# Deteksi Dan Mitigasi Serangan *Distributed Denial of Service* Pada *Software Defined Network*

http://dx.doi.org/10.28932/jutisi.v10i1.6995

Riwayat Artikel
*Received*: 21 Juli 2023 | *Final Revision*: 28 April 2024 | *Accepted*: 28 April 2024



Dheni Yulia Dinda Pratiwi✉ #1, Ronald Adrian*2

# *Program Studi Teknologi Rekayasa Internet, Universitas Gadjah Mada*
*Jalan Notonegoro Bulaksumur, Yogyakarta, 55281, Indonesia*
[1]dhenipratiwi5@mail.ugm.ac.id
[2]ronald.adr@ugm.ac.id
✉Corresponding author: ronald.adr@ugm.ac.id

*Abstrak —* ***Software Defined Network* merupakan pendekatan dalam pengelolaan jaringan yang memisahkan lapisan kontrol (*control plane*) dan lapisan pengiriman (*data plane*) dalam jaringan. Pada jaringan *Software Defined Network*, *control plane* dikendalikan secara sentral melalui perangkat lunak yang disebut *controller*, sementara *data plane* terdiri dari perangkat jaringan fisik seperti *switch* dan *router*. Akan tetapi, pemisahan ini menimbulkan banyak masalah keamanan. Oleh karena itu, kebutuhan untuk melindungi jaringan dari berbagai serangan menjadi hal yang wajib dilakukan. *Distributed Denial of Service* adalah salah satu serangan bagi pengguna *Software Defined Network*. Upaya melindungi jaringan SDN dari serangan *Distributed Denial of Service* diperlukan sebuah sistem yang dapat mendeteksi dan mencegah serangan tersebut. Pada penelitian ini, dibuat sebuah sistem yang digunakan untuk mendeteksi adanya serangan *Distributed Denial of Service* dengan menggunakan Snort IDS (*Intrusion Detection System*) dan pencegahannya dengan implementasi *firewall* pada server dengan menggunakan Iptables. Implementasi Snort pada sistem *Software Defined Network* mampu mendeteksi serangan *Distributed Denial of Service* dengan akurasi mencapai 95% serangan slowhttptest, 90% serangan slowloris dan 100% serangan *low orbit ion cannon*. Rata-rata penggunaan waktu yang diperlukan untuk mendeteksi adanya serangan slowhttptest sebesar 0,72 detik, serangan slowloris sebesar 0,36 detik, dan serangan *low orbit ion cannon* sebasar 0,3 detik. Implementasi iptables pada sistem *Software Defined Network* mampu memblokir serangan *Distributed Denial of Service* dengan rata – rata waktu pemblokiran 0.91 detik terhadap serang slowhttptest, 1,89 detik terhadap serangan slowloris, 0,77 detik terhadap serangan *low orbit ion cannon*, dan sistem mampu mengelola volume koneksi yang besar sehingga mampu menjaga ketersediaan sistem *Software Defined Network*.**

*Kata kunci—* ***distributed denial of service; iptables; snort ; software defined network.***

# *Detection and Mitigation Distributed Denial of Service Attack in Software Defined Network*

*Abstract —****Software-Defined Networking is an approach to network management that separates the control plane from the data plane of the network. In Software-Defined Networking, the control plane is centrally controlled by software called a "controller", while the data plane consists of physical network devices such as switches and routers. However, this separation creates many security issues. Therefore, it is imperative to protect the network from various attacks. Distributed Denial of Service is one such attack that poses a hurdle for Software-Defined Networking users. Efforts to protect the Software-Defined Networking network from Distributed Denial of Service attacks require a system that can detect and***

***prevent these attacks. In this research, a system is created that detects Distributed Denial of Service attacks using Snort IDS (Intrusion Detection System) and prevents them by implementing a firewall on the server using Iptables. The implementation of Snort in the Software-Defined Networking system is able to detect Distributed Denial of Service attacks with an accuracy of 95% for slowhttptest attacks, 90% for slowloris attacks, and 100% for low orbit ion cannon attacks. The average time to detect a slowhttptest attack is 0.72 seconds, a slowloris attack is 0.36 seconds, and a low orbit ion cannon attack is 0.3 seconds. The implementation of iptables in the Software-Defined Networking system is able to block Distributed Denial of Service attacks with an average blocking time of 0.91 seconds against slowhttptest attacks, 1.89 seconds against slowloris attacks, and 0.77 seconds against low orbit ion cannon attack attacks, and the system is able to manage large connection volumes to maintain the availability of the Software-Defined Networking system.***

*Keywords*— ***distributed denial of service; iptables; snort ; software defined network.***

