Virtual Machine Workstation Pro 16.

3.3. KEBUTUHAN ALAT DAN BAHAN

Implementasi sistem dilakukan menggunakan perangkat keras dan perangkat lunak yang disesuaikan dengan kebutuhan layanan server. Spesifikasi perangkat yang digunakan dinilai cukup untuk menjalankan layanan virtualisasi, web server, serta mekanisme pengamanan jaringan secara bersamaan tanpa mengalami penurunan performa yang signifikan.

Tabel 1. Kebutuhan Sistem, Perangkat Keras dan Lunak

No	Alat dan Bahan	Parameter dan Konfigurasi
1	Laptop Intel(R) Core(TM) i5-13420H | ram 16 GB | SSD 512GB | NVIDIA® GeForce® RTX 2025	Untuk Penginstalan *Software Vmware Workstation*
2	Sistem Operasi Windows 11 64bit	Sistem operasi sebagai Host
3	Software Vmware Workstation 17 pro	Download dan Install di *system* operasi windows 11 64bit
4	Software Ubuntu server 20.04 LTS	Untuk penginstalan Web Server LNMP
5	Software aapanel	Untuk pengelola Web server LNMP

3.4. TAHAPAN DESAIN

3.4.1. PENGALAMATAN IPADDRESS

Pada tahap desain jaringan dilakukan perancangan skema pengalamatan *IPAddress* untuk seluruh perangkat yang terlibat dalam sistem, baik pada sisi host maupun mesin virtual. Perancangan ini bertujuan memastikan setiap komponen jaringan dapat berkomunikasi dengan baik serta mendukung operasional layanan web server secara stabil.

Server utama menggunakan sistem operasi Ubuntu Server yang dijalankan pada lingkungan virtualisasi untuk mengelola layanan web melalui aaPanel serta mekanisme pengamanan jaringan menggunakan *firewall iptables*. Penggunaan alamat IP statis pada mesin virtual bertujuan menjaga konsistensi akses layanan sehingga server dapat diakses secara stabil tanpa mengalami perubahan alamat IP ketika sistem dijalankan kembali.

Skema pengalamatan IP yang digunakan dalam implementasi sistem ditunjukkan pada Tabel 2.

Tabel 2. Pengalamatan IPAddress

Komputer	Network Card	Ip address
Windows 11 64bit [Host]	Intel(R) Wi-Fi 6AX203	10.238.148.149/24
	VMware Network Adapter VMnet1	192.168.101.1/24
	VMware Network Adapter VMnet8	192.168.100.2/24
VMWare Workstation 17 Pro	VMnet0 [Brige]	10.238.148.168/24
	VMnet1 [Host-only]	192.168.101.5/24

	Vmnet8 [NAT]	192.168.100.3/24
Ubuntu Server 20.04 LTS	Ens33	192.168.100.10/24

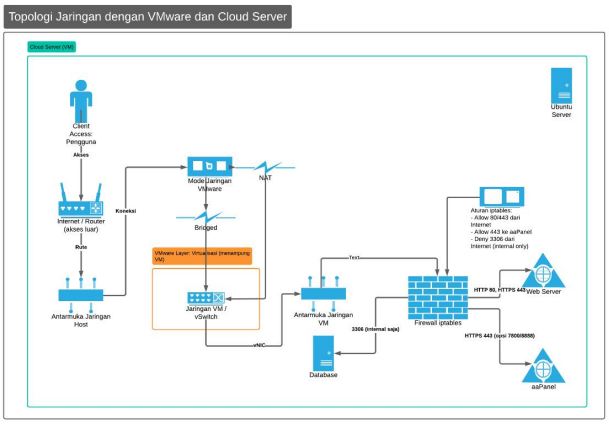
Berdasarkan konfigurasi jaringan yang tersedia, implementasi layanan server menggunakan adapter jaringan VMnet8 (NAT) karena mampu menyediakan akses jaringan internal sekaligus memungkinkan server tetap terhubung ke jaringan luar melalui host tanpa memerlukan konfigurasi perangkat jaringan tambahan.

Pemilihan mode NAT juga membantu meningkatkan keamanan server karena akses langsung dari jaringan luar dapat dibatasi melalui konfigurasi firewall yang diterapkan.

3.4.2. TOPOLOGI JARINGAN

Berdasarkan hasil desain, dibangun topologi jaringan cloud web server yang menghubungkan mesin host, server virtual, serta koneksi jaringan local yang digunakan dalam pengujian layanan. Topologi ini memungkinkan server berjalan dalam lingkungan virtualisasi namun tetap dapat diakses oleh pengguna pada jaringan local.

Pada Gambar 3 menunjukkan topologi jaringan yang digunakan dalam implementasi sistem cloud web server pada lingkungan Laboratorium Teknologi Rekayasa Perangkat Lunak. Topologi ini dirancang untuk mendukung pengelolaan layanan web secara terpusat sekaligus memudahkan proses pengujian keamanan jaringan.

