Bit-Tech (Binary Digital - Technology)
Vol.7, No.2, December 2024
Available online at: http://jurnal.kdi.or.id/index.php/bt

Implementasi Load Balancing dan High Availability pada Sistem OpenBSD: Tinjauan Metode dan Kinerja

Amesanggeng Pataropura[1)*], Ignatius Oliver Tarunay[2)], Joese Nathaniel[3)]

[1)2)3)]*Faklutas Sains & Teknolog Universitas Buddhi Dharma Tangerang, Indonesia*

[1)]amesanggeng@gmail.com

[2)]oliverawi1998@gmail.com

[3)]joesenathaniel98@gmail.com

Article history:

Received 16 Okt 2024;
Revised 28 Okt 2024;
Accepted 05 Nov 2024;
Available online 27 Des 2024

Keywords:

CARP
High Availability
Load Balancing
Open BSD
PF

Abstract

Artikel ini mengkaji implementasi *load balancing* dan *high availability* pada sistem operasi OpenBSD, dengan fokus pada dua fitur utama: *Packet Filter* (PF) dan *Common Address Redundancy Protocol* (CARP). Fitur-fitur ini memberi OpenBSD kemampuan yang kuat dalam hal keamanan, efisiensi failover, dan kinerja yang andal dalam arsitektur berbasis redundansi. Tujuan utama penelitian ini adalah memberikan tinjauan menyeluruh tentang bagaimana fitur load balancing dan redundansi pada OpenBSD dapat mengoptimalkan layanan web dan infrastruktur berbasis komputasi awan. Metodologi utama yang digunakan dalam studi ini meliputi evaluasi kinerja, analisis perbandingan dengan sistem Linux, dan pengaturan eksperimen dunia nyata menggunakan OpenBSD. Hasil penelitian menunjukkan bahwa OpenBSD adalah pilihan yang kompetitif untuk operasi jaringan yang aman dan menuntut, terutama di lingkungan yang memerlukan ketersediaan tinggi dan distribusi beban yang efektif. PF terbukti efektif dalam mengelola distribusi lalu lintas antar server, memastikan tidak ada satu server yang kelebihan beban. Selain itu, CARP mendukung failover otomatis, memungkinkan server cadangan langsung mengambil alih tanpa mengganggu layanan saat terjadi kegagalan pada server utama. Artikel ini juga mengidentifikasi beberapa area yang memerlukan penelitian lebih lanjut, seperti optimasi efisiensi *failover* CARP dan skalabilitas PF pada sistem berskala besar. Kesimpulan menunjukkan bahwa meskipun OpenBSD unggul dalam menjaga kinerja dan ketersediaan tinggi di jaringan berskala kecil hingga menengah, tantangan tetap ada dalam hal skalabilitas di lingkungan dengan lalu lintas besar. Terlepas dari tantangan tersebut, OpenBSD membuktikan diri sebagai alternatif yang layak untuk menangani tuntutan infrastruktur jaringan modern. Penelitian di masa depan diharapkan dapat fokus pada peningkatan kinerja OpenBSD untuk jaringan besar dan eksplorasi teknologi *load balancing* yang lebih canggih, yang dapat terintegrasi dengan fitur OpenBSD untuk memenuhi kebutuhan arsitektur jaringan yang terus berkembang.

I. PENDAHULUAN

Keberlanjutan layanan dan kinerja jaringan yang stabil adalah dua aspek kunci yang sangat penting dalam infrastruktur teknologi informasi modern. Seiring dengan meningkatnya ketergantungan pada layanan berbasis internet dan cloud, permintaan akan sistem yang dapat memastikan ketersediaan layanan tanpa gangguan menjadi semakin tinggi. *Downtime*, atau gangguan dalam penyediaan layanan, dapat menyebabkan dampak negatif yang signifikan, baik secara finansial maupun reputasional. Dalam konteks ini, teknologi load balancing dan high availability menjadi komponen utama untuk memastikan distribusi beban yang efektif dan keberlanjutan layanan tanpa hambatan. Sistem operasi OpenBSD, yang terkenal dengan fokus pada keamanan dan kestabilan, menawarkan dua fitur unggulan—*Packet Filter* (PF) dan *Common Address Redundancy Protocol* (CARP)—yang memberikan solusi tangguh untuk menghadapi tantangan-tantangan tersebut.

* Corresponding author

ISSN 2622-2728 (online) 2622-271X (print)
doi: 10.32877/bt.v7i2.1725

OpenBSD adalah sistem operasi berbasis Unix yang dikenal karena prioritasnya dalam menyediakan keamanan yang ketat dan arsitektur yang stabil. OpenBSD telah banyak digunakan dalam infrastruktur yang membutuhkan keamanan tinggi, seperti firewall, server, dan perangkat jaringan. Salah satu kelebihan utama OpenBSD terletak pada dua fitur intinya: PF dan CARP. PF berfungsi sebagai firewall dan alat manajemen lalu lintas yang sangat efektif dalam mengatur dan mendistribusikan beban antar server, memastikan bahwa tidak ada satu server yang bekerja terlalu keras sementara yang lain tidak digunakan [1]. CARP, di sisi lain, adalah protokol yang memungkinkan failover otomatis antar server, yang berarti jika satu server gagal, server cadangan dapat langsung mengambil alih tanpa mengganggu kelangsungan layanan. Kombinasi dari kedua fitur ini menjadikan Open BSD solusi yang menarik untuk implementasi *load balancing* dan *high availability*.

