Analisis Sistem Keamanan Pada *Software-Defined Network* Dengan *Hybrid Honeypot* Menggunakan *Quality Of Service*

Revanza Hafiz Erianto
Teknik Informatika
Informatics and Computer Science Group
Purwokerto, Indonesia
revanzahafiz@student.telkomuniversity.ac.id

Wahyu Adi Prabowo
Teknik Informatika
Informatics and Computer Science Group
Purwokerto, Indonesia
wahyup@telkomuniversity.ac.id

Trihastuti Yuniati
Teknik Informatika
Informatics and Computer Science Group
Purwokerto, Indonesia
trihastutiy@telkomuniversity.ac.id

***Abstrak* — Pada awal 2024, Indonesia memiliki 185,3 juta pengguna internet, namun juga menghadapi peningkatan ancaman siber yang signifikan. Pada 2023 tercatat 603,3 juta serangan siber, termasuk *DDoS* yang mengganggu domain Pemilu 2024 dan 11 juta upaya *SSH Brute Force* terhadap situs pemerintah. Untuk mengatasi ini, penelitian ini merancang sistem keamanan berbasis *Software-Defined Network (SDN)* dengan *Hybrid Honeypot* yang dilengkapi *Intrusion Detection System (IDS)*. Hasil penelitian menunjukkan serangan *DDoS (ICMP Flood, SYN Flood, HTTP Flood)* dan *SSH Brute Force* berhasil dialihkan ke *Honeypot* melalui *Flow Rules* pada *Floodlight Controller*, sehingga *server* tetap aman. Pengukuran kualitas layanan menunjukkan bahwa serangan *SYN Flood* meningkatkan *throughput* dari 172,5 *bit/s* menjadi 182,6 *bit/s* dengan *delay* menurun dari 13.219 *ms* menjadi 10.244 *ms*. Serangan *HTTP Flood* mengurangi *throughput* dari 545,5 *bit/s* menjadi 490,3 *bit/s* dan *delay* meningkat dari 153.945 *ms* menjadi 296.158 *ms*. Serangan *ICMP Flood* menunjukkan *throughput* stabil sekitar 8.000 *bit/s* dengan *packet loss* meningkat menjadi 3,05% dan *delay* berubah dari 60,06 *ms* menjadi 62,11 *ms*. Serangan *SSH Brute Force* meningkatkan *throughput* dari 3,24 *bit/s* menjadi 3,81 *bit/s* dan *delay* menurun dari 367 *ms* menjadi 272 *ms*.**

Kata kunci* — *Software-Defined Network, Intrusion Detection Systems, Hybrid Honeypot, Distributed Denial of Service, SSH BruteForce, Quality of Service

I. PENDAHULUAN

Indonesia memiliki 185,3 juta pengguna internet pada awal 2024 atau 66,5% dari total populasi yang menjadikan negara ini salah satu yang terbesar dalam penggunaan internet [1]. Namun, peningkatan ini juga disertai dengan ancaman siber yang semakin besar, termasuk serangan *DDoS* dan *SSH Brute Force*. Pada 2023, tercatat 603,3 juta serangan siber dan 11 juta upaya serangan *SSH Brute Force* terhadap situs pemerintah [2]. Serangan *DDoS* dapat menyebabkan kegagalan sistem, seperti yang terjadi pada Pemilu 2024, sementara *SSH Brute Force* berpotensi mengakses data sensitif atau merusak sistem [3].

Untuk mengatasi ancaman ini, penelitian ini merancang sistem keamanan berbasis *Software-Defined Network (SDN)* yang menggunakan *Hybrid Honeypot, Intrusion Detection System (IDS),* dan *Quality of Service (QoS)* untuk memantau performa jaringan. Penelitian ini bertujuan merancang sistem yang dapat mendeteksi serangan dan meningkatkan keamanan serta ketersediaan layanan jaringan. Dengan menggunakan platform *Floodlight, Open Virtual Switch, Snort,* dan perangkat lunak *Honeypot* seperti *Dionaea* dan *Cowrie*, penelitian ini juga akan mengukur kinerja sistem dalam menghadapi serangan *DDoS (HTTP Flood, ICMP Flood, SYN Flood)* dan *SSH BruteForce*.

II. KAJIAN TEORI

A. *Software-Defined Network*

Software-Defined Networking (SDN) adalah pendekatan dalam bidang jaringan komputer yang memisahkan antara sistem pengendali (*control plane*) dengan perangkat keras jaringan (*data plane*). Dalam arsitektur *SDN*, *control plane* bertanggung jawab untuk mengatur lalu lintas jaringan, sementara data plane bertugas meneruskan lalu lintas berdasarkan keputusan yang diambil oleh *control plane*. *SDN* memungkinkan pengelolaan jaringan yang lebih fleksibel dan terpusat, di mana satu entitas perangkat lunak dapat mengendalikan beberapa elemen jaringan seperti *router*, *switch*, dan *middlebox* melalui *API* yang telah ditetapkan, seperti *OpenFlow*. Keunggulan utama *SDN* adalah kemampuannya untuk diprogram secara dinamis, memungkinkan *administrator* jaringan untuk mengelola dan mengoptimalkan jaringan dengan lebih efisien [4].

B. *Floodlight*

Floodlight adalah salah satu perangkat lunak pengendali (*controller*) dalam arsitektur jaringan berbasis *SDN*. Sebagai kontroler *SDN*, *Floodlight* bertugas mengelola dan mengatur lalu lintas jaringan dengan memisahkan lapisan kontrol (*control plane*) dari lapisan penerusan (*data plane*). *Floodlight* menggunakan protokol *OpenFlow* sebagai standar dalam mengendalikan jaringan *SDN* [5]. Keunggulan *Floodlight* meliputi performa yang tinggi, kemampuan untuk mengendalikan jaringan *SDN* yang besar secara efisien, dan arsitektur yang mudah dipahami. *Floodlight* juga mendukung pengembangan aplikasi jaringan yang inovatif dan dapat disesuaikan dengan kebutuhan pengguna [6].

C. *Wireshark*

Wireshark adalah perangkat lunak *open-source* yang digunakan untuk analisis jaringan. *Wireshark* memungkinkan pengguna untuk menangkap dan memeriksa data yang bergerak melalui jaringan komputer secara *real-time*. Sebagai alat analisis protokol jaringan, *Wireshark* mendukung berbagai protokol seperti *TCP/IP, HTTP, FTP,* dan *DNS* [7]. *Wireshark* menyediakan fitur untuk menangkap data dari antarmuka jaringan, menyaring paket untuk menampilkan informasi spesifik, dan menyimpan hasil tangkapan untuk analisis lebih lanjut. *Wireshark* juga mendukung *scripting* dengan menggunakan bahasa pemrograman seperti *Lua*, yang memungkinkan otomatisasi tugas-tugas analisis jaringan. *Wireshark* banyak digunakan dalam penelitian akademis dan keamanan siber untuk mempelajari perilaku jaringan, menganalisis serangan siber, dan mengevaluasi kinerja protokol jaringan [8].