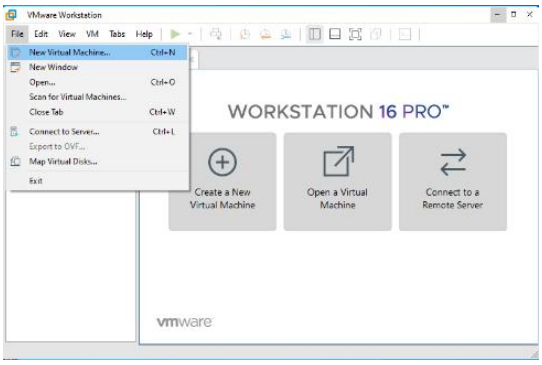


Gambar 3. Topologi Jaringan Cloud Webserver

4.1. TAHAPAN SIMULATION DAN PROTOTYPE

Tahapan simulasi dan pembuatan prototipe dilakukan untuk memastikan rancangan arsitektur jaringan dan konfigurasi server dapat berjalan dengan baik sebelum diterapkan pada lingkungan implementasi sebenarnya. Simulasi dilakukan menggunakan VMware Workstation sebagai lingkungan virtualisasi untuk menjalankan sistem operasi Ubuntu Server yang digunakan sebagai web server.

Pada tahap ini dilakukan pembuatan mesin virtual sebagai server utama dengan konfigurasi sumber daya yang disesuaikan dengan kebutuhan layanan, meliputi kapasitas penyimpanan dan alokasi memori agar server mampu menjalankan layanan web server dan panel manajemen secara stabil. Gambar 4 menunjukkan proses pembuatan mesin virtual yang digunakan sebagai server pada lingkungan simulasi.



Gambar 4. Membuat Virtual Machine

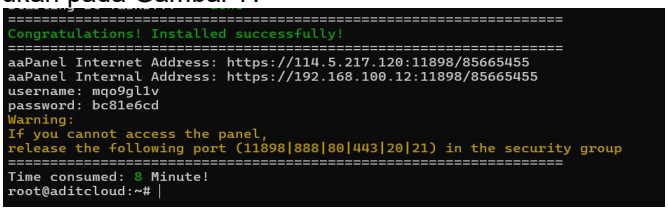
Konfigurasi mesin virtual diawali dengan penyesuaian kapasitas memori dan media penyimpanan guna menjaga stabilitas sistem tanpa mengganggu kinerja perangkat *host*. Setelah Ubuntu Server terinstal sebagai fondasi utama, dilakukan pengaturan jaringan virtual dan implementasi IP Statis untuk memastikan aksesibilitas layanan yang berkelanjutan. Optimasi performa dilakukan melalui manajemen partisi dan penyediaan ruang *swap* untuk mendukung pemrosesan data saat beban memori meningkat. Prosedur simulasi dinyatakan selesai setelah verifikasi profil dan otentikasi *login* berhasil dilakukan, yang menunjukkan kesiapan infrastruktur untuk tahap *deployment* layanan *web server*. Proses login sistem ditampilkan pada Gambar 6.

Gambar 6. Login Ubuntu server

4.2. IMPLEMENTASI

4.2.1. IMPLEMENTASI PANEL MANAJEMEN SERVER

Tahap implementasi diawali dengan instalasi panel manajemen server aaPanel pada sistem operasi ubuntu server 20.04 LTS. Instalasi dilakukan melalui terminal server menggunakan skrip instalasi otomatis yang disediakan oleh aaPanel untuk mempermudah proses konfigurasi awal layanan. Setelah proses instalasi selesai, sistem menampilkan informasi akses berupa Alamat IP Server, username, dan password awal yang digunakan untuk masuk ke panel melalui browser. Melalui antarmuka aaPanel, administrator dapat mengelola layanan web server, database, serta konfigurasi server secara lebih praktis dibandingkan konfigurasi manual command line. Proses instalasi aaPanel ditunjukan pada Gambar 7.

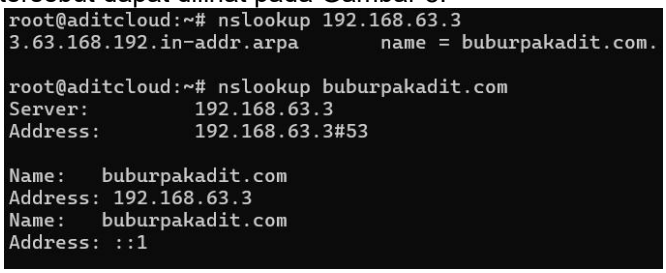


Gambar 7. Proses Instalasi aaPanel berhasil

4.2.2. IMPLEMENTASI LAYANAN DOMAIN NAME SYSTEM (DNS)

Layanan *Domain Name System* (DNS) diimplementasikan menggunakan perangkat lunak Bind9 untuk memfasilitasi akses layanan melalui nama domain. Tujuan utama dari konfigurasi ini adalah untuk menerjemahkan nama domain ke alamat IP server sehingga memudahkan pengguna dalam mengakses layanan *web*. Proses pengerjaan meliputi pembuatan zona domain serta