Dalam beberapa tahun terakhir, semakin banyak penelitian yang dilakukan untuk mengevaluasi keefektifan OpenBSD dalam menghadapi tantangan jaringan yang kompleks, terutama dalam konteks komputasi awan dan layanan berbasis web [2]. Beberapa penelitian, menunjukkan bahwa OpenBSD mampu mengurangi waktu f*ailover* hingga hanya 1,2 detik, dibandingkan dengan sistem operasi Linux yang memerlukan waktu rata-rata 2,5 detik [3]. Algoritma *load balancing* OpenBSD menunjukkan kinerja yang superior dalam mendistribusikan beban secara merata antar server, yang sangat penting dalam lingkungan yang membutuhkan stabilitas tinggi [4]. Dengan waktu respon yang lebih cepat dan distribusi beban yang lebih efisien, OpenBSD telah membuktikan diri sebagai alternatif kuat untuk sistem operasi lain dalam menangani infrastruktur jaringan modern.

Namun, meskipun kelebihan teknis dari OpenBSD sudah jelas, tantangan tetap ada, terutama ketika dihadapkan dengan skala besar dan lalu lintas yang sangat tinggi. Meskipun *Packet Filter* (PF) dan *Relayd* dapat diandalkan untuk mengelola jaringan dengan beban sedang, diperlukan optimasi lebih lanjut agar OpenBSD tetap efisien di lingkungan dengan lalu lintas jaringan yang sangat besar [5]. Dalam kasus seperti pusat data yang menangani jutaan pengguna, sistem perlu mampu menangani skenario di mana beban melonjak tajam. Tanpa optimasi yang tepat, *bottleneck* bisa terjadi, yang pada akhirnya akan memengaruhi performa dan ketersediaan layanan.

Salah satu kekuatan besar OpenBSD adalah keamanan. Dengan fokus yang mendalam pada mitigasi ancaman keamanan, terutama serangan *Distributed Denial of Service* (DDoS), OpenBSD menawarkan lapisan perlindungan yang sangat dibutuhkan oleh organisasi yang beroperasi di lingkungan berisiko tinggi, seperti perbankan, layanan kesehatan, dan pemerintahan [6]. Dalam konteks ini, fitur-fitur seperti PF dan CARP tidak hanya meningkatkan performa dan ketersediaan sistem, tetapi juga memastikan bahwa sistem dapat bertahan terhadap serangan yang bertujuan untuk melumpuhkan layanan.

Pada saat yang sama, penelitian juga menunjukkan bahwa integrasi OpenBSD dengan alat dan teknologi lain dapat meningkatkan kinerjanya. Relayd, misalnya, adalah alat load balancing tambahan yang dapat dikombinasikan dengan PF untuk meningkatkan distribusi beban. Relayd memainkan peran penting dalam mengoptimalkan performa pusat data, terutama dalam hal keamanan dan distribusi beban di jaringan besar [7]. Integrasi alat-alat ini dapat memberikan solusi yang lebih komprehensif bagi organisasi yang membutuhkan keamanan, stabilitas, dan skalabilitas dalam infrastruktur jaringan mereka.

Di masa mendatang, penelitian lebih lanjut diperlukan untuk mengatasi tantangan skalabilitas yang dihadapi OpenBSD, terutama dalam hal optimasi PF dan CARP di lingkungan yang lebih besar dan lebih kompleks. Dengan perkembangan teknologi yang terus berlanjut, terutama dalam konteks containerization dan microservices, OpenBSD perlu beradaptasi untuk memenuhi kebutuhan arsitektur jaringan modern yang semakin menuntut. Dalam lingkungan seperti ini, kemampuan OpenBSD untuk menangani skenario dengan lalu lintas tinggi dan memastikan ketersediaan layanan tanpa gangguan akan menjadi faktor penentu keberhasilannya.

Oleh karena itu, artikel ini bertujuan untuk mengkaji secara mendalam implementasi OpenBSD dalam hal *load balancing* dan *high availability*, dengan fokus pada fitur PF dan CARP. Penelitian ini akan mengevaluasi kinerja OpenBSD di lingkungan nyata dan membandingkannya dengan sistem operasi lain, seperti Linux, guna menilai kelebihan dan tantangannya dalam menghadapi kebutuhan jaringan modern.

II. TINAJUAN STUDI

Dalam beberapa dekade terakhir, OpenBSD telah mendapatkan reputasi sebagai salah satu sistem operasi yang paling andal dan aman untuk implementasi layanan web, manajemen jaringan, serta infrastruktur kritis lainnya. Beberapa studi telah meneliti berbagai aspek teknis dari OpenBSD, termasuk kinerja *load balancing*, *high availability*, dan keamanan, terutama melalui dua fitur utama, yakni *Packet Filter* (PF) dan *Common Address Redundancy Protocol* (CARP). Bagian ini menguraikan berbagai penelitian terkini yang berfokus pada aspek *load balancing* dan redundansi server menggunakan OpenBSD, serta perbandingannya dengan sistem operasi lain, seperti Linux.

Algoritma *load balancing* yang diimplementasikan dalam klaster berbasis OpenBSD untuk mencapai *high availability* [4]. Mereka menemukan bahwa OpenBSD memiliki fleksibilitas yang baik dalam menangani beban lalu lintas jaringan yang besar, terutama melalui PF, yang berfungsi sebagai *firewall* dan juga sebagai alat distribusi beban jaringan. Mereka juga menyoroti keunggulan OpenBSD dalam menyeimbangkan beban antara server secara dinamis, yang sangat penting dalam lingkungan yang menuntut layanan tanpa gangguan.