## I. PENDAHULUAN

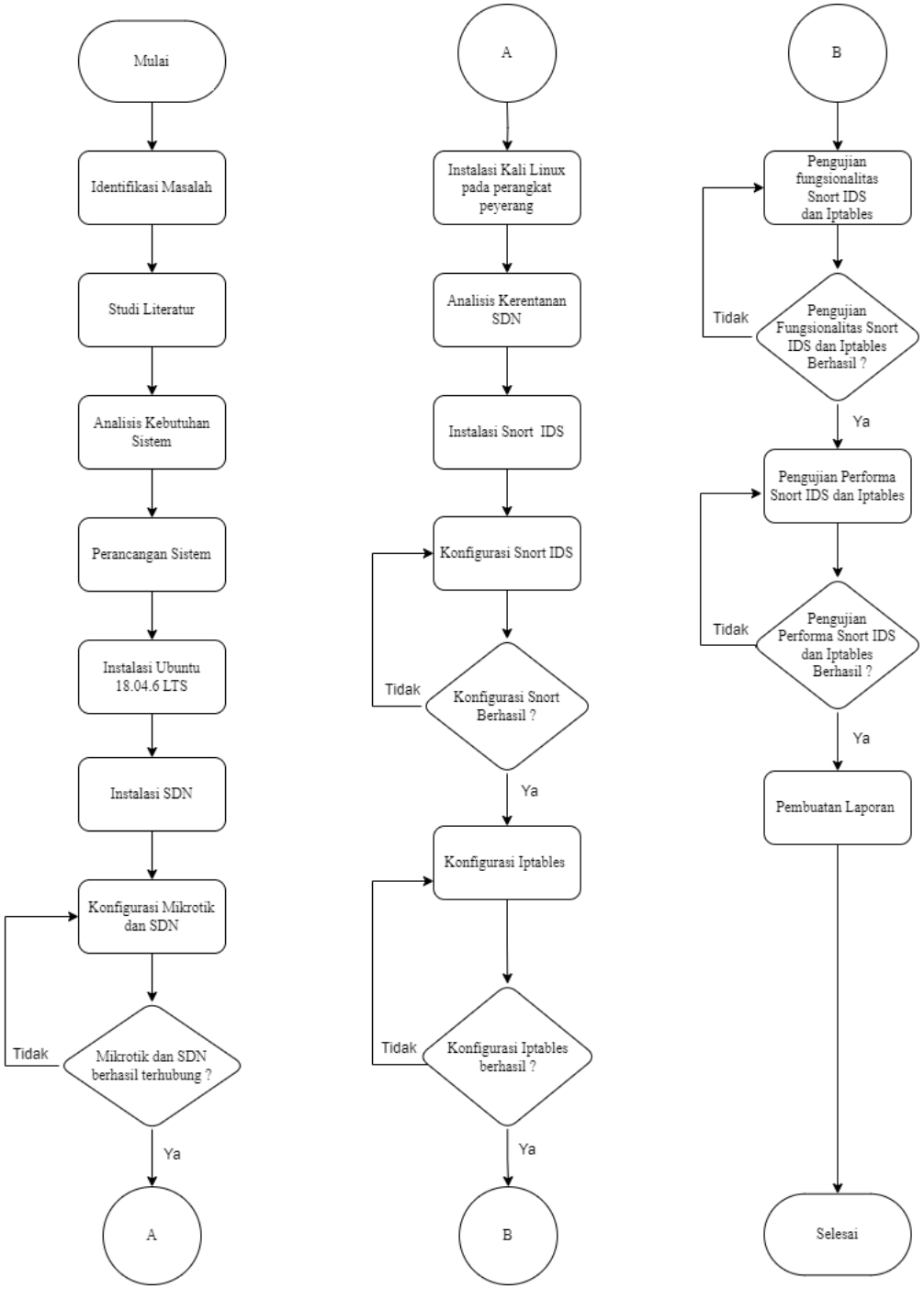
Beberapa tahun terakhir, teknologi *Software Defined Network* telah menjadi salah satu topik utama dalam pengembangan jaringan. *Software Defined Network* atau disingkat SDN merupakan paradigma baru dalam sistem jaringan komputer. Konsep SDN adalah memisahkan *control plane* dan *data plane* [1]. Dengan memisahkan *control plane* dan *data plane*, logika kontrol diimplemestasikan dalam *contoller*, sedangkan *switch* bertindak sebagai *forwarding device* yang dihubungkan oleh protokol OpenFlow [2]. OpenFlow adalah protokol utama yang digunakan pada jaringan SDN. OpenFlow merupakan salah satu jenis dari API dalam jaringan SDN yang digunakan untuk mengontrol atau mengatur *traffic flows* pada *switch* yang disebut dengan *southbound interfaces* [3]. Teknologi SDN ini masih terus dalam tahap pengujian dan pengembangan sehingga diperlukan perhatian lebih pada sisi keamanan.

Arsitektur jaringan SDN menawarkan keunggulan kontrol terpusat melalui *controller*. *Controller* pada SDN berfungsi sebagai otak atau pusat kontrol dari seluruh jaringan yang memungkinkan pengelolaan dan pengaturan jaringan secara terpusat. Namun, hal tersebut juga dapat menjadi kerentanan tersendiri apabila antara *control plane* pada *controller* dan *data plane* pada *switch* mengalamai serangan ataupun gangguan keamanan [4]. Salah satu serangan jaringan yang dapat menyerang antara *control plane* dan *data plane* yaitu DDoS (*Distributed Denial of Service*). DDoS (*Distributed Denial of Service*) adalah serangan yang dilakukan terhadap sebuah sistem komputer atau jaringan dengan cara membanjiri *resource* yang ada sehingga membuat sistem tersebut tidak lagi dapat merespon permintaan dari pengguna yang sah [5] [6] [7].

Pada penelitian ini dibuat sebuah sistem untuk mengatasi kerentanan keamanan antara *control plane* dan *data plane* dalam arsitektur jaringan *Software Defined Network* (SDN) terhadap serangan DDoS [8] [9] [10]. Pendekatan utama melibatkan pengembangan mekanisme deteksi dan mitigasi DDoS yang terintegrasi secara langsung dengan controller SDN, serta penguatan keamanan pada interaksi antara control plane dan data plane melalui implementasi *firewall* pada server [11] [12].

## II. METODE PENELITIAN

Penelitian ini dilakukan dengan empat tahapan. Pertama yaitu tahap perancangan, tahap perancangan merupakan tahap pertama yang dilakukan dalam penelitian ini dengan melakukan perancangan dan desain sistem yang sesuai untuk penelitian. Kedua, tahap instalasi dan konfigurasi. Pada tahap ini dilakukan instalasi dan konfigurasi perangkat lunak dan keras untuk menunjang penelitian. Ketiga, tahap pengujian. Pengujian dilakukan setelah instalasi dan konfigurasi SDN, Snort, dan Iptables sudah berjalan dengan baik. Kemudian dilakukan percobaan serangan untuk menganalisis fungsionalitas serta performa Snort dan Iptables. Metode penelitian yang dilakukan dalam penelitian ini dapat dilihat pada bagan alir yang ditunjukkan pada gambar 1.



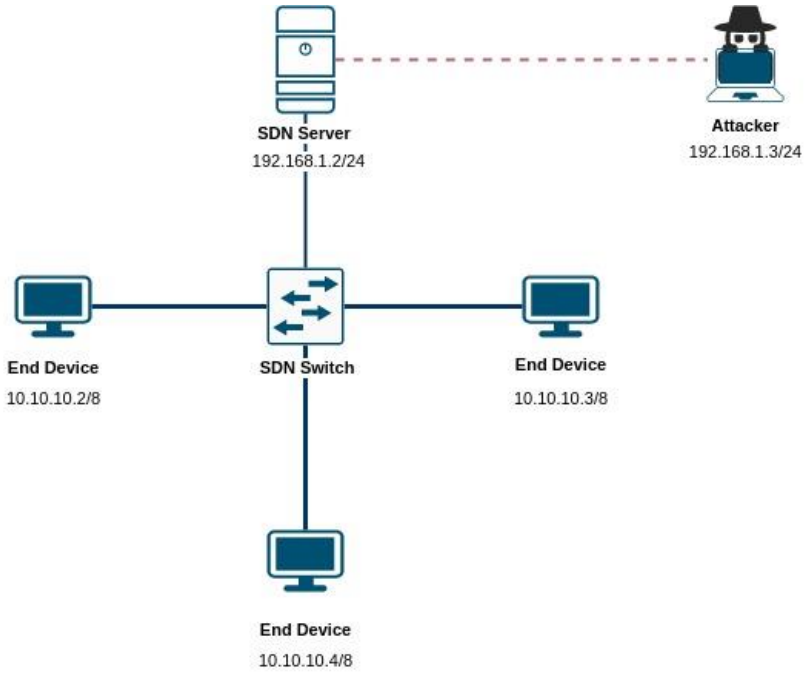

Gambar 1 Diagram alir penelitian

## III. HASIL DAN PEMBAHASAN

### *A. Penerapan Software Defined Network*

*Software Defined Network* diimplementasikan dengan menggunakan satu *router* MikroTik RB951-2nD yang digunakan sebagai *openswitch*, tiga komputer sebagai *end device*, dan satu laptop sebagai *controller* SDN. *Router* MikroTik RB951-2$^{nD}$ telah *support* terhadap protokol *openflow* sehingga cocok digunakan untuk penelitian ini. Gambar 2 merupakan hasil dari implementasi jaringan *software defined network*.