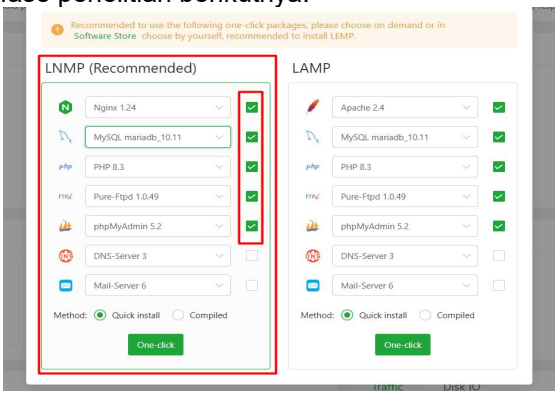
pengaturan file konfigurasi guna memastikan resolusi domain berjalan optimal. Keberhasilan instalasi diverifikasi melalui pengujian perintah *nslookup*, yang menunjukkan bahwa layanan DNS mampu merespons permintaan domain dengan benar sesuai konfigurasi yang diterapkan. Dokumentasi hasil pengecekan status DNS tersebut dapat dilihat pada Gambar 8.



Gambar 8. Hasil Pengecekan Nslookup status DNS

4.2.3. IMPLEMENTASI LAYANAN WEB SERVER LNMP

Tahapan selanjutnya dalam pembangunan infrastruktur adalah instalasi *stack* LNMP yang mengintegrasikan sistem operasi Linux, *web server* Nginx, manajemen basis data MySQL, dan bahasa pemrograman PHP. Proses implementasi ini dilakukan melalui antarmuka aaPanel dengan memanfaatkan fitur *Quick Install* untuk efisiensi waktu dan minimalisasi kesalahan konfigurasi manual. Seluruh komponen utama layanan dapat beroperasi dengan optimal setelah prosedur instalasi selesai dilaksanakan. Indikasi keberhasilan implementasi ditunjukkan oleh dasbor utama aaPanel yang menampilkan status aktif pada setiap layanan, sebagaimana terdokumentasi pada Gambar 9. Integrasi ini memastikan bahwa lingkungan server telah siap digunakan untuk tahap penyebaran aplikasi (*deployment*) pada fase penelitian berikutnya.

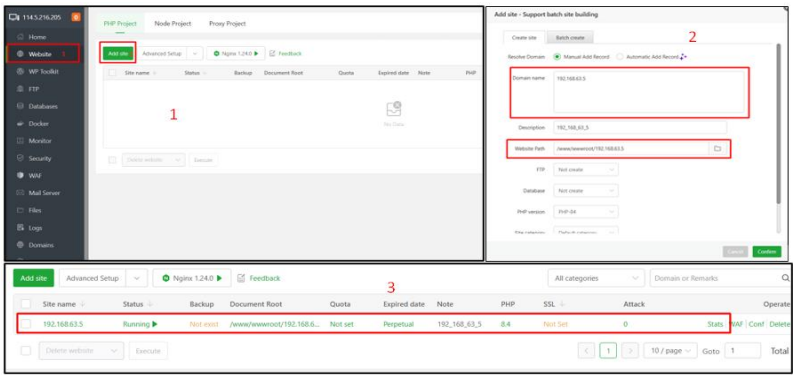


Gambar 9. Menu Quick Install LNMP

5.1. DEPLOYMENT SYSTEM

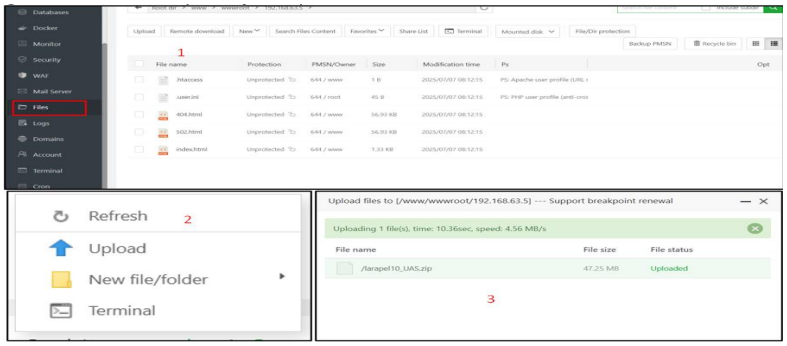
Fase *deployment* dilaksanakan setelah seluruh infrastruktur server dinyatakan operasional melalui panel manajemen aaPanel. Penggunaan aaPanel sebagai kontrol panel berbasis grafis berfungsi untuk mengefisiensikan pengelolaan layanan dalam ekosistem LNMP (*Linux, Nginx, MySQL, PHP*), sekaligus meminimalisasi risiko kesalahan konfigurasi manual yang umum terjadi pada antarmuka baris perintah (*CLI*).

Prosedur penyebaran diawali dengan konfigurasi entitas situs web pada panel manajemen, yang mencakup pengaturan domain, penentuan direktori penyimpanan, serta sinkronisasi versi PHP dan sertifikat SSL sesuai spesifikasi aplikasi sebagaimana ditunjukan pada Gambar 10.