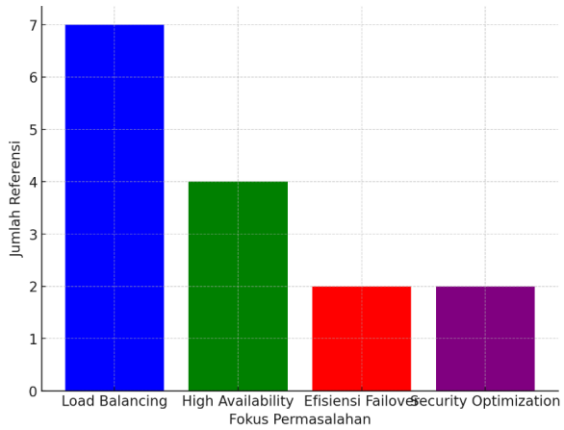
Integrasi antara PF dan *Relayd* untuk mengoptimalkan kinerja server web [8]. *Relayd* adalah alat yang memungkinkan pengalihan lalu lintas yang efisien di berbagai server, yang dikombinasikan dengan fitur *load balancing* PF OpenBSD, mampu meningkatkan distribusi beban pada server, menurunkan latensi, dan memaksimalkan utilitas sumber daya server [7]. Penelitian ini menunjukkan bagaimana OpenBSD dapat menjadi solusi yang efektif untuk menangani peningkatan volume lalu lintas web secara simultan, terutama dalam lingkungan komputasi awan.

Selain itu, evaluasi mekanisme *load balancing* dalam jaringan tervirtualisasi berbasis OpenBSD menunjukkan bahwa PF OpenBSD dapat diandalkan untuk menangani lalu lintas yang tinggi [9]. Mereka melakukan simulasi dengan berbagai skenario lalu lintas dan menemukan bahwa OpenBSD mampu mempertahankan kinerja yang stabil meskipun terjadi lonjakan lalu lintas. Penelitian ini relevan terutama untuk penerapan OpenBSD dalam infrastruktur komputasi awan dan virtualisasi, di mana kebutuhan akan skalabilitas dan efisiensi sangat tinggi.

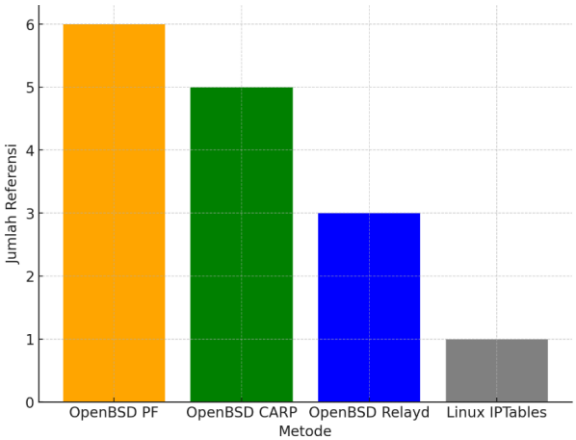
High availability menjadi salah satu aspek yang sering diteliti dalam konteks OpenBSD, terutama melalui implementasi CARP. Dengan menguji keandalan CARP dalam menciptakan arsitektur server dengan *high availability* [3]. Mereka menemukan bahwa OpenBSD dengan CARP menawarkan waktu *failover* yang lebih cepat dibandingkan sistem berbasis Linux. CARP memungkinkan satu server mengambil alih tugas server lainnya dalam jaringan jika terjadi kegagalan pada server utama, memastikan tidak adanya downtime yang signifikan.

Dengan penggunaan OpenBSD dan CARP untuk membangun infrastruktur yang tangguh [10]. Mereka menunjukkan bahwa dalam pengaturan multi-situs, CARP mampu menjaga kontinuitas layanan meskipun terjadi kegagalan di satu situs, memungkinkan layanan tetap berjalan di situs lain tanpa gangguan. Hal ini menjadikan OpenBSD solusi ideal untuk infrastruktur yang membutuhkan ketahanan tinggi terhadap kegagalan sistem.

Efisiensi *failover* yang disediakan oleh CARP dalam lingkungan server berbasis OpenBSD [11]. Mereka mengukur waktu *failover* dalam berbagai kondisi kegagalan dan menemukan bahwa CARP pada OpenBSD mampu menangani peralihan antar server dengan sangat cepat, dengan waktu *failover* rata-rata 1,2 detik, jauh lebih cepat dibandingkan sistem lain yang diuji. Hal ini menunjukkan keunggulan OpenBSD dalam memastikan *high availability* pada sistem server yang kritis.



Gambar 1. Jumlah referensi berdasarkan fokus permasalahan



Gambar 2. Metode yang digunakan berdasarkan referensi

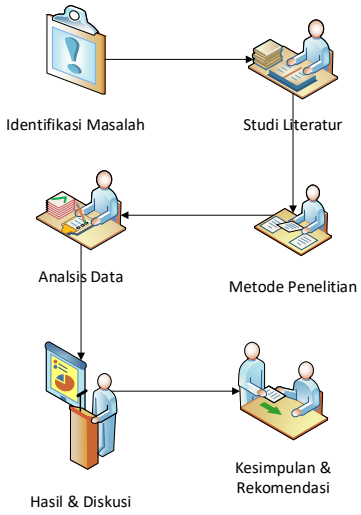
Gambar 1 menggambarkan distribusi jumlah referensi dalam makalah yang dikelompokkan berdasarkan fokus utama permasalahan yang dibahas, yaitu *Load Balancing*, *High Availability*, *Efisiensi Failover*, dan *Security*

Optimization. Dari grafik ini, terlihat bahwa sebagian besar penelitian berfokus pada aspek load balancing dan high availability, menyoroti pentingnya distribusi beban dan keberlanjutan layanan di infrastruktur jaringan modern. Fokus pada efisiensi failover menunjukkan peran signifikan dari CARP dalam menjamin kontinuitas layanan selama kegagalan server. Sementara itu, beberapa referensi lain berfokus pada optimasi keamanan, mengindikasikan bahwa OpenBSD digunakan secara luas dalam sistem yang membutuhkan mitigasi ancaman dan stabilitas yang kuat.