*p-ISSN : 2443-2210*
*e-ISSN : 2443-2229*




Gambar 2 Tampilan topologi jaringan SDN

*B. Analisis Kerentanan Sistem*

*Software defined network* berhasil diimplementasikan dan dipastikan konfigurasi berjalan dengan baik, maka dapat dilakukan analisis kerentanan pada sistem. Analisis kerentanan pada sistem dapat menggunaakan *tools* Nmap pada *metasploit framework* yang ada pada Kali linux. Kali linux berada satu *network* dengan SDN *controller*. Setelah Nmap dijalankan, terlihat beberapa *port* terbuka yang dapat menjadi sumber kerentanan pada sistem SDN ini.

Pertama, terdapat *port* 1099 yang digunakan oleh layanan RMI (*Remote Method Invocation*) *Registry* dalam komunikasi jaringan. Kedua, terdapat *port* 9876 yang digunakan untuk berkomunikasi antar *controller* pada jaringan SDN. Ketiga, terdapat *port* 8181 yang digunakan untuk mengakses web dari ONOS. Melalui *port* 8181, ONOS menyediakan antarmuka web yang memungkinkan pengguna untuk berinteraksi dengan sistem, mengelola jaringan, dan mengonfigurasi perangkat jaringan yang terhubung. Antarmuka web ini dapat diakses melalui browser web dengan mengunjungi alamat IP atau nama host ONOS yang sesuai, diikuti oleh *port* 8181. Ketiga *port* tersebut dilakukan penyerangan atau ekploitasi untuk memastikan adanya kerentanan pada sistem SDN ini.

Penyerangan pertama yaitu pada *port* 1099 dilakukan dengan menggunakan *metasploit framework*. Eksploitasi dimulai dengan menjalankan perintah *Started reverse TCP handler on 192.168.1.3:4444* yang mana penanganan TCP terbalik telah dimulai pada alamat IP 192.168.1.3 dengan *port* 4444. Kemudian, server memulai mengirimkan RMI *Header* dan *Call* untuk melakukan eksploitasi. Hasil ekploitasi menunjukan bahwa *port* tersebut gagal untuk di eksploitasi. Kegagalan tersebut karena versi JDK pada sistem sudah versi terbaru, yaitu versi 7. Pada versi tersebut JDK sudah dapat mencegah adanya eksploitasi RMI. Gambar 3 menunjukkan hasil dari eksploitasi *port* 1099.

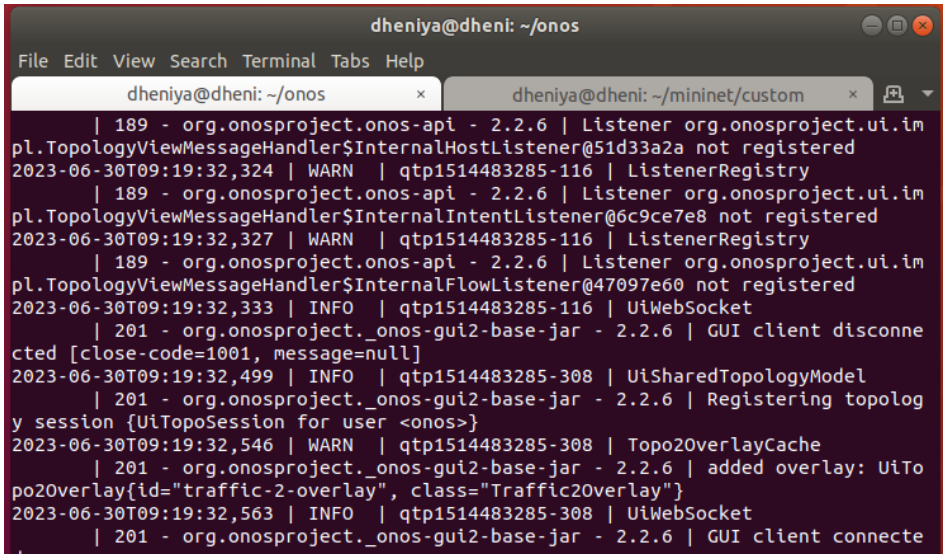
```
msf6 exploit(multi/misc/java_rmi_server) > run

[*] Started reverse TCP handler on 192.168.1.3:4444
[*] 192.168.1.2:1099 - Using URL: http://192.168.1.3:8080/7CO3Yws
[*] 192.168.1.2:1099 - Server started.
[*] 192.168.1.2:1099 - Sending RMI Header ...
[*] 192.168.1.2:1099 - Sending RMI Call ...
[-] 192.168.1.2:1099 - Exploit failed [not-vulnerable]: RuntimeError Exploit
aborted due to failure not-vulnerable The RMI class loader is disabled
[*] 192.168.1.2:1099 - Server stopped.
[*] Exploit completed, but no session was created.
msf6 exploit(multi/misc/java_rmi_server) >
```

Gambar 3 Hasil eksploitasi *port* 1099

Penyerangan kedua yaitu *port* 9876 dilakukan dengan mengirimkan DDoS pada *port* 9876 yang mana menunjukkan tidak ada perubahan apapun pada sistem SDN. Hal ini dikarenakan pada penelitian ini hanya menggunakan satu *controller* sehingga serangan DDoS pada port 9876 tidak berefek pada sistem. Gambar 4 menunjukkan hasil dari eksploitasi *port* 9876 yang mana system berjalan dengan normal.

.