D. *Honeypot*

Honeypot adalah mekanisme keamanan yang dirancang khusus untuk menarik, menangkap, dan menganalisis serangan siber yang dilakukan oleh pihak yang bermaksud jahat. Konsep utama *honeypot* adalah untuk menarik perhatian penyerang dengan menampilkan sasaran yang tampak rentan atau menarik, sehingga memungkinkan pihak pengelola sistem untuk memperoleh pemahaman yang mendalam terhadap taktik, teknik, dan perangkat yang digunakan oleh penyerang. *Honeypot* dapat digunakan untuk memantau dan menganalisis kegiatan penyerang yang tertangkap di *honeypot*. *Honeypot* dibagi menjadi tiga jenis berdasarkan tingkat interaksinya: *low-interaction honeypot* dan *high-interaction honeypot*. *Low-interaction honeypot* mensimulasikan layanan atau aplikasi tertentu dengan tingkat fungsionalitas yang terbatas, sementara *high-interaction honeypot* memberikan lingkungan yang lebih realistis dengan emulasi penuh sistem operasi dan aplikasi yang berjalan [9].

E. *Cowrie*

Cowrie adalah *honeypot* interaktif yang dirancang untuk mensimulasikan layanan *SSH* dan *Telnet* guna memantau serta menganalisis serangan siber. *Cowrie* mencatat seluruh aktivitas penyerang, termasuk perintah yang diberikan, *file* yang diunggah atau diunduh, serta perubahan yang dilakukan pada sistem. Keunggulan utama *Cowrie* terletak pada kemampuannya untuk merekam secara rinci seluruh sesi interaksi penyerang. Data yang dikumpulkan dapat dianalisis untuk mengidentifikasi pola serangan dan mengembangkan strategi mitigasi yang lebih efektif. *Cowrie* juga mendukung integrasi dengan berbagai alat analisis dan pemantauan, seperti *Splunk, ELK stack*, dan sistem *SIEM*, memungkinkan analisis data secara *real-time* dan integrasi dengan sistem keamanan yang lebih luas [10].

F. *Dionaea*

Dionaea adalah *honeypot* yang dirancang untuk mendeteksi, menganalisis, serta mengatasi aktivitas serangan dengan metode menjebak penyerang ke dalam suatu area yang terlihat seperti target yang rentan. *Dionaea* mengemulasi berbagai layanan yang rawan diserang, seperti *HTTP, SMB, FTP,* dan *MySQL*, untuk menarik perhatian penyerang yang berusaha mengeksploitasi celah dalam layanan tersebut. Sebagai *low-interaction honeypot*, *Dionaea* berperan untuk meniru layanan legal, menyediakan koneksi yang tidak legal, serta mendokumentasikan seluruh kegiatan yang dilakukan oleh penyerang, termasuk eksploitasi atau pengunduhan *malware*. Seluruh interaksi yang terjadi direkam dalam *log* untuk analisis lebih lanjut, sehingga memungkinkan *Dionaea* untuk mengenali pola serangan, mengumpulkan sampel *malware*, serta membagikan data yang bermanfaat untuk memperkuat pertahanan sistem yang sesungguhnya [11].

G. *Hybrid Honeypot*

Hybrid honeypot adalah sistem keamanan yang mengkombinasikan beberapa jenis *honeypot*, baik *low-interaction* maupun *high-interaction*, untuk mendeteksi dan menganalisis serangan keamanan. *Hybrid honeypot* dapat digunakan untuk mengumpulkan informasi tentang serangan dan menarik perhatian peretas. *Hybrid honeypot* dapat mendeteksi dan menganalisis serangan seperti *DoS, DDoS, SQL Injection, phishing, man-in-the-middle, ransomware, spoofing,* dan *brute force*. *Hybrid honeypot* juga dapat digunakan untuk mengamankan sistem asli agar tidak diserang atau disusupi oleh penyerang. Jika terjadi serangan terhadap suatu sistem asli, serangan tersebut akan dialihkan ke *honeypot*, sehingga sistem yang asli tetap aman dan terhindar dari serangan [4].

H. *Open Virtual Switch*

Open Virtual Switch (OVS) adalah elemen penting dalam virtualisasi jaringan yang berfungsi untuk mengatur lalu lintas data antar mesin virtual pada host fisik. *OVS* dirancang dengan performa tinggi dan fleksibilitas, menggunakan standar *OpenFlow* yang mendukung berbagai fungsi jaringan lanjutan seperti manajemen *VLAN*, penyeimbangan beban, dan pengaturan kualitas layanan (*QoS*). Keunggulan utama *OVS* terletak pada kemampuannya mengelola aliran data secara efisien dalam jaringan virtual melalui penggunaan tabel aliran yang dapat diprogram, memungkinkan *administrator* untuk menentukan jalur, filter, atau modifikasi lalu lintas jaringan secara dinamis. *OVS* juga mendukung integrasi dengan berbagai alat manajemen dan analisis jaringan, seperti sistem manajemen informasi keamanan (*SIEM*) dan platform pemantauan jaringan lainnya, yang meningkatkan visibilitas dan keamanan jaringan secara keseluruhan [12].

I. *Openflow*

OpenFlow adalah protokol yang terkait erat dengan arsitektur jaringan *Software-Defined Networking (SDN),* di mana fungsi pengaturan lalu lintas jaringan dipisahkan secara jelas antara *control plane* dan *data plane*. *Control plane* mengatur keputusan mengenai alur lalu lintas, sedangkan *data plane* bertanggung jawab untuk meneruskan lalu lintas sesuai dengan keputusan dari *control plane*. *OpenFlow* memungkinkan *controller SDN* untuk memberikan instruksi kepada perangkat jaringan seperti *switch* dan *router* untuk mengarahkan lalu lintas sesuai dengan kebutuhan jaringan. Keunggulan *OpenFlow* termasuk kemampuannya dalam menangani kebutuhan *bandwidth* yang tinggi, adaptasi terhadap aplikasi dan layanan yang dinamis, serta kemudahan dalam pengelolaan perangkat jaringan dalam suatu jaringan [13].

J. *Intrusion Detection System*

Intrusion Detection System (IDS) dirancang untuk mengidentifikasi aktivitas mencurigakan atau tidak sah dalam jaringan komputer dan sistem informasi dengan memantau lalu lintas jaringan atau *log* sistem. *IDS* mendeteksi serangan keamanan, pelanggaran kebijakan, dan aktivitas berbahaya lainnya yang mengancam integritas, kerahasiaan, dan ketersediaan data. *IDS* dapat dibagi menjadi dua kategori utama: *Network-based Intrusion Detection Systems (NIDS)* dan *Host-based Intrusion Detection Systems (HIDS)*. *NIDS* ditempatkan pada titik-titik strategis dalam jaringan untuk memantau lalu lintas yang masuk dan keluar dari seluruh perangkat dalam jaringan tersebut, sementara *HIDS* diinstall pada perangkat individu dan memantau aktivitas dalam perangkat tersebut, termasuk *log* sistem, file sistem, dan proses-proses yang berjalan [14].

K. *Snort*

Snort adalah sistem pendeteksi intrusi *Network-Based Intrusion Detection System (NIDS)* yang bersifat *open source* dan menggunakan aturan *(rule-driven*) sebagai dasar kerjanya. *Snort* memungkinkan pengguna untuk memonitor lalu lintas jaringan secara pasif dan memberikan peringatan atau *alert* saat terdeteksi ancaman keamanan. *Snort* dapat dijalankan pada berbagai platform sistem operasi, termasuk *Linux, Windows,* dan *MacOS*. *Snort* memiliki empat mode operasi yang berbeda, yaitu *sniffer mode, logger mode, intrusion detection mode,* dan *inline mode*, yang dapat disesuaikan dengan kebutuhan pengguna dalam memonitor dan mengamankan jaringan. *Snort* dapat digunakan secara mandiri atau diintegrasikan dengan perangkat *firewall* seperti *Cisco* untuk meningkatkan keamanan jaringan dengan memberikan deteksi yang lebih canggih dan respons yang lebih cepat terhadap serangan yang terdeteksi [15].