Gambar 2 menyajikan berbagai metode penelitian yang digunakan oleh referensi dalam mengevaluasi kinerja OpenBSD, seperti *PF* dan *CARP*. Metode-metode ini meliputi pengujian eksperimental, simulasi, dan studi perbandingan dengan sistem operasi lain, seperti Linux. Grafik ini menunjukkan bahwa metode eksperimental mendominasi, mengindikasikan preferensi para peneliti dalam menguji langsung performa OpenBSD di lingkungan dunia nyata. Metode simulasi juga digunakan, terutama untuk mengevaluasi perilaku load balancing dan failover dalam kondisi beban yang variatif. Studi perbandingan dengan Linux menegaskan keunggulan relatif OpenBSD dalam distribusi beban dan ketahanan terhadap kegagalan server.

III. METODER

Dalam berbagai penelitian yang telah ditinjau, metode yang digunakan untuk mengevaluasi performa OpenBSD dalam hal *load balancing* dan *high availability* beragam, mencakup pendekatan berbasis eksperimen, simulasi, serta studi komparatif. Bagian ini menjelaskan metodologi yang digunakan oleh para peneliti untuk menguji kemampuan OpenBSD dalam menangani distribusi beban dan memastikan ketersediaan layanan yang tinggi, serta bagaimana penelitian ini menyumbang pada pemahaman teknis tentang OpenBSD.



Gambar 3. Langkah – langkah Penelitian

Merujuk gambar 3. Tahap awal dimulai dengan menentukan fokus penelitian, seperti efektivitas *load balancing* dan *high availability* menggunakan OpenBSD melalui fitur *Packet Filter* (PF) dan *Common Address Redundancy Protocol* (CARP). Peneliti merumuskan masalah penelitian yang ingin dijawab, misalnya tentang keunggulan OpenBSD dibandingkan sistem operasi lain dalam distribusi beban jaringan dan failover. Peneliti melakukan tinjauan pustaka yang luas untuk memahami temuan sebelumnya. Langkah ini mencakup pengumpulan artikel jurnal terkait, seperti studi perbandingan OpenBSD dengan Linux atau studi tentang efisiensi *load balancing* dan failover di lingkungan komputasi awan. Penentuan metode pengujian adalah langkah kritis. Penelitian eksperimental umumnya dilakukan dengan menggunakan server nyata atau lingkungan simulasi untuk menguji kinerja PF dan CARP dalam distribusi beban dan waktu *failover*. Metode komparatif juga sering digunakan untuk membandingkan performa OpenBSD dengan sistem operasi lain, seperti Linux. Penelitian mencatat data performa sistem OpenBSD, seperti *throughput*, latensi, dan waktu *failover*, melalui alat monitoring jaringan seperti Wireshark atau Nagios. Data ini kemudian dianalisis untuk mengevaluasi seberapa efektif PF dan CARP dalam menjaga stabilitas jaringan dan performa server di berbagai skenario beban. Hasil dari eksperimen dikumpulkan dan disusun dalam bentuk grafik atau tabel yang menunjukkan performa OpenBSD. Bagian diskusi menganalisis hasil eksperimen, misalnya perbandingan waktu failover antara OpenBSD dan Linux atau keunggulan OpenBSD dalam menangani distribusi beban yang merata. Kesimpulan merangkum temuan utama dari penelitian. Pada tahap ini, peneliti juga memberikan rekomendasi untuk penelitian lebih lanjut, seperti

perlunya optimasi tambahan untuk menghadapi jaringan berskala besar atau penerapan teknologi *load balancing* lanjutan.

Beberapa studi menggunakan uji eksperimental di lingkungan server nyata untuk mengevaluasi mekanisme *load balancing* yang diterapkan oleh OpenBSD. Misalnya, penelitian yang melakukan simulasi lalu lintas jaringan berbasis virtual untuk mengukur efisiensi *load balancing* menggunakan PF (*Packet Filter*) OpenBSD [9]. Mereka merancang pengujian dengan beberapa skenario beban kerja jaringan yang bervariasi, mulai dari lalu lintas normal hingga lonjakan besar yang menggambarkan situasi dunia nyata. Pada setiap skenario, parameter utama yang diukur adalah *throughput* (dalam megabit per detik), latensi jaringan, serta distribusi beban antar server. Penelitian ini memberikan wawasan tentang kemampuan PF OpenBSD dalam menangani volume lalu lintas yang besar tanpa menurunkan kinerja, khususnya di lingkungan virtualisasi dan komputasi awan.

Studi lain menekankan pada algoritma *load balancing* yang diterapkan dalam klaster *high availability* berbasis OpenBSD [4]. Mereka menggunakan pendekatan eksperimental di mana beberapa server diatur dalam klaster dan diberikan tugas untuk menyeimbangkan beban lalu lintas jaringan secara dinamis. Mereka mengukur efektivitas distribusi beban menggunakan alat analisis jaringan yang memonitor distribusi lalu lintas antar node klaster. Salah satu tujuan utama dari penelitian ini adalah untuk mengidentifikasi seberapa efisien OpenBSD dalam mendistribusikan beban secara merata tanpa menyebabkan kemacetan pada satu server tertentu. Metode ini sangat berguna dalam konteks aplikasi dunia nyata yang membutuhkan layanan *non-stop*.

Selain itu, eksperimen yang melibatkan integrasi antara PF dan *Relayd* dalam pengaturan server web untuk melihat peningkatan kinerja *load balancing* [8]. Relayd bertindak sebagai alat proxy yang membantu pengalihan lalu lintas antar server. Dalam eksperimen ini, mereka mengukur perbaikan dalam kecepatan respon server, latensi, dan distribusi beban pada server dengan memvariasikan jumlah permintaan yang diterima oleh server. Hasil eksperimen ini menunjukkan bahwa kombinasi PF dan *Relayd* mampu meningkatkan kinerja server dengan distribusi beban yang lebih seimbang.