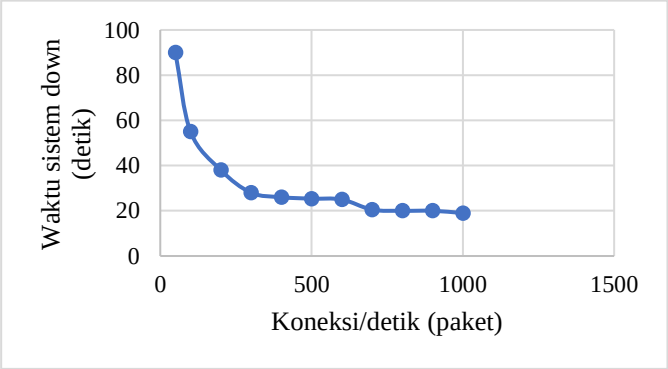

Gambar 4 Hasil eksploitasi *port* 9876

Penyerangan ketiga yaitu pada *port* 8181 menggunakan *tools* slowhttptest. slowloris, dan LOIC. Penyerangan dengan menggunakan slowhttptest menimbulkan efek pada SDN hingga menyebabkan sistem SDN down seperti yang ditunjukkan pada gambar 5.

TABEL 1
HASIL PENYERANGAN MENGGUNAKAN TOOLS SLOWHTTPTEST

| No | Jumlah Koneksi Keseluruhan (paket) | Koneksi/detik (paket) | Waktu Sistem Down (detik) |
|---|---|---|---|
| 1 | 10000 | 10 | Tidak down |
| 2 | 10000 | 50 | 90 |
| 3 | 10000 | 100 | 55 |
| 4 | 10000 | 200 | 38 |
| 5 | 10000 | 300 | 28 |
| 6 | 10000 | 400 | 26 |
| 7 | 10000 | 500 | 25,37 |
| 8 | 10000 | 600 | 25 |
| 9 | 10000 | 700 | 20,49 |
| 10 | 10000 | 800 | 20 |
| 11 | 10000 | 900 | 20 |
| 12 | 10000 | 1000 | 19 |

*p-ISSN : 2443-2210*
*e-ISSN : 2443-2229*




Gambar 5 Hasil eksploitasi *port* 8181

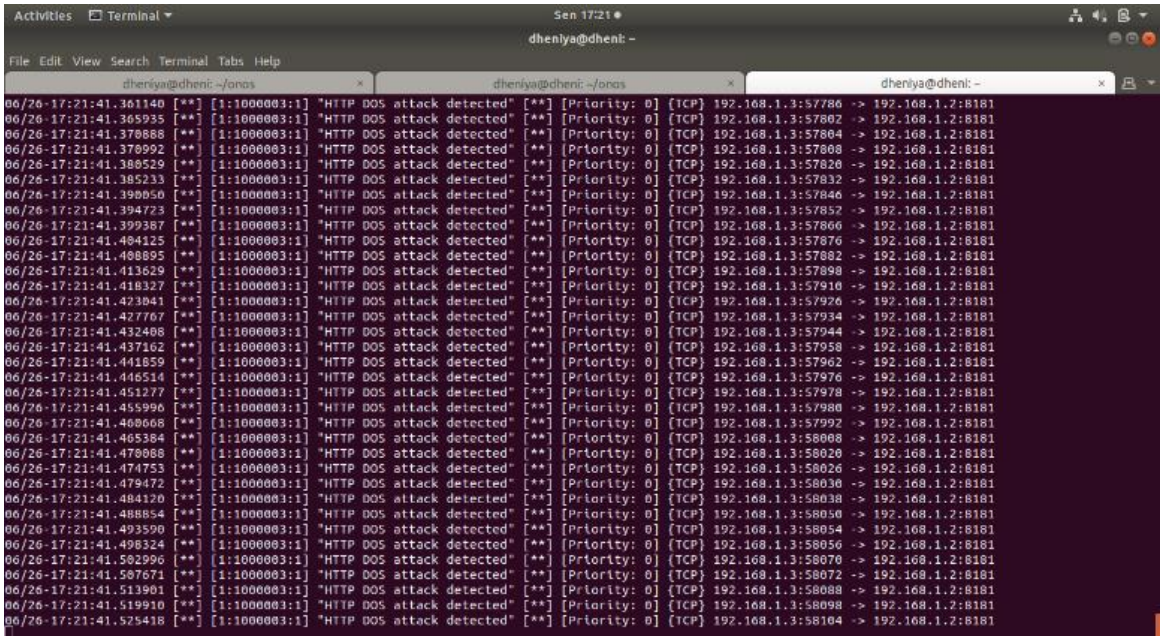
Berdasarkan data pada tabel 1 dapat diketahui bahwa penyerangan sistem dilakukan dengan mengirimkan serangan dengan jumlah 10000 koneksi. Penyerangan ini dilakukan selama 120 detik. Jumlah koneksi tiap detiknya dimulai dari 10 per detik hingga 1000 per detik. Terlihat bahwa semakin tinggi jumlah koneksi per detik, waktu sistem *down* cenderung lebih cepat. Hal ini dapat dianggap sebagai indikasi bahwa semakin tinggi beban koneksi, semakin cepat sistem akan mati. Waktu terlama sistem *down* ialah 90 detik dengan jumlah koneksi per detik 50 koneksi. Sedangakan waktu tercepat sistem *down* ialah 19 detik dengan jumlah koneksi per detik 1000 koneksi.

### *C. Pengujian Penerapan Snort IDS*

Snort IDS (*Intrusion Detection System*) memberikan solusi yang digunakan untuk mendeteksi adanya serangan DDoS. Pada penelitian ini, Snort IDS berhasil mendeteksi adanya serangan DDoS yang menyerang sistem SDN. Pengujian dilakukan untuk mengetahui fungsionalitas dan performa dari IDS ini.

#### *1. Pengujian Fungsionalitas*

Pengujian fungsionalitas digunakan untuk menguji kemampuan sistem dalam mendeteksi serangan. Pada pengujian ini dilakukan sepuluh kali percobaan dengan menggunakan tiga *tools* penyerangan, diantaranya slowhttptest, slowloris, dan LOIC. Slowhttptest, slowloris, dan LOIC merupakan jenis DDoS yang menyerang suatu server atau layanan sehingga server atau layanan tersebut menjadi tidak responsif atau tidak dapat diakses oleh pengguna yang sah. Gambar 6 merupakan *log* dari Snort IDS mendeteksi adanya serangan DDoS yang menyerang pada server atau layanan dengan IP 192.168.1.2 *port* 8181 dengan menggunakan koneksi TCP.




Gambar 6 *Log* Snort IDS

TABEL 2
HASIL SNORT DALAM MENDETEKSI SERANGAN

| No | Serangan Slowhttptest (paket) | Serangan Slowloris (paket) | Serangan LOIC (paket) | Berhasil Terdeteksi |
|---|---|---|---|---|
| 1 | 10000 | 10000 | 10000 | Ya |
| 2 | 10000 | 10000 | 10000 | Ya |
| 3 | 10000 | 10000 | 10000 | Ya |
| 4 | 10000 | 10000 | 10000 | Ya |
| 5 | 10000 | 10000 | 10000 | Ya |
| 6 | 10000 | 10000 | 10000 | Ya |
| 7 | 10000 | 10000 | 10000 | Ya |
| 8 | 10000 | 10000 | 10000 | Ya |
| 9 | 10000 | 10000 | 10000 | Ya |
| 10 | 10000 | 10000 | 10000 | Ya |

Pada pengujian yang dilakukan, sistem berhasil mendeteksi serangan DDoS pada berbagai *tools* penyerangan yang digunakan seperti yang ditunjukkan pada tabel 2. Pengujian ini dilakukan dengan mengirimkan 10000 koneksi selama 120 detik.