L. *SSH BruteForce*

SSH Brute Force adalah metode serangan yang bertujuan untuk mendapatkan akses ilegal ke sistem melalui layanan *SSH* dengan mencoba berbagai kombinasi *username* dan *password* secara berulang-ulang. Metode ini memanfaatkan kapabilitas komputasi untuk mencoba beragam kemungkinan kredensial hingga ditemukan kombinasi yang tepat. Serangan *SSH Brute Force* menjadi ancaman signifikan dalam keamanan jaringan karena keberhasilannya dapat memberikan penyerang akses penuh ke sistem target, memungkinkan pencurian data sensitif, modifikasi konfigurasi sistem, atau eksekusi *malware*. Alat yang sering digunakan untuk melakukan serangan *SSH Brute Force* adalah *Hydra, Medusa,* dan *John the Ripper* [16].

M. *Distribute Denial of Service*

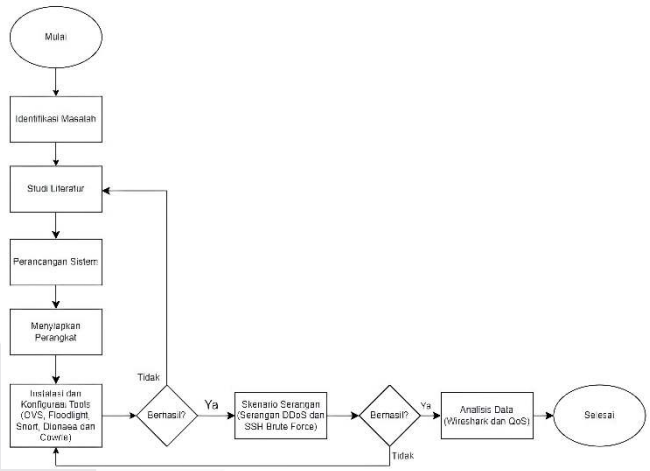
Distributed Denial of Service (DDoS) adalah bentuk serangan siber yang bertujuan untuk membuat layanan, jaringan, atau server menjadi tidak dapat diakses oleh pengguna yang berhak. Serangan ini dilakukan dengan mengirimkan sejumlah besar permintaan atau lalu lintas data secara serentak dari berbagai sumber yang tersebar di internet. Akibatnya, sumber daya *server* atau jaringan menjadi kewalahan dan tidak mampu menangani permintaan tersebut, sehingga menyebabkan layanan menjadi lambat atau bahkan tidak berfungsi sama sekali. Jenis serangan *DDoS* meliputi *volumetric attacks, protocol attacks, application layer attacks, DNS amplification attacks,* dan *reflected attacks*. Serangan *DDoS* dapat menyebabkan gangguan yang signifikan pada layanan *online*, termasuk penurunan kinerja sistem, risiko kebocoran data, dan potensi gangguan terhadap layanan-layanan publik [17].

N. *Quality of Service*

Quality of Service (QoS) adalah istilah yang mengacu pada serangkaian teknologi dan mekanisme yang dirancang untuk memastikan bahwa aplikasi jaringan memenuhi kinerja yang diharapkan, terutama dalam hal keandalan, kecepatan, dan efisiensi. *QoS* memainkan peran penting dalam manajemen jaringan dengan mengontrol dan mengoptimalkan berbagai atribut jaringan seperti *throughput*, *delay* (latensi), *jitter*, dan *packet loss*. *QoS* biasanya diukur menggunakan beberapa metrik utama, yaitu *throughput* (jumlah data yang berhasil dikirimkan dalam satuan waktu tertentu), *delay* (waktu yang dibutuhkan data untuk berpindah dari sumber ke tujuan), *jitter* (variasi dalam waktu *delay* antara paket data yang diterima), dan *packet loss* (persentase paket data yang hilang selama transmisi). *QoS* sangat penting untuk aplikasi *real-time* seperti *VoIP*, *video streaming*, dan *gaming*, di mana kualitas layanan yang konsisten sangat dibutuhkan [18].

III. METODE

Untuk menyusun penelitian ini, ada beberapa tahapan yang harus dilakukan. Berikut adalah diagram alir dari tahapan-tahapan penelitian :



GAMBAR 1
Diagram Alur Penelitian

Tahap awal dari penelitian ini adalah mengidentifikasi permasalahan yang akan dibahas, yaitu meningkatnya jumlah pengguna internet telah membawa risiko yang signifikan terhadap serangan siber, termasuk *DDoS* dan *SSH BruteForce*. Meski *Software Defined Network (SDN)* menawarkan fleksibilitas dan kontrol yang lebih baik dari infrastruktur jaringan tradisional, isu keamanan tetap menjadi perhatian utama. *Floodlight* sebagai *SDN controller*, *Snort* sebagai *Intrusion Detection System (IDS), Dionaea* dan *Cowrie* sebagai *Hybrid Honeypot* menghadirkan pendekatan yang efektif untuk mendeteksi dan merespons serangan siber. Implementasi teknologi ini di dalam jaringan *SDN* memungkinkan peningkatan keamanan melalui deteksi dini dan respons cepat terhadap ancaman siber.

Selanjutnya adalah perancangan sistem, langkah ini dimulai dengan mengidentifikasi kebutuhan perangkat keras dan perangkat lunak yang diperlukan. Tabel 1 merupakan kebutuhan perangkat kearas dan Tabel 2 merupakan kebutuhan perangkat lunak.

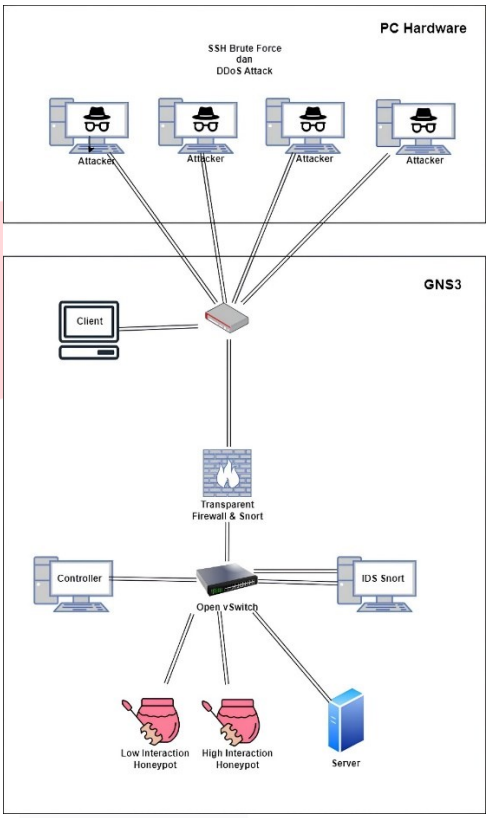
TABEL 1
Kebutuhan Perangkat Keras

No	Perangkat	Unit	Keterangan
1	Komputer	4	Sebagai *attacker*
2	*BitBox*	1	Sebagai *GNS3 Server* dan penyedia konektivitas *hardware*