Gambar 10. Proses Penambahan site

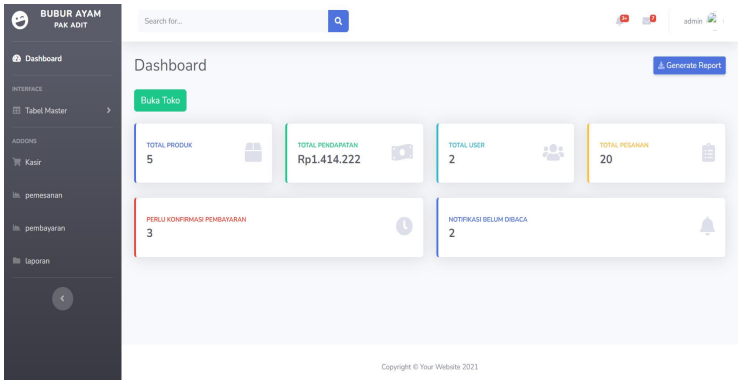
Selanjutnya, artefak aplikasi diunggah dan diekstraksi ke dalam direktori tujuan menggunakan fitur manajemen file terintegrasi. Tahapan ini memastikan seluruh dependensi file tersedia dan dapat dieksekusi oleh *web server* secara optimal, sebagaimana ditunjukkan pada Gambar 11. Integrasi langkah-langkah ini menandai kesiapan sistem untuk memasuki tahap pengujian validasi dan keamanan.



Gambar 11. Proses Deploy Sistem

Setelah ekstraksi file, dilakukan konfigurasi hak akses pada direktori strategis (*bootstrap, storage,* dan *public*) untuk mendukung fungsi *caching* serta *logging* aplikasi secara optimal. Tahapan dilanjutkan dengan pengaturan *document root* ke folder *public* dan aktivasi *URL rewrite* guna menjamin kelancaran mekanisme *routing*.

Proses diakhiri dengan migrasi basis data melalui phpMyAdmin agar seluruh struktur tabel dapat terintegrasi dengan sistem. Hasil implementasi menunjukkan bahwa aplikasi dapat diakses melalui domain yang ditetapkan tanpa kendala teknis, sebagaimana terdokumentasi pada Gambar 13.



Gambar 13. Halaman Utama Sistem setelah di Deploy

Secara keseluruhan, penggunaan aaPanel terbukti meningkatkan efisiensi pengelolaan dan mempermudah konfigurasi layanan dibandingkan metode manual, sehingga sistem dapat beroperasi secara stabil di lingkungan laboratorium.

.

6.1. Pengujian Keamanan server

Konfigurasi keamanan server dilakukan dengan mengimplementasikan aturan *firewall iptables* pada *INPUT chain* untuk memvalidasi seluruh lalu lintas jaringan yang masuk. Kebijakan ini mencakup manajemen koneksi yang telah terjalin (*established*), restriksi akses jarak jauh, serta pengaturan akses khusus untuk layanan aplikasi dan basis data guna memitigasi berbagai potensi serangan jaringan.

Selain itu, diterapkan mekanisme *rate limiting* serta kebijakan *default* **DROP** untuk menjamin bahwa hanya lalu lintas data yang sah yang diizinkan mengakses sistem. Melalui konfigurasi ini, seluruh aktivitas jaringan yang tidak sesuai dengan kebijakan keamanan akan dibatasi secara otomatis, sehingga integritas dan stabilitas layanan server tetap terjaga dengan optimal.

Tabel 3. Penerapan Aturan Firewall Iptables yang digunakan dalam implementasi sistem.

No	Perintah (syntax)	Keterangan / Fungsi
1	sudo iptables -A INPUT -m conntrack --ctstate RELATED, ESTABLISHED -j ACCEPT	Mengizinkan koneksi yang sudah terjalin (*ESTABLISHED*) atau berkaitan (*RELATED*) agar komunikasi aktif tidak terputus.
2	sudo iptables -A INPUT -p tcp -s 192.168.63.149 --dport 2323 -j ACCEPT	Memberikan izin akses SSH hanya untuk alamat IP tertentu (192.168.63.149) melalui port 2323.
3	sudo iptables -A INPUT -p tcp --dport 80 -j ACCEPT sudo iptables -A INPUT -p tcp --dport 81 -j ACCEPT sudo iptables -A INPUT -p tcp --dport 85 -j ACCEPT	Mengizinkan akses layanan HTTP pada port 80, 81, dan 85 agar aplikasi web dapat diakses.
4	sudo iptables -A INPUT -p tcp -s 192.168.63.0/24 --dport 3306 -j ACCEPT	Mengizinkan koneksi ke MySQL Database (port 3306) hanya dari jaringan lokal 192.168.63.0/24 untuk menjaga keamanan data.
5	sudo iptables -A INPUT -p tcp --dport 84 -m conntrack --ctstate NEW -j CONNECTION_LIMIT	Menerapkan pembatasan koneksi baru (*rate limiting*) pada port 84 untuk mencegah penyalahgunaan akses.
6	sudo iptables -A INPUT -p tcp --tcp-flags SYN SYN -m limit --limit 5/sec --limit-burst 10 -j ACCEPT sudo iptables -A INPUT -p tcp --tcp-flags SYN SYN -j DROP	Membatasi koneksi TCP SYN maksimum 5 per detik untuk mencegah serangan SYN Flood.
7	sudo iptables -A INPUT -p icmp --icmp-type 8 -m limit --limit 1/sec --limit-burst 3 -j ACCEPT sudo iptables -A INPUT -p icmp --icmp-type 8 -j DROP	Membatasi permintaan ping (ICMP) maksimal 1 per detik untuk mencegah ping flood atau serangan DDoS berbasis ICMP.
8	sudo iptables -A INPUT -p tcp --dport 53 -j ACCEPT sudo iptables -A INPUT -p udp --dport 53 -j ACCEPT	Mengizinkan lalu lintas DNS Server melalui port TCP dan UDP 53 agar sistem resolusi domain berjalan normal.
9	sudo iptables -N CONNECTION_LIMIT 2>/dev/null sudo iptables -A CONNECTION_LIMIT -m limit --limit 10/min --limit-burst 5 -j RETURN sudo iptables -A CONNECTION_LIMIT -j DROP	Membuat chain khusus *CONNECTION_LIMIT* untuk membatasi maksimal 10 koneksi per menit dan menolak koneksi berlebih.
10	sudo iptables -P INPUT DROP	Menetapkan kebijakan default DROP, menolak semua koneksi masuk yang tidak diizinkan secara eksplisit.