*2. Pengujian Performa*

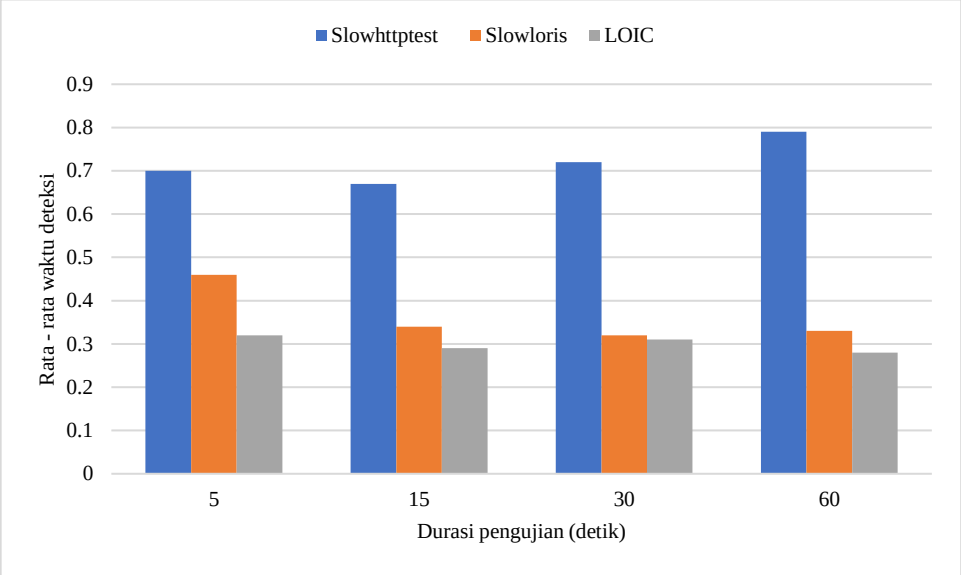
Pengujian performa pada Snort IDS dilakukan dengan pengujian waktu deteksi dan akurasi deteksi. Pengujian pertama yaitu pengujian waktu deteksi yang digunakan untuk mengukur seberapa cepat sistem dapat mengidentifikasi dan memberikan peringatan terhadap beberapa jenis serangan DDoS yang sedang terjadi. Pada pengujian ini sebanyak lima serangan dikirimkan pada setiap variasi waktu yang diujikan dengan menggunakan tiga *tools* penyerangan DDoS.

Berdasarkan tabel 3 dan gambar 7 didapatkan bahwa variasi durasi pengujian antara 5, 15, 30, dan 60 detik. Durasi pengujian mengindikasikan jangka waktu yang diamati dalam upaya mendeteksi serangan DDoS. Dalam pengujian ini,

seluruh serangan DDoS berhasil terdeteksi pada setiap pengamatan. Pengujian ini menggunakan tiga *tools* pemyerangan, yaitu slowhttptest, slowloris, dan LOIC. Pada penyerangan menggunakan slowhttptest, Rata-rata waktu deteksi cenderung stabil, dengan variasi yang relatif kecil antara 0,67 hingga 0,79 detik. Pada penyerangan menggunakan slowloris, menunjukkan sistem dapat mendeteksi cukup cepat dengan rentang antara 0,32 hingga 0,46 detik. Ada sedikit fluktuasi dalam waktu deteksi, tetapi perbedaannya tidak signifikan antara durasi pengujian yang berbeda. Penyerangan menggunakan LOIC menunjukkan waktu deteksi yang relatif rendah, dengan rentang antara 0,28 hingga 0,32 detik. Serupa dengan metode lainnya, Sistem juga menunjukkan waktu deteksi yang cukup cepat pada durasi pengujian yang berbeda.

TABEL 3
HASIL PENGUJIAN WAKTU DETEKSI

| Durasi Pengujian (detik) | Slowhttptest (detik) | Slowloris (detik) | LOIC (detik) |
|---|---|---|---|
| 5 | 0,7 | 0,46 | 0,32 |
| 15 | 0,67 | 0,34 | 0,29 |
| 30 | 0,72 | 0,32 | 0,31 |
| 60 | 0,79 | 0,33 | 0,28 |
| Rata -rata | 0,72 | 0,36 | 0,3 |




Gambar 7 Hasil pengujian waktu deteksi

Pengujian kedua, yaitu pengujian akurasi deteksi. Dalam pengujian ini, dilakukan pengiriman serangan DDoS sebanyak 5000 koneksi dan dilakukan analisis terhadap akurasi deteksi. Pengujian dilakukan sebanyak lima kali. Data yang ditampilkan mencakup jumlah koneksi yang dikirim, jumlah koneksi yang terdeteksi, dan persentase akurasi deteksi. Untuk menguji akurasi ini, rumus yang dapat digunakan adalah:

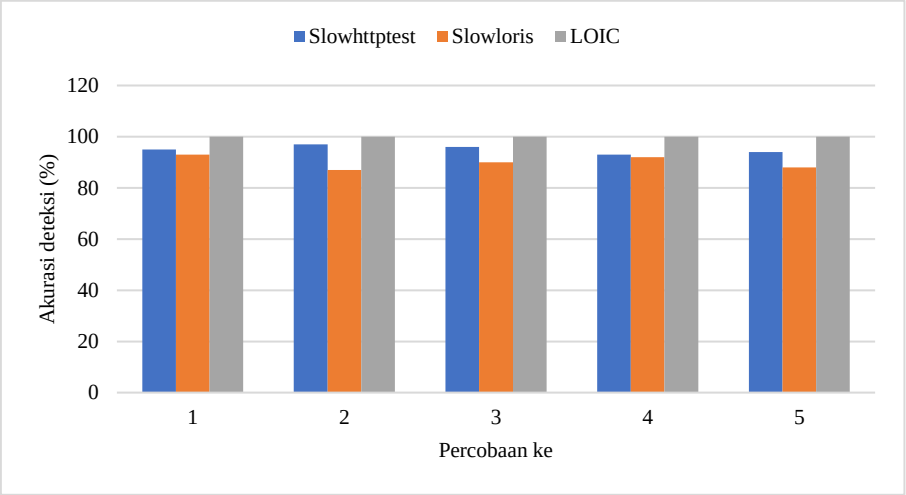
$$\textbf{Akurasi deteksi} = \frac{\textbf{Paket yang terdeteksi}}{\textbf{Paket yang dikirim}} \textbf{x 100\%} \quad (1)$$

Paket yang terdeteksi adalah jumlah koneksi yang berhasil terdeteksi sebagai serangan DDoS oleh sistem deteksi. Paket yang dikirim adalah jumlah total koneksi yang dikirim selama pengujian. Dengan menggunakan rumus tersebut, dapat dilakukan penghitungan persentase akurasi deteksi untuk setiap pengujian yang dilakukan dan mendapatkan gambaran tentang sejauh mana sistem deteksi mampu mengenali serangan DDoS yang dikirimkan. Hasil dari pengujian akurasi deteksi dengan mengguanakan tiga *tools* penyerangan ditunjukkan tabel 4 seperti berikut.