TABEL 2
Kebutuhan Perangkat Lunak

No	Pengguna	Perangkat Lunak	Keterangan
1	*PC Server*	*Ubuntu Desktop version 18.04*	Sistem operasi *server*
		Apache Web Server	Penyedia *web server*
2	*PC IDS*	*Ubuntu Desktop version 18.04*	Sistem operasi *PC IDS*
		Snort	*Tools IDS*
3	*PC Controller*	*Ubuntu Desktop version 18.04*	Sistem operasi *PC Controller*
		Floodlight	*Tools SDN Controller*
4	*PC Hybrid Honeypot*	*Ubuntu Desktop version 18.04*	Sistem operasi *PC Hybrid Honeypot*
		Dionaea	*Tools Low Interaction Honeypot*
5	*PC Hybrid Honeypot*	*Ubuntu Desktop version 20.04*	Sistem operasi *PC Hybrid Honeypot*
		Cowrie	*Tools High Interaction Honeypot*
6	*PC Client*	*Ubuntu version 18.04*	Sistem operasi *client*
7	*PC* Penyerang	*Ubuntu Desktop version 18.04*	Sistem operasi penyerang
		Hping3	*Tools DDoS Attack*
		Hydra	*Tools Brute force*
8	*BitBox*	*Ubuntu Server 22.04*	Sistem operasi *BitBox Server*
		Virtualization	*GNS3 Server*
9	*Router*	*OPNSense*	Pembagi jaringan
10	*Switch*	*OpenVSwitch*	Penyalur *traffic* jaringan

Selanjutnya adalah merancang sistem, langkah pertama yang dilakukan adalah membuat rancangan topologi jaringan. Gambar 2 merupakan Topologi Jaringan yang digunakan dalam penelitian ini. Selanjutnya, Tabel 3 menunjukkan spesifikasi untuk perangkat keras, Tabel 4 menunjukkan spesifikasi untuk perangkat lunak yang digunakan dalam penelitian ini dan Tabel 5 menunjukkan alamat IP yang digunakan dalam penelitian ini.



GAMBAR 2
Topologi Jaringan

TABEL 3
Spesifikasi Perangkat Keras

No	Nama Perangkat	Spesifikasi
1	PC *Attacker*	*Ubuntu 18.04 Asus pc all in one v222gakwa141t-dualcore*
2	*BitBox*	*Ubuntu Server 22.04*

TABEL 4
Spesifikasi Perangkat Lunak

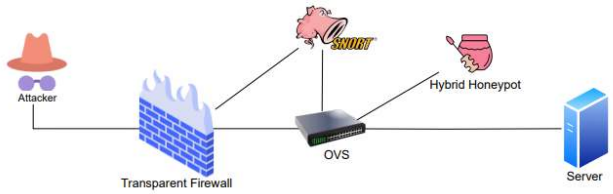
No	Nama Perangkat	Spesifikasi
1	*PC IDS Snort*	*Ubuntu 18.04* 15GB
2	*PC High Interaction Honeypot*	*Ubuntu 20.04* 15GB
3	*PC Low Interaction Honeypot*	*Ubuntu 18.04* 15GB
4	*PC Controller*	*Ubuntu 18.04* 15GB
5	*PC Server*	*Ubuntu 18.04* 15GB
6	*PC Client*	*Ubuntu 18.04* 15GB

7	*Open Virtual Switch*	*Open vSwitch 2.17.9*
8	*Router OPNSense*	*OPNSense 24.1*

TABEL 5
IP Perangkat

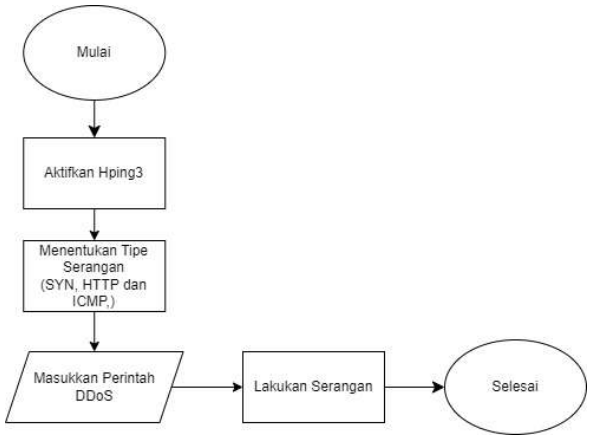
No	Nama Perangkat	Alamat *IP*
1	*Client*	192.168.42.2
2	*Attacker 1*	192.168.42.3
3	*Attacker 2*	192.168.42.4
4	*Attacker 3*	192.168.42.5
5	*Attacker 4*	192.168.42.6
6	*Controller*	192.168.24.2
7	*Honeypot Cowrie*	192.168.24.3
8	*Honeypot Dionaea*	192.168.24.4
9	*Server*	192.168.24.5
10	*IDS Snort*	192.168.24.6
11	*OpenvSwitch*	192.168.24.14

Selanjutnya adalah menyusun rancangan sistem untuk menguji respons terhadap serangan yang dilakukan.



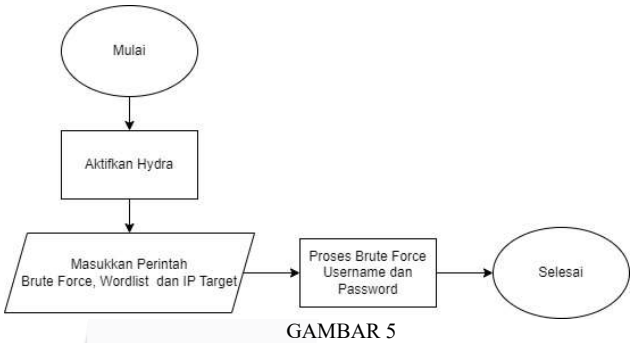
GAMBAR 3
Perancangan Sistem Menggunakan *Tools Snort, Hybrid Honeypot, Open Virtual Switch,* dan *SDN*

Pada gambar 3 terdapat attacker yang masuk melewati *Transparent Firewall*. *Transparent firewall* yang juga disebut *Bridge Firewall* bertindak sebagai penghubung tanpa memerlukan konfigurasi alamat *IP*. Setelah melewati *transparent firewall* seluruh trafik jaringan diarahkan melalui *Open Virtual Switch* yang menggunakan *port mirroring* untuk mengawasi semua paket yang melewatinya. *Port mirroring* ini digunakan untuk memantau oleh *PC IDS* yang menjalankan *Snort*. *IDS Snort* akan menganalisis paket-paket tersebut dan mengirimkan peringatan *alert* kepada *Floodlight Controller* jika mendeteksi adanya serangan. Setelah menerima *alert* dari *Snort*, *Floodlight* akan mengubah jalur trafik jaringan untuk mengarahkan trafik yang mencurigakan menuju *Hybrid Honeypot* menggunakan *Open Virtual Switch* yang bertindak sebagai protokol *OpenFlow* dengan bantuan *flow rules*. *Floodlight* dan *Open Virtual Switch* saling terhubung melalui *REST API* yang memungkinkan *administrator* jaringan untuk mengelola aturan dan konfigurasi jaringan secara dinamis berdasarkan data ancaman yang diterima dari *Snort*. *Hybrid Honeypot* dirancang untuk menangkap *attacker* masuk ke *server* palsu. *Attacker* dapat menyerang *Hybrid Honeypot* sementara *administrator* dapat memantau paket guna melakukan penghitungan performansi *Quality of Service* menggunakan *PC Hybrid Honeypot*.