Penerapan aturan iptables tersebut mampu membatasi jumlah koneksi yang masuk serta memblokir lalu lintas jaringan yang mencurigakan, sehingga server menjadi lebih terlindungi dari

potensi serangan Distributed Denial of Service (DDoS) maupun akses ilegal. Konfigurasi ini meningkatkan ketahanan dan stabilitas server ketika menghadapi lalu lintas jaringan berlebih.
Gambar 14 menampilkan hasil implementasi aturan iptables pada server.

```
Chain INPUT (policy DROP 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
    0     0 DROP       0    --  *      *       0.0.0.0/0            0.0.0.0/0           match-set aapanel.ipv4.blacklist src
    0     0 ACCEPT     0    --  *      *       0.0.0.0/0            0.0.0.0/0           match-set aapanel.ipv4.whitelist src
 5040 2087K ACCEPT     0    --  *      *       0.0.0.0/0            0.0.0.0/0           ctstate RELATED,ESTABLISHED
    0     0 ACCEPT     6    --  *      *       10.238.148.149       0.0.0.0/0           tcp dpt:2323
    0     0 ACCEPT     6    --  *      *       0.0.0.0/0            0.0.0.0/0           tcp dpt:80
    0     0 ACCEPT     6    --  *      *       10.238.148.0/24      0.0.0.0/0           tcp dpt:3306
    0     0 CONNECTION_LIMIT  6    --  *      *       0.0.0.0/0            0.0.0.0/0           tcp dpt:84 ctstate NEW
    0     0 ACCEPT     6    --  *      *       0.0.0.0/0            0.0.0.0/0           tcp flags:0x02/0x02 limit: avg 5/sec bur
 10
    0     0 DROP       6    --  *      *       0.0.0.0/0            0.0.0.0/0           tcp flags:0x02/0x02
    0     0 ACCEPT     1    --  *      *       0.0.0.0/0            0.0.0.0/0           icmptype 8 limit: avg 1/sec burst 3
    0     0 DROP       1    --  *      *       0.0.0.0/0            0.0.0.0/0           icmptype 8
    0     0 ACCEPT     6    --  *      *       0.0.0.0/0            0.0.0.0/0           tcp dpt:53
    3   236 ACCEPT     17   --  *      *       0.0.0.0/0            0.0.0.0/0           udp dpt:53

Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination

Chain OUTPUT (policy ACCEPT 4983 packets, 397K bytes)
 pkts bytes target     prot opt in     out     source               destination

Chain CONNECTION_LIMIT (1 references)
 pkts bytes target     prot opt in     out     source               destination
    0     0 RETURN     0    --  *      *       0.0.0.0/0            0.0.0.0/0           limit: avg 10/min burst 3
    0     0 DROP       0    --  *      *       0.0.0.0/0            0.0.0.0/0
    0     0 RETURN     0    --  *      *       0.0.0.0/0            0.0.0.0/0           limit: avg 10/min burst 5
    0     0 DROP       0    --  *      *       0.0.0.0/0            0.0.0.0/0
```

Gambar 14. Hasil Implementasi Iptables

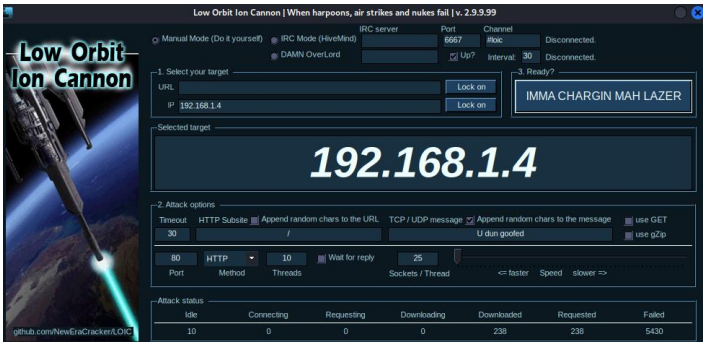
6.2. PENGUJIAN KEAMANAN SERVER

Pengujian keamanan server dilakukan untuk mengevaluasi tingkat ketahanan sistem terhadap serangan jaringan yang berpotensi mengganggu ketersediaan layanan web server. Pengujian difoskuskan pada simulasi Serangan Distributed Denial of Services (DDos) menggunakan metode HTTP Flood dengan memanfaatkan aplikasi Low Orbit Cannon (LOIC).