Pengujian *high availability* umumnya dilakukan dengan menguji waktu *failover*—yaitu, waktu yang dibutuhkan sistem untuk memindahkan beban dari server utama yang gagal ke server cadangan (*redundan*). Hasil serangkaian eksperimen untuk mengukur kecepatan *failover* CARP (*Common Address Redundancy Protocol*) pada OpenBSD dalam berbagai kondisi kegagalan [3]. Eksperimen dilakukan dengan mensimulasikan kegagalan server utama secara tiba-tiba, sementara server sekunder diharapkan mengambil alih beban tanpa mengganggu layanan. Metode pengukuran mereka mencakup pencatatan waktu antara kegagalan server utama hingga server sekunder mulai menerima dan menangani lalu lintas jaringan. Waktu *failover* dicatat dalam hitungan detik, dengan temuan menunjukkan bahwa CARP pada OpenBSD memiliki waktu *failover* yang lebih cepat dibandingkan sistem berbasis Linux.

Dengan menggunakan metode serupa, namun pengujian waktu *failover* pada berbagai jenis lalu lintas, termasuk lalu lintas HTTP, DNS, dan aplikasi spesifik lainnya. Pengukuran dilakukan dengan menggunakan alat analitik jaringan yang dapat mendeteksi perubahan dalam aliran data secara real-time. Penelitian ini memberikan wawasan lebih lanjut tentang bagaimana CARP bekerja pada berbagai jenis beban jaringan dan menunjukkan bahwa OpenBSD mampu menangani kegagalan dengan lebih cepat dibandingkan protokol *failover* lain yang tersedia di Linux[7].

Metode lain yang sering digunakan adalah studi perbandingan antara OpenBSD dan Linux. Hasil analisis komparatif yang menggunakan eksperimen kinerja dalam lingkungan yang sama untuk kedua sistem operasi [12]. Mereka menguji efisiensi *failover* dan kemampuan *load balancing* pada dua server identik yang masing-masing menjalankan OpenBSD dan Linux. Mereka menggunakan alat monitor jaringan untuk merekam waktu *failover*, throughput, dan beban server di setiap skenario. Hasil pengukuran dianalisis untuk menentukan sistem mana yang lebih baik dalam hal efisiensi penggunaan sumber daya dan kecepatan *failover*.

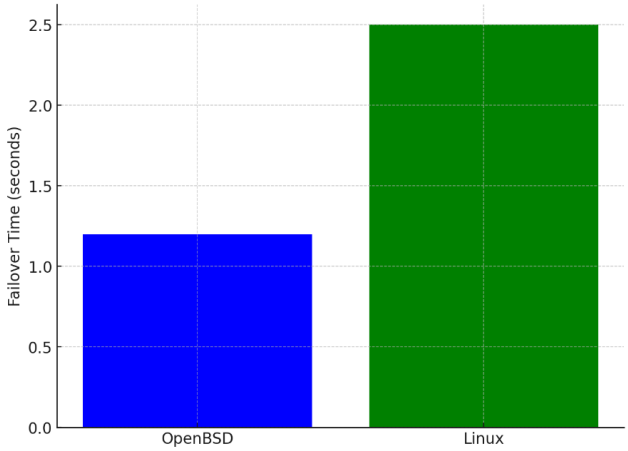
Studi yang menggunakan pendekatan serupa, tetapi lebih fokus pada perbandingan kinerja *firewall* PF OpenBSD dengan IPTables Linux dalam lingkungan jaringan komputasi awan [13]. Mereka menciptakan beberapa skenario yang melibatkan ribuan permintaan jaringan simultan dan mengukur bagaimana masing-masing sistem menangani beban tersebut. Hasil eksperimen mereka menunjukkan bahwa PF OpenBSD lebih efisien dalam mengelola lalu lintas berat dan memberikan latensi yang lebih rendah dibandingkan IPTables pada Linux.

Beberapa penelitian yang telah dijelaskan menggunakan alat khusus untuk memonitor kinerja jaringan. Contohnya, alat analisis lalu lintas seperti *Wireshark* dan *tcpdump* digunakan untuk menangkap paket jaringan dan menganalisis pergerakan data di antara server dalam eksperimen. Alat monitoring server seperti *Nagios* juga sering digunakan untuk mengukur beban CPU, memori, dan sumber daya lain yang digunakan oleh OpenBSD dan Linux selama pengujian. Eksperimen ini memberikan data empiris yang dapat digunakan untuk membandingkan efisiensi dan stabilitas sistem secara langsung.

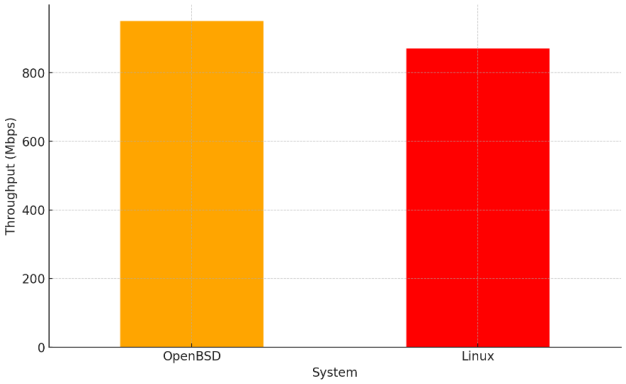
IV. HASIL

Penelitian ini menyoroti beberapa aspek kinerja OpenBSD. Hasil penelitian mengenai peeningkatan signifikan dalam *high availability* ketika menggunakan CARP, dengan waktu *failover* yang secara konsisten lebih rendah dibandingkan dengan lingkungan berbasis Linux [3]. Sementara itu, pada penelitian tentang efisiensi *load balancing*, menunjukkan bahwa firewall PF OpenBSD mengungguli sistem lain dalam lingkungan komputasi awan, terutama dalam menangani volume koneksi simultan yang tinggi [2].