GAMBAR 4
Diagram Alur Pengujian Serangan *DDoS*

Serangan yang dilakukan merupakan serangan *DDoS* menggunakan *Tools Hping3*. Langkah pertama adalah mengaktifkan aplikasi *Hping3* dan menentukan tipe serangan yang akan dilakukan. Metode serangan yang dipilih mencakup *SYN, HTTP* dan *ICMP*. Kemudian masukkan perintah *DDoS* yang meliputi jumlah paket, interval waktu, *port* yang akan diserang dan alamat *IP* target yang akan diserang Setelah semua pengaturan tersebut selesai, *Tools Hping3* dijalankan dengan mengklik tombol "*Enter*" untuk melancarkan serangan.



GAMBAR 5
Diagram Alur Pengujian Serangan SSH BruteForce

Serangan kedua yang dilakukan merupakan *SSH Brute force* menggunakan *Tools Hydra*. Langkah pertama adalah mengaktifkan aplikasi *Hydra* lalu memasukkan perintah *Brute force*, *Wordlist* dan alamat *IP Target*. Kemudian ketika proses serangan berjalan, *Hydra* akan melakukan *Brute force Username* dan *Password*.

Selanjutnya ada analisis data, penelitian ini menggunakan metode *Quality of Service (QoS)*. Parameter *QoS* yang diukur meliputi *throughput, delay, jitter, dan packet loss.*

A. *Throughput*

$$Throughput = \frac{paket\ data\ yang\ diterima}{waktu\ pengiriman\ data} \quad (1)$$

B. *Delay*

$$Delay = \frac{total\ delay}{jumlah\ total\ paket} \quad (2)$$

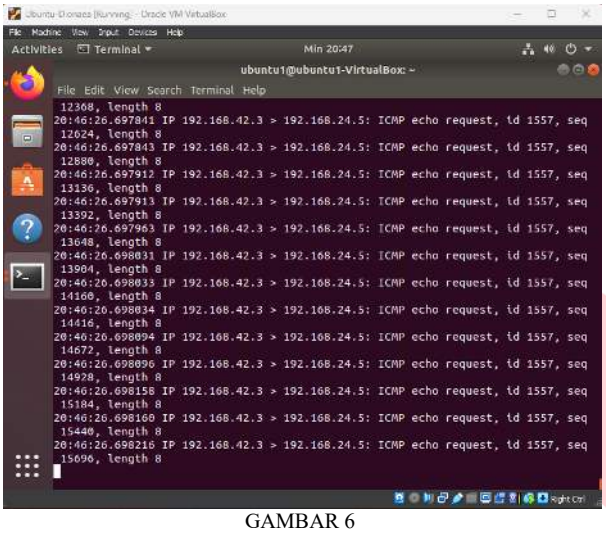
C. *Jitter*

$$Jitter = \frac{total\ variasi\ delay}{total\ paket\ yang\ diterima} \quad (3)$$

D. *Packet* Loss

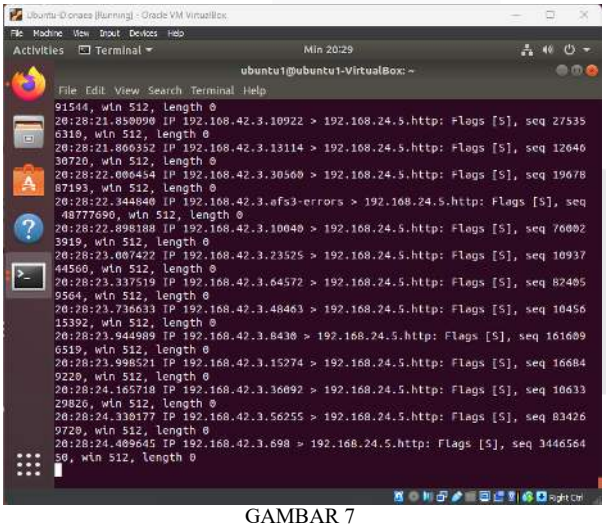
$$Packet\ Loss = \frac{(paket\ dikirim - paket\ diterima)}{paket\ dikirim} x100\% (4)$$

IV. HASIL DAN PEMBAHASAN



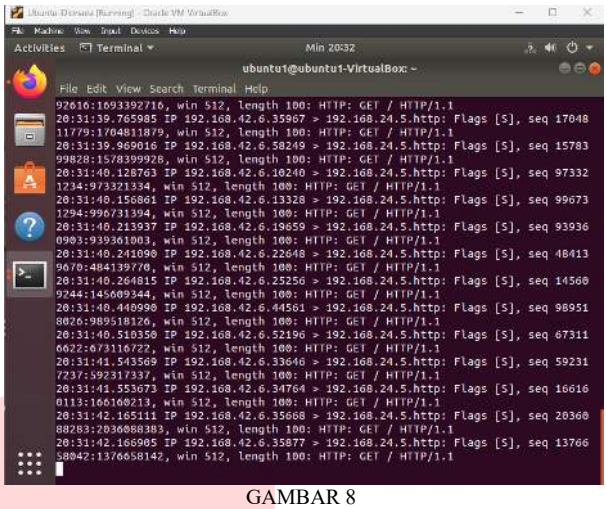
GAMBAR 6
Serangan *ICMP Flood*

Gambar 6 menunjukkan lalu lintas *ICMP echo request* yang berasal dari *IP* 192.168.42.3 dan ditujukan ke *IP* 192.168.24.5, dengan pengiriman yang cepat dan berulang. Pola ini merupakan karakteristik dari serangan *ICMP Flood*. Namun, paket-paket *ICMP* tersebut tidak sampai ke *Server* 192.168.24.5 karena telah berhasil dibelokkan oleh *Flow rules* yang telah dikonfigurasi di *Floodlight Controller*. Sebagai bagian dari mitigasi serangan, paket-paket *ICMP* tersebut diarahkan ke perangkat *Honeypot Dionaea*.



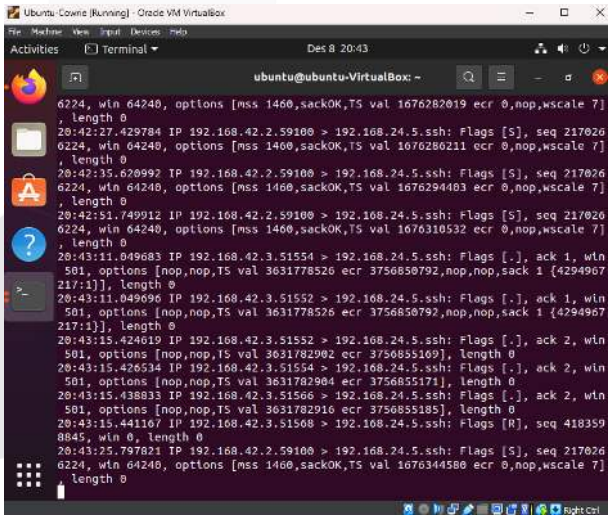
GAMBAR 7
Serangan *SYN Flood*

Gambar 7 menunjukkan lalu lintas jaringan berupa paket *SYN* dengan *Flags [S]* yang diarahkan ke 192.168.24.5 pada *port HTTP*. Paket-paket *SYN* ini merupakan bagian dari serangan *SYN Flood*, di mana sejumlah besar permintaan koneksi *TCP* dikirimkan dalam waktu singkat dengan tujuan membanjiri *server* dan mengganggu layanan jaringan. Namun, serangan tersebut tidak lagi mencapai *Server* 192.168.24.5 karena berhasil dialihkan ke perangkat *Honeypot Dionaea* menggunakan *Flow Rules* yang diterapkan melalui *Floodlight Controller*.