Serangan HTTP Flood dilakukan dengan mengirimkan sejumlah besar permintaan HTTP secara terus menerus ke web server dengan tujuan membebani sumber daya sistem sehingga layanan tidak dapat diakses normal. Pengujian dilakukan dengan dua scenario utma yaitu:

1. Server tanpa konfigurasi firewall iptables
2. Server dengan konfigurasi firewall iptables aktif

Parameter pengujian yang diamati meliputi utilisasi CPU, penggunaan memori, throughput jaringan, response time, packet drop rate, serta error rate layanan web server sebagaimana terlihat pada Gambar 15.



Gambar 15. Proses Simulasi serangan DDoS metode HTTP Flood menggunakan LOIC

6.2.1. HASIL PENGUJIAN SERANGAN HTTP FLOOD

Hasil pengujian menunjukan adanya perbedaan yang signifikan antara kondisi server tanpa perlindungan firewall dan kondisi server setelah penerapan aturan firewall iptables. Perbandingan hasil pengujian ditunjukan pada table 4.

Tabel 4. Perbadingan Performa Server Terhadap Serangan DDoS (HTTP Flood)

Parameter Pengujian	Kondisi Normal (Baseline)	Saat Serangan (Tanpa Iptables)	Saat Serangan (Dengan Iptables)
Utilitas CPU (vCPU)	**0,5**%	**100**% (Kritis)	**4,3**%
Penggunaan RAM	**1,03** GB	**1,64** GB (Kritis)	**1,03** GB
Throughput (*Request per sec*)	**50** req/sec	**8.500** req/sec (Anomali)	0 req/sec (Anomali di-*drop*)

.

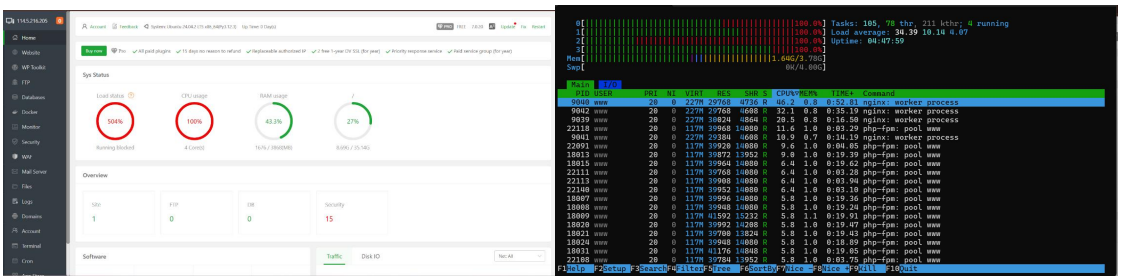
Response Time	**45** ms	> 10.000 ms (*Timeout*)	**120** ms
Packet Drop Rate (*Iptables*)	0%	0% (Paket masuk semua)	> 95.8% (Paket anomali ditolak)
Error Rate (HTTP 502/Timeout)	0%	**99**%	**0**%

6.2.2. ANALISIS SERANGAN TANPA IPTABLES

Berdasarkan hasil pengujian pada kondisi tanpa perlindungan firewall, server mengalami peningkatan beban sangat signifikan Ketika menerima serangan HTTP Flood. Utilisasi CPU meningkat drastic dari kondisi normal sebesar 0,5% menjadi 100%, yang menunjukan bahwa sumber daya pemrosesan server telah mencapai kondisi kritis.

Penggunaan memori juga meningkat dari 1,03 GB menjadi 1,64 GB yang menunjukan adanya peningkatan jumlah proses layanan web yang harus ditangani server akibat lonjakan permintaan HTTP secara bersamaan.

Troughput jaringan meningkat tajam hingga mencapai sekitar 8500 request perdetik, yang menyebabkan server tidak mampu memproses permintaan secara normal. Kondisi ini mengakibatkan waktu respon layanan meningkat hingga lebih dari 10.000 ms, sehingga Sebagian besar permintaan berakhir dengan timout. Selain itu, tingkat kesalahan layanan web (error rate) mencapai 99%, yang menunjukan bahwa hamper seluruh permintaan tidak berhasil diproses oleh server. Kondisi ini menunjukan bahwa server mengalami gangguan sercara serius akibat serangan HTTP Flood sebagaimana terlihat pada Gambar 16 dan Gambar 17.



Gambar 16. Kondisi Penggunaan CPU server saat serangan HTTP Flood tanpa Perlindungan Iptables.