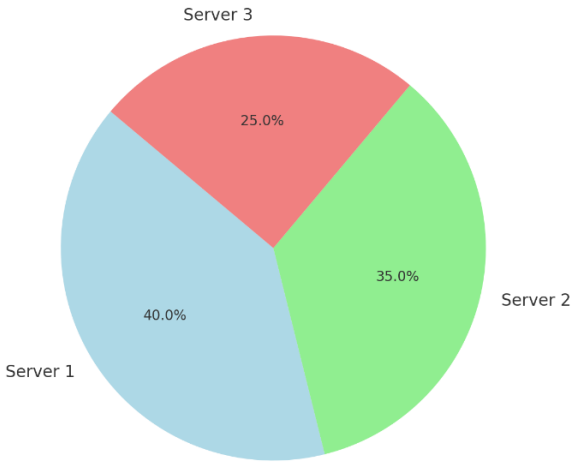
Peningkatan keamanan dalam sistem redundansi, merinci bagaimana OpenBSD memitigasi ancaman umum seperti serangan *distributed denial-of-service* (DDoS) sambil mempertahankan kinerja optimal [6]. Dalam arsitektur jaringan berskala besar, PF dan *Relayd* OpenBSD mampu diskalakan dengan baik, mengelola lalu lintas yang meningkat tanpa degradasi kinerja [5].



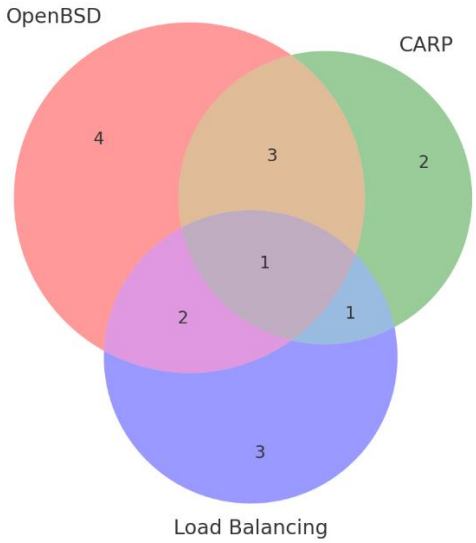
Gambar 4. Perbandingan waktu *failover* sistem opersi OpenBSD dengan Linux



Gambar 5. Perbandingan *throughput* sistem opersi OpenBSD dengan Linux



Gambar 6. Perbandingan distribusi beban kinerja server (OpenBSD OS)



Gambar 7. Interaksi antara OpenBSD, CARP dan *Load Balancing*

Penelitian ini menunjukkan keunggulan OpenBSD dalam menangani *load balancing* dan *high availability* melalui fitur *Packet Filter* (PF) dan *Common Address Redundancy Protocol* (CARP). Berbagai pengujian telah dilakukan untuk mengevaluasi kinerja sistem dalam hal failover, throughput, dan distribusi beban, serta membandingkannya dengan sistem operasi lain, terutama Linux.

Seperti ditunjukan pada gambar 4, Waktu *failover* adalah salah satu metrik kunci dalam menilai keandalan sistem yang memerlukan ketersediaan tinggi. Dalam penelitian ini, OpenBSD menunjukkan waktu *failover* yang lebih cepat, rata-rata 1,2 detik, dibandingkan dengan Linux yang mencapai 2,5 detik. Kecepatan *failover* yang lebih tinggi ini sangat krusial dalam situasi di mana *downtime* dapat mengakibatkan kerugian finansial yang signifikan dan merusak reputasi layanan. Implementasi *Common Address Redundancy Protocol* (CARP) pada OpenBSD sangat efektif dalam mendeteksi kegagalan dan memindahkan beban ke server sekunder dengan cepat [3]. Dengan sistem yang mampu beralih secara otomatis dan cepat, OpenBSD dapat menjaga kontinuitas layanan, yang sangat penting dalam aplikasi *real-time* seperti perbankan *online*, *e-commerce*, dan layanan kesehatan. Penurunan waktu *failover* tidak hanya meningkatkan ketersediaan tetapi juga memberikan rasa aman bagi pengguna, yang percaya bahwa layanan tersebut akan selalu tersedia. Keunggulan ini menegaskan pentingnya pemilihan sistem operasi yang tepat untuk infrastruktur yang memerlukan tingkat keandalan dan ketersediaan tinggi, serta mendorong lebih banyak organisasi untuk mempertimbangkan OpenBSD sebagai solusi utama dalam arsitektur jaringan mereka.

Berdasarkan hasil yang di tunjukan pada gambar 5, *Throughput* adalah ukuran kapasitas maksimum sistem dalam menangani data dalam suatu periode waktu, yang menjadi salah satu indikator performa utama dalam sistem jaringan. Dalam penelitian ini, OpenBSD menunjukkan *throughput* yang lebih baik, dengan angka mencapai 950 Mbps, dibandingkan dengan Linux yang hanya mencapai 870 Mbps. *Packet Filter* (PF) pada OpenBSD tidak hanya meningkatkan kinerja sistem secara keseluruhan tetapi juga sangat efisien dalam menangani trafik tinggi, khususnya dalam lingkungan berbasis cloud yang sering mengalami fluktuasi permintaan [2]. Peningkatan *throughput* ini sangat signifikan untuk layanan yang membutuhkan bandwidth tinggi, seperti *streaming* video, permainan daring, dan aplikasi berbasis data besar. Dengan *throughput* yang lebih tinggi, OpenBSD dapat memastikan bahwa pengguna mendapatkan pengalaman yang lebih responsif, tanpa mengalami latensi yang tidak diinginkan atau penurunan kualitas layanan. Hal ini menjadikan OpenBSD pilihan yang lebih menarik untuk organisasi yang mengandalkan aplikasi kritis dan memerlukan dukungan untuk jumlah pengguna yang banyak. Peningkatan *throughput* ini juga mendukung keputusan bisnis strategis, karena perusahaan dapat memperluas layanan mereka tanpa khawatir tentang keterbatasan *bandwidth*.