GAMBAR 8
Serangan *HTTP Flood*

Gambar 8 menunjukkan lalu lintas jaringan berupa permintaan *HTTP GET* dari beberapa *IP attacker* seperti 192.168.42.6 yang diarahkan ke 192.168.24.5 pada *port HTTP (port 80)*. Paket-paket ini memiliki pola pengiriman yang cepat dengan panjang data sebesar 100 *byte*, yang merupakan karakteristik serangan *HTTP Flood*. Serangan ini bertujuan untuk membanjiri *server* dengan permintaan *HTTP* palsu sehingga layanan menjadi terganggu. Namun, serangan tidak lagi mencapai *Server* 192.168.24.5 karena telah berhasil dialihkan menggunakan *Flow Rules* yang diterapkan melalui *Floodlight Controller*. Dengan aturan tersebut, lalu lintas *HTTP Flood* diarahkan ke perangkat *Honeypot Dionaea.*



GAMBAR 9
Serangan *SSH BruteForce*

Gambar 9 menunjukkan lalu lintas jaringan berupa paket *TCP* yang mengarah ke *port SSH* (22) pada *IP* 192.168.24.5. Paket-paket ini mencakup *Flags [S]* yang menunjukkan adanya permintaan koneksi *TCP (SYN)* serta beberapa *ACK* dan *RST* yang terkait dengan komunikasi *TCP*. Pola ini merupakan karakteristik serangan *SSH Brute Force* di mana sejumlah besar percobaan koneksi dilakukan dalam waktu singkat untuk mencoba mengakses layanan *SSH*. Namun, lalu lintas ini tidak mencapai *Server* 192.168.24.5, karena telah berhasil dialihkan menggunakan *Flow Rules*

yang diterapkan melalui *Floodlight Controller*. Dengan aturan ini serangan dialihkan ke perangkat *.Honeypot Cowrie.*

TABEL 6
Pengukuran *QoS PC Honeypot Dionaea* Sebelum Serangan

Dionaea	Throughput (bit/s)	Packet Loss	Delay (ms)	Jitter (ms)
5 Menit	2	0%	900,12	928,09
10 Menit	3	0,384%	767,26	782,91
15 Menit	2,45	0,256%	848,72	855,32
20 Menit	2,26	0,192%	908,61	913,66

Tabel 6 menunjukkan pada parameter *throughput* nilai yang diperoleh yaitu 2 *bit/s* pada 5 menit, meningkat menjadi 3 *bit/s* pada 10 menit, turun menjadi 2,45 *bit/s* pada 15 menit dan menurun menjadi 2,26 *bit/s* pada 20 menit. Untuk *Packet Loss* 0% pada 5 menit, 0,384% pada 10 menit, 0,256% pada 15 menit dan 0,192% pada 20 menit. *Delay* menunjukkan penurunan yang konsisten seiring bertambahnya durasi yaitu dari 900,12 *ms* pada 5 menit menjadi 767,26 *ms* pada 10 menit, 848,72 *ms* pada 15 menit dan sedikit meningkat menjadi 908,61 *ms* pada 20 menit. *Jitter* yang mengalami penurunan dari 928,09 *ms* pada 5 menit menjadi 782,91 *ms* pada 10 menit dan 855,32 *ms* pada 15 menit, dan meningkat sedikit menjadi 913,66 *ms* pada 20 menit.

TABEL 7
Pengukuran *QoS PC Honeypot Cowrie* Sebelum Serangan

Cowrie	Throughput (bit/s)	Packet Loss	Delay (ms)	Jitter (ms)
5 Menit	1,51	0%	959,29	952,79
10 Menit	1,71	0%	993,16	963,21
15 Menit	1,51	0,15%	958,70	935,33
20 Menit	2,18	0,19%	917,62	900,31

Tabel 7 menunjukkan pada parameter *Throughput* hasil pengukuran terjadi perubahan yaitu 1,51 *bit/s* pada 5 menit, meningkat menjadi 1,71 *bit/s* pada 10 menit, dan menurun menjadi 1,51 *bit/s* pada 15 menit, dan sedikit meningkat menjadi 2,18 *bit/s* pada 20 menit. Untuk *Packet Loss* pengukuran mencatatkan nilai 0% pada 5 menit, 0% pada 10 menit, 0,15% pada 15 menit, dan 0,19% pada 20 menit. *Delay* menunjukkan perubahan seiring bertambahnya durasi, yaitu dari 959,29 *ms* pada 5 menit menjadi 993,16 *ms* pada 10 menit, dan 958,70 *ms* pada 15 menit, dan menurun menjadi 917,62 *ms* pada 20 menit. *Jitter* mengalami perubahan dari 952,79 *ms* pada 5 menit menjadi 963,21 *ms* pada 10 menit dan 935,33 *ms* pada 15 menit, dan menurun menjadi 900,31 *ms* pada 20 menit.

TABEL 8
Pengukuran *QoS* Saat Serangan *SYN Flood*

SYN Flood	Throughput (bit/s)	Packet Loss	Delay (ms)	Jitter (ms)
5 Menit	172,5	0%	13,21	13,12
10 Menit	166,4	0,002%	13,79	13,68
15 Menit	182,6	0,002%	11,00	10,.56
20 Menit	182,6	0,002%	10,24	10,.92

Tabel 8 menunjukkan pada parameter *Throughput* terjadi perubahan secara bertahap dari 172,5 *bit/s* pada 5 menit, menjadi 166,4 *bit/s* pada 10 menit, 182,6 *bit/s* pada 15 menit, dan 182,6 *bit/s* pada 20 menit. *Packet Loss* menunjukkan nilai 0% pada 5 menit, serta 0,002% pada 10 menit, 15 menit dan 20 menit. Pada *Delay* terjadi perubahan, yaitu dari 13.21 *ms* pada 5 menit, menjadi 13.79 *ms* pada 10 menit, 11.00 *ms* pada 15 menit, dan menurun menjadi 10.24 *ms* pada 20 menit. *Jitter* meningkat dari 13.12 *ms* pada 5 menit, menjadi 13.68 *ms* pada 10 menit, 10.96 *ms* pada 15 menit dan menurun menjadi 10.22 *ms* pada 20 menit.

TABEL 9
Pengukuran *QoS* Saat Serangan *HTTP Flood*

HTTP Flood	Throughput (bit/s)	Packet Loss	Delay (ms)	Jitter (ms)
5 Menit	545,5	0,0004%	153,94	149,56
10 Menit	478,1	0,0004%	453,10	458,99
15 Menit	525,0	0,0004%	340,78	344,92
20 Menit	490,3	0,0159%	296,15	300,02

Tabel 9 menunjukkan pada parameter *Throughput* terjadi perubahan dari 545,5 *bit/s* pada 5 menit, menjadi 478,1 *bit/s* pada 10 menit, 525,0 *bit/s* pada 15 menit dan menurun menjadi 490,3 *bit/s* pada 20 menit. *Packet Loss* menunjukkan nilai 0,0004% pada 5 menit, 10 menit dan 15 menit, serta 0,0015% pada 20 menit. *Delay* terjadi perubahan yaitu dari 153.94 *ms* pada 5 menit, menjadi 453.10 *ms* pada 10 menit, 340.78 *ms* pada 15 menit, dan menurun menjadi 296.15 *ms* pada 20 menit. *Jitter* meningkat dari 149.56 *ms* pada 5 menit, menjadi 458.99 *ms* pada 10 menit, 344.92 *ms* pada 15 menit, dan menurun menjadi 300.02 *ms* pada 20 menit.