TABEL 4
HASIL PENGUJIAN AKURASI DETEKSI

| Percobaan ke - | Slowhttptest (%) | Slowloris (%) | LOIC (%) |
|---|---|---|---|
| 1 | 95 | 93 | 100 |
| 2 | 97 | 87 | 100 |
| 3 | 96 | 90 | 100 |
| 4 | 93 | 92 | 100 |
| 5 | 94 | 88 | 100 |
| Rata – rata | 95 | 90 | 100 |

Bedasarkan tabel 4 dan gambar 8 didapatkan bahwa hasil akurasi sistem dalam mendeteksi serangan DDoS dengan mengguanakan alat penyerangan slowhttptest, slowloris, dan LOIC. Angka-angka dalam tabel mewakili persentase akurasi deteksi untuk setiap percobaan yang dilakukan. Penyerangan pertama dilakukan dengan menggunakan slowhttptest, sistem mampu menunjukkan tingkat akurasi deteksi yang cukup tinggi dengan angka yang bervariasi antara 93% hingga 97%. Secara keseluruhan, akurasi deteksi dengan menggunakan alat penyerangan ini mencapai akurasi yang konsisten dalam mendeteksi serangan DDoS. Penyerangan kedua dilakukan dengan menggunakan slowloris, sistem mampu menunjukkan tingkat akurasi deteksi yang baik dengan angka yang bervariasi antara 87% hingga 92%. Meskipun ada sedikit variasi dalam hasilnya, sistem secara umum masih mencapai tingkat akurasi yang cukup tinggi. Penyerangan ketiga dilakukan dengan menggunakan LOIC, sistem mencapai tingkat akurasi deteksi yang sempurna dengan 100% untuk semua percobaan. Hal ini menunjukkan bahwa Snort mampu mendeteksi serangan DDoS dengan akurasi yang sangat tinggi. Secara keseluruhan, dengan beberapa alat penyerangan DDOS yang digunakan (Slowhttptest, Slowloris, dan LOIC) Snort mampu menunjukkan tingkat akurasi yang baik dalam mendeteksi serangan DDoS.




Gambar 8 Hasil pengujian akurasi deteksi

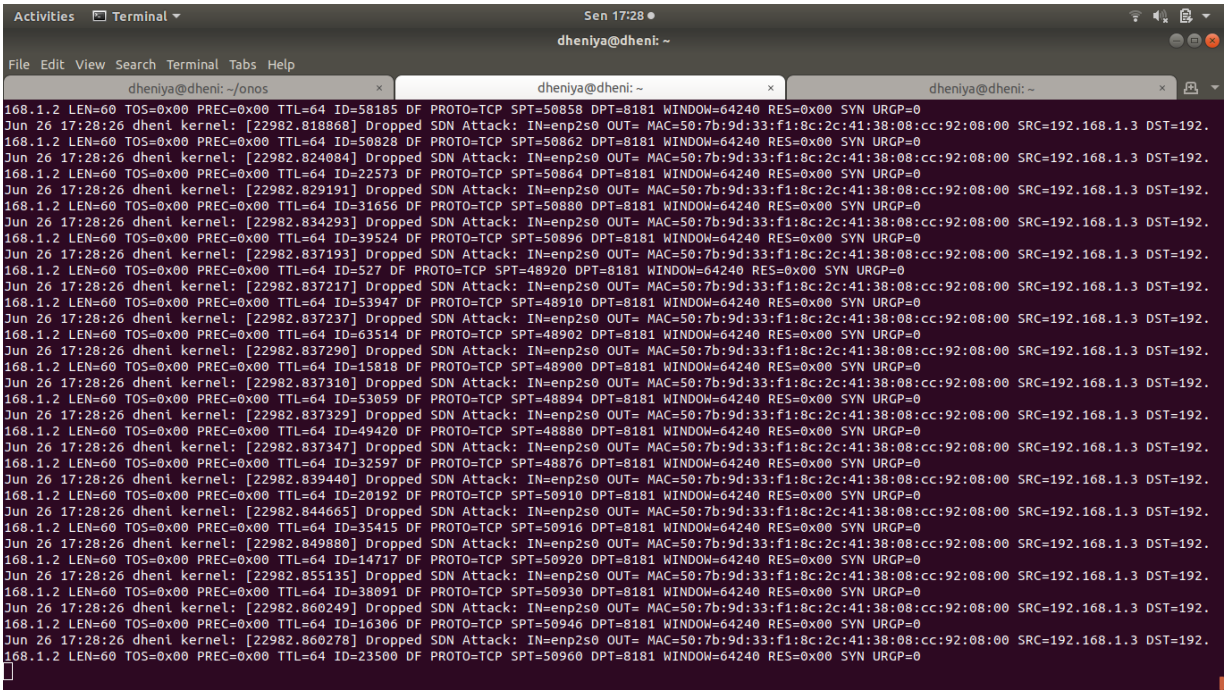
*D. Pengujian Penerapan Iptables*

Iptables merupakan sistem *firewall* yang ada pada sistem operasi linux. Sistem ini digunakan untuk melakukan mitigasi pada serangan DDoS dengan cara membatasi akses dan memblokir serangan tersebut. Pengujian pada sistem ini dilakukan untuk mengetahui fungsionalitas dan performa yang dimiliki.

*1. Pengujian Fungsionalitas*

Pengujian fungsionalitas digunakan untuk menguji kemampuan sistem dalam melakukan blok serangan. Pada pengujian ini dilakukan sepuluh kali percobaan dengan menggunakan tiga *tools* serangan yang berbeda pada setiap serangan seperti yang ditunjukan pada tabel 5. Pada pengujian yang dilakukan, sistem berhasil memblokir serangan DDoS dengan

menggunakan berbagai *tools* serangan DDoS seperti slowhttptest, slowloris, dan LOIC. Pengujian ini dilakukan dengan mengirimkan 10000 paket koneksi. Gambar 9 menunjukkan *log* yang dihasilkan oleh iptables ketika memblokir serangan DDoS. *Log* tersebut menunjukan paket berhasil dibuang sehingga tidak terjadi *flooding* paket pada sistem.