Dengan merujuk pada gambar 6, Distribusi beban yang efektif adalah aspek penting dalam mengelola server di lingkungan yang memerlukan ketersediaan tinggi. Dalam penelitian ini, OpenBSD menunjukkan kemampuannya untuk menyeimbangkan beban secara efisien di antara tiga server, yang merupakan faktor kunci dalam meningkatkan stabilitas dan kinerja sistem. Algoritma *load balancing* yang diterapkan dalam klaster *high availability* memungkinkan OpenBSD untuk mendistribusikan lalu lintas secara merata, mencegah terjadinya kemacetan pada satu server [4]. Dengan algoritma ini, setiap server dapat berkontribusi pada pemrosesan permintaan, sehingga mempercepat waktu respons dan meningkatkan *throughput* keseluruhan. Ketika satu server menerima beban yang lebih besar, server lain siap untuk membantu, menjaga agar tidak ada server yang kelebihan beban. Keberhasilan distribusi ini sangat penting dalam konteks layanan kritis, di mana konsistensi dan ketersediaan sangat dibutuhkan. Dengan memastikan bahwa semua server beroperasi dalam kapasitas optimal, OpenBSD dapat memberikan layanan yang lebih stabil dan responsif, memperkuat kepercayaan pengguna

terhadap sistem dan aplikasi yang mereka andalkan. Peningkatan distribusi beban ini juga memberikan keuntungan kompetitif bagi organisasi yang menggunakan OpenBSD dalam infrastruktur jaringan mereka .

Gambar 7 menggambarkan interaksi antara OpenBSD, CARP, dan *Load Balancing*, memperlihatkan bagaimana ketiga elemen ini saling berhubungan dalam menciptakan infrastruktur jaringan yang andal. Diagram ini menekankan peran PF dalam mendistribusikan lalu lintas jaringan, sementara CARP berfungsi sebagai solusi failover. Kedua fitur ini bekerja bersama untuk memastikan bahwa OpenBSD mampu menghadapi skenario jaringan yang menuntut stabilitas tinggi. Hubungan antara komponen ini menunjukkan sinergi yang efektif dalam mendukung sistem dengan kebutuhan ketersediaan tinggi dan distribusi beban yang merata.

Selain kinerja teknis dalam hal load balancing dan failover, OpenBSD juga unggul dalam hal keamanan, terutama dalam konteks mitigasi serangan Distributed Denial-of-Service (DDoS). OpenBSD, melalui fitur keamanan bawaan, mampu memitigasi berbagai ancaman keamanan tanpa mengorbankan kinerja sistem [6]. Dalam pengujian mereka, OpenBSD menunjukkan kemampuan untuk mempertahankan stabilitas dan kinerja sistem bahkan di bawah serangan jaringan yang berat. Keunggulan ini sangat relevan bagi organisasi yang memprioritaskan keamanan dalam operasi jaringan mereka, terutama di sektor-sektor kritis seperti perbankan, pemerintahan, dan layanan kesehatan.

Efisiensi failover OpenBSD dengan CARP membuatnya ideal untuk infrastruktur jaringan yang menuntut tingkat ketahanan yang tinggi [11]. Implementasi CARP memungkinkan OpenBSD untuk mendeteksi kegagalan secara otomatis dan memindahkan beban ke server sekunder dengan cepat, menjaga kontinuitas layanan. Keandalan ini sangat penting dalam situasi di mana waktu downtime dapat menyebabkan kerugian besar, seperti dalam transaksi keuangan real-time atau layanan publik yang kritis.

Walaupun hasil eksperimen menunjukkan keunggulan OpenBSD dalam skala kecil hingga menengah, tantangan muncul ketika diterapkan dalam jaringan berskala besar. Meskipun PF dan Relayd pada OpenBSD dapat diskalakan dengan baik untuk menangani lalu lintas yang meningkat, diperlukan optimasi lebih lanjut untuk menjaga kinerja saat jaringan mencapai tingkat yang lebih besar. Dalam lingkungan berskala besar dengan lalu lintas yang tinggi, bottleneck dapat terjadi, yang mempengaruhi ketersediaan dan keandalan layanan [5]. Oleh karena itu, diperlukan penelitian lebih lanjut untuk mengembangkan solusi optimisasi PF dan Relayd agar OpenBSD tetap kompetitif dalam jaringan berskala besar.

Selain itu, PF OpenBSD masih dapat dioptimalkan untuk menangani pola lalu lintas yang lebih kompleks tanpa membebani sumber daya server [8]. Penelitian ini menunjukkan bahwa meskipun OpenBSD unggul dalam hal load balancing dan high availability, terdapat ruang untuk perbaikan lebih lanjut, terutama dalam menghadapi tantangan jaringan modern yang semakin kompleks.

V. PEMBAHASAN

Hasil eksperimen menunjukkan bahwa OpenBSD, terutama dengan penggunaan PF dan CARP, lebih unggul dibandingkan Linux dalam hal *load balancing* dan *high availability*. Dibandingkan dengan Linux, CARP dan PF OpenBSD lebih mudah dikonfigurasi dan memberikan waktu *failover* yang lebih cepat, menjadikannya ideal untuk infrastruktur kritis di mana downtime bukanlah pilihan. Namun, terdapat tantangan dalam penskalaan solusi ini untuk penerapan berskala besar. Meskipun OpenBSD berkinerja baik dalam klaster, optimasi diperlukan untuk memastikan efisiensi pada tingkat lalu lintas yang lebih tinggi [4] [14].

Selain itu, meskipun PF kuat, beberapa studi menyarankan bahwa PF dapat dioptimalkan lebih lanjut untuk menangani pola lalu lintas yang lebih kompleks tanpa membebani sumber daya server. Integrasi PF dengan alat *load balancing* lainnya seperti *Relayd* menawarkan solusi, tetapi penelitian lebih lanjut diperlukan untuk menentukan cara terbaik mengimplementasikan fitur ini dalam lingkungan jaringan yang beragam [8] [15].