TABEL 10
Pengukuran *QoS* Saat Serangan *ICMP Flood*

ICMP Flood	Throughput (bit/s)	Packet Loss	Delay (ms)	Jitter (ms)
5 Menit	8014,7	2,4%	60,06	60,06
10 Menit	8069,9	2,7%	64,21	64,21
15 Menit	8073,5	3,46%	62,62	62,62
20 Menit	8036,7	3,05%	62,11	62,11

Tabel 10 menunjukkan pada parameter *Throughput* terjadi perubahan nilai selama serangan berlangsung. *Throughput* tercatat sebesar 8014,7 *bit/s* pada 5 menit,

meningkat sedikit menjadi 8069,9 *bit/s* pada 10 menit, meningkat menjadi 8073,5 *bit/s* pada 15 menit, dan menurun menjadi 8036,7 *bit/s* pada 20 menit. Pada parameter *Packet Loss* tercatat 2,4% pada 5 menit, 2,7% pada 10 menit, 3,46% pada 15 menit, dan 3,05% pada 20 menit, menunjukkan adanya kehilangan paket data selama serangan berlangsung, dengan sedikit peningkatan pada durasi yang lebih panjang. Pada parameter *Delay* terjadi perubahan yaitu dari 60,06 *ms* pada 5 menit, menjadi 64,21 *ms* pada 10 menit, 62,62 *ms* pada 15 menit, dan menurun menjadi 62,11 *ms* pada 20 menit. *Jitter* tercatat sebesar 60,06 *ms* pada 5 menit, meningkat menjadi 64,21 *ms* pada 10 menit, 62,62 *ms* pada 15 menit, dan menurun menjadi 62,11 *ms* pada 20 menit.

TABEL 11
Pengukuran *QoS* Saat Serangan *SSH BruteForce*

SSH BruteForce	Throughput (bit/s)	Packet Loss	Delay (ms)	Jitter (ms)
5 Menit	3,24	0,064%	367,38	363,92
10 Menit	3,76	0,064%	302,27	300,55
15 Menit	3,69	0,066%	283,55	282,40
20 Menit	3,81	0,081%	272,49	271,66

Tabel 4.7 menunjukkan pada parameter *Throughput* terlihat perubahan nilai dari 3,24 *bit/s* pada 5 menit, menjadi 3,76 *bit/s* pada 10 menit, 3,69 *bit/s* pada 15 menit, dan meningkat menjadi 3,81 *bit/s* pada 20 menit. Pada parameter *Packet Loss* seluruh durasi pengukuran menunjukkan nilai yang sangat kecil, yaitu 0,064% pada 5 menit, 0,064% pada 10 menit, 0,066% pada 15 menit, dan 0,081% pada 20 menit. Parameter *Delay* menunjukkan penurunan nilai seiring durasi serangan. *Delay* tercatat sebesar 367.38 *ms* pada 5 menit, menurun menjadi 302.27 *ms* pada 10 menit, 283.55 *ms* pada 15 menit dan semakin rendah menjadi 272.49 *ms* pada 20 menit. Penurunan serupa juga terjadi pada parameter *Jitter*. Nilai *Jitter* sebesar 363.92 *ms* pada 5 menit menurun menjadi 300.55 *ms* pada 10 menit, 282.03 *ms* pada 15 menit dan mencapai 271.66 *ms* pada 20 menit.

TABEL 12
Pengukuran *QoS PC Honeypot Dionaea* Setelah Serangan

Dionaea	Throughput (bit/s)	Packet Loss	Delay (ms)	Jitter (ms)
5 Menit	1,78	0%	882,38	913,85
10 Menit	2	0%	862,57	879,19
15 Menit	1,90	0%	914,72	925,85
20 Menit	1,91	0%	938,22	946,59

Tabel 4.8 menunjukkan pada parameter *Throughput* terlihat adanya perubahan nilai dari 1,78 *bit/s* pada 5 menit, menjadi 2,00 *bit/s* pada 10 menit, 1,90 *bit/s* pada 15 menit dan sedikit meningkat menjadi 1,91 *bit/s* pada 20 menit. Pada parameter *Packet Loss* nilai tetap berada pada 0% untuk semua durasi pengukuran, yang berarti tidak ada paket data yang hilang setelah serangan selesai. Parameter *Delay* menunjukkan perubahan dari 882.38 *ms* pada 5 menit, menjadi 862.57 *ms* pada 10 menit, 914.72 *ms* pada 15 menit, dan 938.22 *ms* pada 20 menit. *Jitter* tercatat sebesar 913.85 *ms* pada 5 menit, menurun menjadi 879.19 *ms* pada 10 menit, 925.85 *ms* pada 15 menit dan 946.59 *ms* pada 20 menit.

TABEL 13
Pengukuran *QoS PC Honeypot Cowrie* Setelah Serangan

Cowrie	Throughput (bit/s)	Packet Loss	Delay (ms)	Jitter (ms)
5 Menit	2,26	0,258%	1,016,28	1,082,80
10 Menit	2	0,212%	1,037,40	1,037,40
15 Menit	1,80	0,176%	997,86	997,86
20 Menit	1,87	0,132%	958,90	958,90

Tabel 4.9 menunjukkan pada parameter *Throughput* terjadi penurunan nilai secara bertahap dari 2,26 *bit/s* pada 5 menit, menjadi 2 *bit/s* pada 10 menit, 1,80 *bit/s* pada 15 menit, dan 1,87 *bit/s* pada 20 menit. Pada parameter *Packet Loss* nilai tercatat kecil yaitu 0,258% pada 5 menit, 0,212% pada 10 menit, 0,176% pada 15 menit, dan 0,132% pada 20 menit. Pada parameter *Delay* terjadi perubahan nilai yaitu dari 1.016.28 *ms* pada 5 menit, menjadi 1.037.40 *ms* pada 10 menit, 997.86 *ms* pada 15 menit dan 958.90 *ms* pada 20 menit. *Jitter* tercatat menurun dari 1.082.80 *ms* pada 5 menit, menjadi 1.037.40 *ms* pada 10 menit, 997.86 *ms* pada 15 menit, dan 958.90 *ms* pada 20 menit.

V. KESIMPULAN

Penelitian ini berhasil merancang dan mengimplementasikan sistem keamanan jaringan berbasis *Software-Defined Network (SDN)* dengan menggunakan *Hybrid Honeypot*. Sistem ini mengintegrasikan *Intrusion Detection System (IDS) Snort, Floodlight Controller,* serta *Honeypot Dionaea* dan *Cowrie* untuk mendeteksi dan mengalihkan serangan seperti *DDoS ICMP Flood, SYN Flood, HTTP Flood,* dan *SSH Brute Force*. Semua lalu lintas serangan berhasil dialihkan ke *Honeypot* melalui penerapan *Flow Rules* pada *Floodlight Controller,* sehingga *server* utama tetap terlindungi. *Honeypot Dionaea* dan *Cowrie* berfungsi sebagai perangkat jebakan yang efektif untuk menangkap dan menganalisis *traffic* serangan.

Pengukuran performa jaringan dilakukan berdasarkan parameter *Quality of Service (QoS),* yaitu *throughput, delay, jitter,* dan *packet loss*. Sebelum serangan dilakukan, performa jaringan menunjukkan kestabilan. Pada *Honeypot Dionaea*, *throughput* meningkat dari 2 *bit/s* (5 menit) menjadi 3 *bit/s* (10 menit), namun menurun menjadi 2,26 *bit/s* (20 menit), sementara *delay* dan *jitter* cenderung stabil. *Honeypot Cowrie* juga menunjukkan kestabilan dengan *throughput* meningkat dari 1,51 *bit/s* (5 menit) menjadi 2,18 *bit/s* (20 menit), dan *delay* serta *jitter* tetap stabil.