Gambar 9 Hasil Iptables dalam memblokir serangan

TABEL 5
HASIL PENGUJIAN FUNGSIONALITAS IPTABLES

| No | Serangan Slowhttptest (paket) | Serangan Slowloris (paket) | Serangan LOIC (paket) | Berhasil Terblokir |
|---|---|---|---|---|
| 1 | 10000 | 10000 | 10000 | Ya |
| 2 | 10000 | 10000 | 10000 | Ya |
| 3 | 10000 | 10000 | 10000 | Ya |
| 4 | 10000 | 10000 | 10000 | Ya |
| 5 | 10000 | 10000 | 10000 | Ya |
| 6 | 10000 | 10000 | 10000 | Ya |
| 7 | 10000 | 10000 | 10000 | Ya |
| 8 | 10000 | 10000 | 10000 | Ya |
| 9 | 10000 | 10000 | 10000 | Ya |
| 10 | 10000 | 10000 | 10000 | Ya |

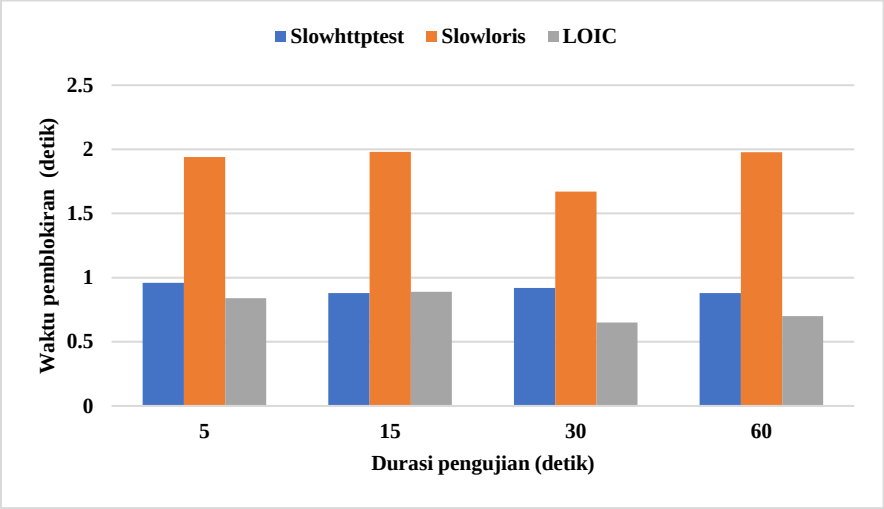
*2. Pengujian Performa*

Pengujian performa dilakukan dengan pengujian waktu pemblokiran paket serangan DDoS dan banyaknya koneksi yang dapat diterima server. Pengujian pertama yaitu pengujian waktu pemblokiran yang digunakan untuk mengukur seberapa cepat sistem dapat mengidentifikasi dan memblokir serangan DDoS sebelum membuat sistem SDN *down*. Pada pengujian ini sebanyak lima serangan dikirimkan pada setiap variasi waktu yang diujikan dengan menggunakan tiga *tools* serangan.

Berdasarkan tabel 6 dan gambar 10 didapatkan bahwa variasi durasi pengujian antara 5, 15, 30, dan 60 detik. Durasi pengujian mengindikasikan jangka waktu yang diamati dalam upaya memblokir serangan DDoS. Dalam pengujian ini, seluruh serangan DDoS berhasil diblokir pada setiap pengamatan. Pengujian ini menggunakan tiga *tools* penyerangan, yaitu slowhttptest, slowloris, dan LOIC. Pada penyerangan menggunakan slowhttptest, rata-rata waktu blokir cenderung stabil, antara 0,88 hingga 0,98 detik. Rata-rata waktu blokir yang relatif singkat, yaitu sekitar 0,91 detik, menunjukkan respon yang efektif terhadap serangan tersebut. Pada penyerangan menggunakan slowloris, sistem membutuhkan waktu yang lebih lama dalam memblokir serangan DDoS dibandingkan dengan *tools* penyerangan slowhttptest. Rata-rata waktu blokir slowloris berkisar antara 1,67 hingga 1,98 detik. Penyerangan menggunakan LOIC menunjukkan waktu blokir yang relatif cepat, dengan rentang antara 0,65 hingga 0,89 detik. Serupa dengan metode lainnya, Sistem mampu melakukan pemblokiran serangan DDoS pada waktu yang cukup cepat sehingga *avaibility* sistem SDN terjaga.

TABEL 6
HASIL PENGUJIAN WAKTU PEMBLOKIRAN SERANGAN

| Durasi Pengujian (detik) | Slowhttptest (detik) | Slowloris (detik) | LOIC (detik) |
|---|---|---|---|
| 5 | 0,96 | 1,94 | 0,84 |
| 15 | 0,88 | 1,98 | 0,89 |
| 30 | 0,92 | 1,67 | 0,65 |
| 60 | 0,88 | 1,97 | 0,7 |
| Rata - rata | 0,91 | 1,89 | 0,77 |




Gambar 10 Hasil pengujian waktu pemblokiran serangan

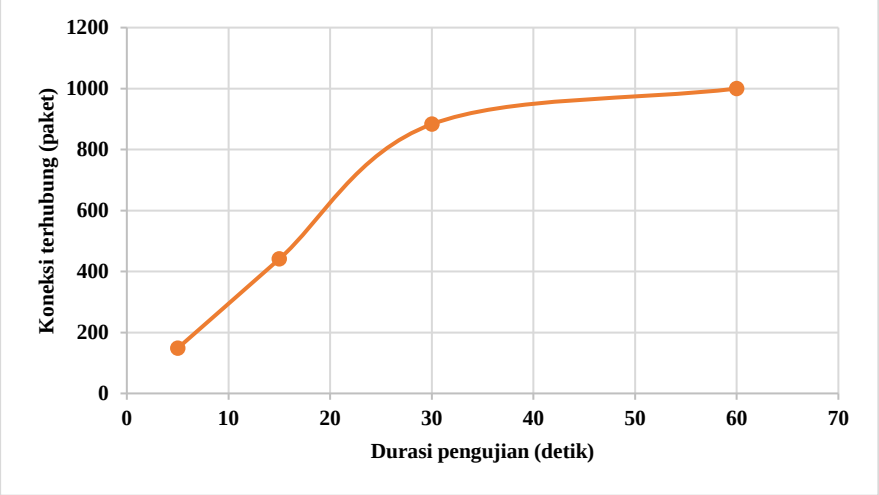
Pengujian kedua yaitu pengujian dengan menghitung seberapa banyak koneksi yang dapat diterima server. Pengujian ini dilakukan dengan menggunakan variasi waktu yang berbeda – beda. Tabel 7 merupakan hasil dari pengujian tersebut.