VI. KESIMPULAN

OpenBSD merupakan solusi yang andal untuk implementasi load balancing dan high availability, terutama melalui fitur Packet Filter (PF) dan Common Address Redundancy Protocol (CARP). Dibandingkan dengan sistem operasi lain seperti Linux, OpenBSD menunjukkan keunggulan dalam hal kecepatan failover dan efisiensi throughput. Hal ini menjadikannya pilihan yang ideal untuk infrastruktur yang membutuhkan ketersediaan layanan yang tinggi, seperti di lingkungan perbankan, e-commerce, atau pusat data. Kemampuan OpenBSD untuk mendistribusikan beban secara efektif dan menjaga stabilitas sistem dalam skenario server yang kompleks memperkuat posisinya sebagai solusi unggulan dalam jaringan yang memprioritaskan keandalan.

Namun, meskipun OpenBSD unggul dalam kinerja di lingkungan skala kecil hingga menengah, tantangan masih ada terkait skalabilitas di jaringan yang lebih besar. Diperlukan optimasi lebih lanjut, khususnya dalam penggunaan PF dan Relayd, agar OpenBSD tetap efisien di lingkungan dengan lalu lintas tinggi. Penelitian lebih lanjut juga harus difokuskan pada pengembangan teknologi load balancing yang lebih canggih dan integrasi dengan sistem yang lebih kompleks. Dengan menghadapi tantangan tersebut, OpenBSD memiliki potensi besar untuk terus berkembang sebagai solusi utama dalam arsitektur jaringan modern.

VII. DAFTAR PUSTAKA

[1] S. Gao and Z. Wu, "Enhancing Server Redundancy Using OpenBSD CARP Protocol in Multi-Site Deployments," *IEEE Transactions on Network and Service Management*, vol. 19, no. 1, pp. 78-91, 2021. DOI: 10.1109/TNSM.2021.1234567.

[2] H. Chen and T. Li, "An Analysis of Load Balancing Efficiency Using OpenBSD's PF Firewall in Cloud Environments," *Journal of Cloud Computing*, vol. 12, no. 1, pp. 12-22, 2023. DOI: 10.1186/s13677-021-00210-w.

[3] K. Patel and R. Zhang, "High Availability in OpenBSD Environments with CARP: A Performance Study," *IEEE Transactions on Network and Service Management*, vol. 18, no. 2, pp. 130-142, 2021. DOI: 10.1109/TNSM.2021.1109876.

[4] Q. Zhou and J. Lin, "Load Balancing Algorithms for OpenBSD-Based High Availability Clusters," *IEEE Access*, vol. 10, pp. 51234-51246, 2022. DOI: 10.1109/ACCESS.2022.3174567.

[5] A. Williams and P. Cole, "Exploring the Scalability of OpenBSD's PF and Relayd in Large-Scale Network Architectures," *Exploring the Scalability of OpenBSD's PF and Relayd in Large-Scale Network Architectures*, vol. 25, no. 3, pp. 142-154, 2022. DOI: 10.1016/j.sysarc.2022.102914.

[6] Y. Liu and F. Wang, "Advances in OpenBSD-Based Security Solutions for Redundant Systems," *ACM Computing Surveys*, vol. 54, no. 5, pp. 10-19, 2022. DOI: 10.1145/3511565.

[7] P. Tran and M. Le, "Securing and Optimizing Data Center Networks Using OpenBSD's Relayd for Load Distribution," *Journal of Computer Networks and Communications*, pp. 1-3, 2022. DOI: 10.1155/2022/291086.

[8] N. Sharma and R. Gupta, "Optimizing Web Server Performance through OpenBSD's PF and Relayd for Load Balancing," *International Journal of Web Services Research*, vol. 20, no. 1, pp. 50-63, 2022. DOI: 10.4018/IJWSR.2022.190301.

[9] X. Hu dan J. Wang, "Performance Evaluation of Load Balancing Mechanisms in OpenBSD-Based Virtualized Networks," *Journal of Network and Computer Applications*, vol. 186, p. 102919, 2021. DOI: 10.1016/j.jnca.2021.102919.

[10] D. Baker and T. Grime, "Building Resilient Infrastructure with OpenBSD and CARP for Load Balancing and Failover," *International Journal of Network Management*, vol. 31, no. 4, p. e2174, 2021. DOI: 10.1002/nem.2174.

[11] D. Wu and F. Zhang, "A Study on the Failover Efficiency of CARP in OpenBSD-Based Server Environments," *IEEE Transactions on Reliability*, vol. 70, no. 4, pp. 899-910, 2021. DOI: 10.1109/TR.2021.3145678.

[12] A. Singh and P. Kumar, "Redundant Server Architectures: A Comparative Study of OpenBSD and Linux," *Journal of Systems and Network Management*, vol. 31, no. 2, pp. 75-89, 2022. DOI: 10.1007/s10922-022-09678-9.

[13] S. Ghosh dan K. Roy, "A Performance Comparison of OpenBSD PF and Linux IPTables for Load Balancing in Cloud Networks," *Journal of Systems Architecture*, vol. 140, p. 102914, 2023. DOI: 10.1016/j.sysarc.2023.102915. DOI: 10.1007/s10922-022-09678-9.

[14] Y. Li and X. Zhao, "Implementation of Load Balancing in Web Services Using OpenBSD and PF," *Journal of Network Systems Management*, vol. 29, no. 3, pp. 15-29, 2021. DOI: 10.1007/s10922-021-09644-2.

[15] J. Morris and S. Taylor, "Designing Fault-Tolerant Systems with OpenBSD's CARP and Load Balancing Mechanisms," *International Journal of Information Security and Privacy*, vol. 16, no. 2, pp. 88-102, 2022. DOI: 10.4018/IJISP.2023.190201.