Selama serangan, dampak pada performa jaringan bervariasi tergantung jenis serangan. Serangan *SYN Flood* mengakibatkan *throughput* menurun dari 6.134 *bit/s* (5 menit) menjadi 6.075 *bit/s* (20 menit), dengan *delay* meningkat dari 765 *ms* menjadi 913 *ms*, dan *jitter* yang naik turun. Serangan *HTTP Flood* menyebabkan *throughput* turun drastis dari 4.578 *bit/s* (3 menit) menjadi 355 *bit/s* (10 menit), disertai peningkatan *delay* dari 2.305 *ms* menjadi 2.843 *ms* dan *jitter* yang meningkat. Serangan *ICMP Flood* menunjukkan *throughput* yang stabil di sekitar 8.000 *bit/s,*

namun *packet loss* meningkat hingga 3,05%, dengan *delay* dan *jitter* yang konsisten di kisaran 60 *ms*. Sementara itu, serangan *SSH Brute Force* justru meningkatkan *throughput* dari 3,24 *bit/s* (5 menit) menjadi 3,81 *bit/s* (20 menit), dengan *delay* yang menurun dari 367 *ms* menjadi 272 *ms* dan *jitter* yang stabil.

Setelah serangan berhenti, performa jaringan menunjukkan pemulihan. Pada *Honeypot Dionaea*, *throughput* meningkat dari 4.792 *bit/s* (3 menit) menjadi 5.625 *bit/s* (10 menit), dengan *delay* dan *jitter* yang terus menurun. *Honeypot Cowrie* juga menunjukkan pemulihan, meskipun *throughput* sedikit menurun dari 2,26 *bit/s* (5 menit) menjadi 1,87 *bit/s* (20 menit), sementara *delay* dan *jitter* tetap stabil. Secara keseluruhan, sistem ini terbukti efektif dalam melindungi *server* utama dari serangan siber, sambil mempertahankan performa jaringan yang stabil dan mampu pulih setelah serangan berakhir.

REFERENSI

[1] Simon Kemp, "Digital 2024 Indonesia," Datareportal. Accessed: May 07, 2024. [Online]. Available: https://datareportal.com/reports/digital-2024-indonesia

[2] Badan Siber dan Sandi Negara, "Laporan Tahunan Layanan Honeynet BSSN 2024," 2023. Accessed: May 07, 2024. [Online]. Available: https://www.bssn.go.id/wp-content/uploads/2024/04/LAPTAH_HONEYNET_2023.pdf

[3] CNN Indonesia, "KPU Ungkap Situs Resmi Alami Ratusan Juta Serangan pada Pemilu 2024," *CNN Indonesia*, Jakarta, Feb. 15, 2024. Accessed: Mar. 10, 2024. [Online]. Available: https://www.cnnindonesia.com/nasional/20240215005235-617-1062767/kpu-ungkap-situs-resmi-alami-ratusan-juta-serangan-pada-pemilu-2024

[4] H. Wang and B. Wu, "SDN-based hybrid honeypot for attack capture," *IEEE 3rd Information Technology,Networking,Electronic and Automation Control Conference*, pp. 1602–1606, 2019.

[5] M. Iqbal and M. Arif Ramadhan, "Analisa Quality of Service pada Jaringan Wireless Berbasis Software-Defined Network dengan Protokol Openflow Menggunakan Floodlight Controller," 2020.

[6] S. Yoga, "Analisis Performansi Software Defined Network (SDN)," Universitas Islam Riau, Riau, 2021.

[7] F. Yasin, Abdul Fadlil, and Rusydi Umar, "Identifikasi Bukti Forensik Jaringan Virtual Router Menggunakan Metode NIST," *Jurnal RESTI (Rekayasa Sistem dan Teknologi Informasi)*, vol. 5, no. 1, pp. 91–98, Feb. 2021, doi: 10.29207/resti.v5i1.2784.

[8] B. Dodiya and U. K. Singh, "Malicious Traffic analysis using Wireshark by collection of Indicators of Compromise," *Int J Comput Appl*, vol. 183, no. 53, pp. 1–6, Feb. 2022, doi: 10.5120/ijca2022921876.

[9] M. H. Asadullah, "Sistem Keamanan Server Dengan Honeypot Dan Intrusion Detection System (IDS) (Studi Kasus Perusahaan Printing Somatex)," Universitas Sebelas Maret, Surakarta, 2019. Accessed: Mar. 22, 2024. [Online]. Available: https://core.ac.uk/download/pdf/211786224.pdf

[10] Ilham Fitra Pradana, "Deteksi Keamanan Server Menggunakan Cowrie Dan Fortigate Pada Web Server Skripsi," Universitas Islam Negeri Syarif Hidayatullah, 2023.

[11] J. Tamalaki Ohyver and D. W. Chandra, "Simulasi Keamanan Jaringan pada DPDK OpenvSwitch Berbasis Network-Based Intrusion Detection System (NIDS)," *Universitas Kristen Satya Wacana*, vol. 7, no. 3, Feb. 2023, doi: 10.35870/jti.

[12] L. R. Kalluru, "Implementing Programmable Data Plane in Open vSwitch using P4 language," Iowa State University, Iowa, 2023. Accessed: Mar. 12, 2024. [Online]. Available: https://dr.lib.iastate.edu/server/api/core/bitstreams/82f3d70e-e27d-4a5b-a081-e9e8d8e62dd1/content

[13] M. Alsaeedi, M. M. Mohamad, and A. A. Al-Roubaiey, "Toward Adaptive and Scalable OpenFlow-SDN Flow Control: A Survey," *IEEE Access*, vol. 7, pp. 107346–107379, 2019, doi: 10.1109/ACCESS.2019.2932422.

[14] Adam Fahsyah Nurzaman, "Sistem Deteksi dan Pencegahan Intrusi," School Of Information System. Accessed: Mar. 12, 2024. [Online]. Available: https://sis.binus.ac.id/2020/12/09/sistem-deteksi-dan-pencegahan-intrusi/

[15] M. Affandi *et al.*, "Implementasi Snort Sebagai Alat Pendeteksi Intrusi Menggunakan Linux," *Jurnal Teknologi Informasi*, vol. 4, no. 2, pp. 1–15, 2019, [Online]. Available: www.linux.org

[16] Alibaba Clouder, "CIA Triad and SSH Brute-Forcing," Alibaba Cloud. Accessed: Mar. 09, 2024. [Online]. Available: https://www.alibabacloud.com/blog/cia-triad-and-ssh-brute-forcing_594914

[17] T. Ariyadi, A. Restu Mukti, and H. Saputra, "Mitigation of Distributed Denial of Service (DDoS) Attacks on Software Defined Network (SDN) Architecture," vol. 21, no. 4, pp. 878–886, 2022.

[18] W. A. Prabowo, K. Fauziah, A. S. Nahrowi, M. N. Faiz, and A. W. Muhammad, "Strengthening Network Security: Evaluation of Intrusion Detection and Prevention Systems Tools in Networking Systems," *IJACSA) International Journal of Advanced Computer Science and Applications*, vol. 14, no. 9, 2023, [Online]. Available: www.ijacsa.thesai.org