Gambar 17. Hasil Serangan Tanpa Iptbles

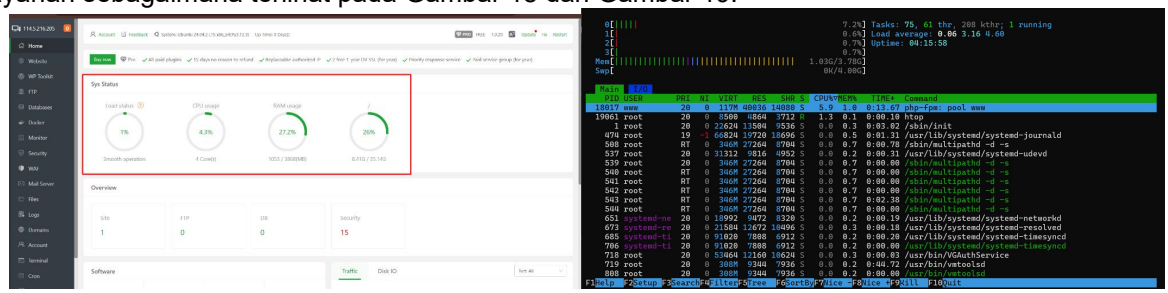
6.2.3. ANALISISS SERANGAN DENGAN IPTABLES

Setelah konfigurasi firewall iptables diterapkan, server menunjukkan peningkatan stabilitas yang signifikan meskipun melalui mekanisme rate limiting dan connection limiting, sehingga lalu lintas jairngan yang tidak normal dapat diblokir sebelum mencapai layanan web server.

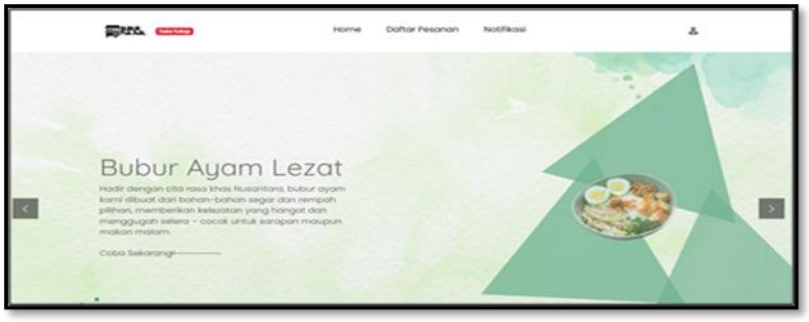
Utilitasi CPU berhasil ditekan hingga 4,3%, yang menunjukkan bahwa Sebagian besar paket serangan berhasil ditangani oleh firewall tanpa membenani proses layanan web server. Penggunaan memori tetap stabil pada kisaran 10,03 GB, yang menunjukan jumlah proses layanan web tetap terkendali.

Throughput jaringan yang masuk ke layanan web server berhasil ditekan hingga mendekati 0 request per detik, karena paket anomali telah diblokir oleh firewall. Hal ini menunjukkan bahwa iptables mampu bekerja sebagai lapisan perlindungan pertama dalam menyaring lalu lintas jaringan.

Firewall iptables berhasil menolak sekitar 95,8% paket serangan, yang menunjukkan efektivitas mekanisme filtering terhadap lalu lintas anomali. Dengan demikian, layanan web server tetap dapat diakses secara normal dengan waktu respon sekitar 120 ms dan tanpa terjadinya error layanan sebagaimana terlihat pada Gambar 18 dan Gambar 19.



Gambar 18. Kondisi Penggunaan CPU server saat serangan HTTP Flood dengan Perlindungan Iptables.



Gambar 19. Hasil Serangan dengan perlindungan iptables

6.2.4. EVALUASI EFEKTIVITAS IPTABLES

Berdasarkan hasil pengujian yang telah dilakukan, penerapan firewall iptables terbukti efektif dalam mengurangi dampak serangan DDoS berbasis HTTP Flood terhadap kestabilan server. Tanpa perlindungan firewall, server mengalami kondisi overload yang menyebabkan layanan tidak dapat diakses secara normal. Sebaliknya, setelah penerapan iptables, server mampu mempertahankan stabilitas layanan dengan penggunaan sumber daya yang relatif rendah.

Hasil pengujian menunjukkan bahwa firewall iptables mampu bekerja secara proaktif dengan melakukan penyaringan paket pada tingkat kernel sehingga beban pemrosesan aplikasi dapat diminimalkan. Pendekatan ini sangat sesuai untuk implementasi private cloud dengan keterbatasan sumber daya perangkat keras.

Secara keseluruhan, penerapan firewall iptables memberikan peningkatan ketahanan server terhadap serangan *Distributed Denial of Service* (DDoS) serta menjaga ketersediaan layanan web server tetap stabil.

7. KESIMPULAN

Berdasarkan hasil perancangan, implementasi, dan pengujian yang telah dilakukan, penelitian ini berhasil membangun infrastruktur web server berbasis private cloud Infrastructure as a Service (IaaS) menggunakan sistem operasi Ubuntu Server dengan panel manajemen server aaPanel serta mekanisme pengamanan jaringan menggunakan firewall iptables. Implementasi sistem menunjukkan bahwa penggunaan aaPanel mampu mempermudah proses instalasi, konfigurasi, dan pengelolaan layanan web server sehingga sistem dapat dioperasikan dengan lebih efisien dan stabil pada lingkungan laboratorium.

Hasil pengujian keamanan menggunakan simulasi serangan Distributed Denial of Service (DDoS) dengan metode HTTP Flood menunjukkan bahwa server tanpa perlindungan firewall mengalami penurunan performa yang signifikan. Utilisasi CPU meningkat dari kondisi normal sebesar 0.5% menjadi 100%, penggunaan memori meningkat dari 1.03 GB menjadi 1.64 GB, serta throughput jaringan meningkat hingga sekitar 8500 request per detik. Kondisi tersebut menyebabkan waktu

.

respon layanan meningkat hingga lebih dari 10.000 ms dan menghasilkan tingkat kesalahan layanan (error rate) mencapai 99%, sehingga layanan web server tidak dapat diakses secara normal.

Setelah penerapan firewall iptables, server menunjukkan peningkatan stabilitas yang signifikan meskipun serangan masih berlangsung. Utilisasi CPU berhasil ditekan menjadi sekitar 4.3%, penggunaan memori tetap stabil pada kisaran 1.03 GB, serta waktu respon layanan dapat dipertahankan pada kisaran 120 ms. Firewall iptables mampu menolak sekitar 95.8% lalu lintas anomali, sehingga permintaan tidak sah tidak membebani layanan web server dan error rate dapat ditekan hingga 0%.

Hasil penelitian menunjukkan bahwa penerapan firewall iptables mampu meningkatkan ketahanan dan stabilitas infrastruktur web server berbasis private cloud secara signifikan dengan penggunaan sumber daya yang relatif rendah. Mekanisme penyaringan paket pada tingkat kernel memungkinkan iptables bekerja secara efisien dalam membatasi koneksi tidak sah serta memitigasi serangan Distributed Denial of Service (DDoS).

Dengan demikian, model implementasi private cloud yang dikembangkan pada penelitian ini dapat menjadi solusi yang efektif dan efisien untuk meningkatkan keamanan dan keandalan layanan web server, khususnya pada lingkungan pendidikan atau laboratorium yang memiliki keterbatasan sumber daya infrastruktur.

DAFTAR PUSTAKA

[1] D. Anugrah, R. Ananda, and E. I. Wahyuni, “Evaluasi Keandalan Infrastruktur Cloud Computing Di Politeknik Belitung Untuk Mendukung Layanan Kampus Digital,” *J. Informatics Busisnes*, vol. 03, no. 01, p. 1, 2025.

[2] M. Arman and Meiriyama, “Rancang Bangun Web Server Blog Dengan Layanan Vps Dan Navigasi,” *Betrik*, vol. 16, no. 01, pp. 59–73, 2025, doi: 10.36050/w5xy6d50.

[3] Z. Mutaqin Subekti, D. Pranowo Kuswandono, A. Hafiz, H. Akmal, U. Bani Saleh, and I. Teknologi Bisnis dan Bahasa Dian Cipta Cendikia, “Rancang Bangun Cloud Storage Berbasis Docker Container,” *Jurnal*, vol. 01, no. 01, p. 2025, 2025, [Online]. Available: https://doi.org/10.9000/jupasti.v1i1.1

[4] H. Fathi, “PERANCANGAN CLOUD STORAGE PADA LABORATORIUM KOMPUTER TEKNOLOGI REKAYASA PERANGKAT LUNAK DENGAN MENGGUNAKAN FreeNAS,” *J. Inform. dan Tek. Elektro Terap.*, vol. 12, no. 3, 2024, doi: 10.23960/jitet.v12i3.4446.

[5] M. Ilham and K. C. Kaler, “SISTEM INFORMASI MAINTENANCE PERSONAL KOMPUTER BERBASIS WEB DI DPMPTSP KOTA,” vol. 13, no. 3.

[6] M. Y. Bagus, W. Di, and E. R. A. Industri, “Port Knocking Pada Mikrotik Untuk Meningkatkan Keamanan,” vol. 8, pp. 2115–2122, 2025.

[7] H. A. Damanik and M. Anggraeni, “Analisis dan Mitigasi Kerentanan DDoS pada Infrastuktur Jaringan dengan Teknik Hierarchical Clustering dan Firewall IPTables,” *J. Pekommas*, vol. 10, no. 1, pp. 29–38, 2025, doi: 10.56873/jpkm.v9i1.5551.

[8] D. W. Adi, Z. R. Athallah, F. Irawan, and S. N. Neyman, “Implementasi Firewall menggunakan Fitur dari IPTables pada Sistem Operasi Linux,” *J. Internet Softw. Eng.*, vol. 1, no. 2, p. 7, 2024, doi: 10.47134/pjise.v1i2.2671.

[9] B. R. Utomo, A. K. Jati, I. Ady, P. Informatika, and S. Indonesia, “Fail2Ban Terhadap Serangan Bruteforce,” no. November, pp. 1211–1223, 2024.

[10] J. Jamaluddin, Muhammad Zamroni Uska, R. H. Wirasasmita, and M. Roziki, “Development of Smart Servers for Informatics Education Program Using NDLC Method,” *J. Informatics Telecommun. Eng.*, vol. 7, no. 2, pp. 597–606, 2024, doi: 10.31289/jite.v7i2.10853.