*p-ISSN : 2443-2210*
*e-ISSN : 2443-2229*

TABEL 7
BANYAKNYA KONEKSI YANG DAPAT DITERIMA SERVER

| Durasi Pengujian (detik) | Koneksi Terhubung (koneksi) |
|---|---|
| 5 | 148 |
| 15 | 441 |
| 30 | 883 |
| 60 | 1000 |

Berdasarkan tabel 7 dan gambar 11 terlihat adanya peningkatan jumlah koneksi terhubung seiring dengan bertambahnya durasi pengujian. Pada awal pengujian, durasi 5 detik, terdapat 148 koneksi terhubung. Namun, seiring dengan bertambahnya waktu hingga 60 detik, jumlah koneksi terhubung mengalami peningkatan yang signifikan, mencapai puncaknya pada durasi 60 detik dengan 1000 koneksi terhubung. Hal ini mengindikasikan bahwa sistem mampu membatasi koneksi agar SDN tidak *down*. Peningkatan jumlah koneksi terhubung seiring dengan durasi pengujian yang lebih lama menunjukkan kemampuan sistem dalam menangani dan memproses volume koneksi yang lebih besar sehingga meningkatkan efektivitas dan mengurangi kemungkinan pemblokir koneksi yang sah.




Gambar 11 Banyaknya koneksi yang dapat diterima server

## IV. SIMPULAN

Implementasi Snort pada sistem SDN mampu mendeteksi serangan DDoS dengan akurasi mencapai 95% serangan slowhttptest, 90% serangan slowloris dan 100% serangan LOIC. Rata-rata penggunaan waktu yang diperlukan untuk mendeteksi adanya serangan slowhttptest sebesar 0,72 detik, serangan slowloris sebesar 0,36 detik, dan serangan LOIC sebasar 0,3 detik. Implementasi iptables pada sistem SDN mampu memblokir serangan DDoS dengan rata – rata waktu pemblokiran 0.91 detik terhadap serang slowhttptest, 1,89 detik terhadap serangan slowloris, 0,77 detik terhadap serangan LOIC, dan sistem mampu mengelola volume koneksi yang besar sehingga mampu menjaga ketersediaan sistem SDN. Pada penelitian sebelumnya[8], sistem hanya mampu mendeteksi serangan DDoS. Dengan adanya penelitian ini, sistem mampu menghentikan serangan DDoS dan memberikan *avaibility* pada SDN. Akan tetapi, penelitian ini memiliki kelemahan yang mana pengujian hanya dilakukan dengan tiga *tools* penyerangan DDoS. Peneliti berharap, penelitian ini dapat lebih dikembangkan lagi oleh peneliti – peneliti selanjutnya.

## UCAPAN TERIMA KASIH

Kami ingin mengucapkan terima kasih kepada semua pihak yang telah membantu dan mendukung penelitian ini. Tanpa bantuan mereka, penyelesaian penelitian ini tidak akan terwujud. Pertama-tama, kami berterima kasih kepada Universitas Gadjah Mada atas izin dan dukungannya untuk melaksanakan penelitian ini. Terima kasih juga kami sampaikan kepada Dosen Pembimbing Bapak Dr. Ronald Adrian, S.T., M. Eng. yang telah membimbing kami dengan penuh semangat. Kami juga ingin menyampaikan apresiasi kepada rekan-rekan sejawat yang telah memberikan masukan dan kritik yang membangun selama proses penelitian ini. Kontribusi mereka sangat berarti bagi perbaikan dan peningkatan kualitas penelitian ini.Tak lupa, ucapan terima kasih kami tujukan kepada keluarga dan teman-teman yang selalu memberikan dukungan moral dan semangat kepada kami dalam menghadapi tantangan penelitian ini. Semoga hasil penelitian ini dapat memberikan manfaat dan kontribusi positif bagi pengembangan teknologi di masa depan.

## DAFTAR PUSTAKA

[1] N. Ashodia and K. Makadiya, "Detection and Mitigation of DDoS attack in Software Defined Networking: A Survey," *International Conference on Sustainable Computing and Data Communication Systems (ICSCDS)*, pp. 1175-1180, 2022.

[2] S. Muzafar, N. Jhanjhi, N. A. Khan and F. Ashfaq, "DDoS Attack Detection Approaches in on Software Defined Network," *14th International Conference on Mathematics, Actuarial Science, Computer Science and Statistics (MACS)*, pp. 1-5, 2022.

[3] S. Hamid, N. Z. Bawany and J. A. Shamsi, "ReCSDN: Resilient Controller for Software Defined Networks," *International Journal of Advanced Computer Science and Applications*, vol. 8, no. 8, pp. 202-208, 2017.

[4] M. Shaikh, F. Y. Khuhawar, K. Nisar, A. A. Memon and A. S. Khan, "Vulnerability Assesment & Analysis of Sofware Defined Networking using a Virtual Testbed," *Global Conference on Wireless and Optical Technologies (GCWOT)*, pp. 1-7, 2022.

[5] S. Badotra and S. N. Panda, "SNORT based Early DDoS detection system using Opendaylight and open networking operating system in software defined networking," *Cluster Computing*, vol. 24, pp. 501-513, 2020.

[6] B. Habib, F. Khuurshid, A. H. Dar and Z. Shah, "DDoS Mitigation in Eucalyptus Cloud Platform Using Snort and Packet Filtering - IP Tables," *4th International Conference on Information Systems and Computer Networks (ISCON)*, pp. 546-550, 2019.

[7] S. Raj and N. K. Walia, "A Study on Metasploit Framework: A Pen-Testing Tool," *International Conference on Computational Performance Evaluation (ComPE)*, pp. 296-302, 2020.

[8] D. N. M. R. Varre and J. Bayana, "A Secured Botnet Prevention Mechanism for HTTP Flooding Based DDoS Attack," *3rd International Conference for Emerging Technology (INCET)*, pp. 1-5, 2022.

[9] A. Khraisat, I. Gondal, P. Vamplew and J. Kamruzzaman, "Survey of intusion detection system: techniques, datasets and challenges," *Cybersecurity*, vol. 2, no. 20, 2019.

[10] P. Oktivasari, A. R. Zain, M. Agustin, A. Kurniawan, F. A. Murad and M. F. Anshor, "Analysis of Effectiveness of Iptables on Web Server from Slowloris Attack," *5th International Conference of Computer and Informatics Engineering (IC2IE)*, pp. 215-219, 2022.

[11] N. Khamphakdee, N. Benjamas and S. Saiyod, "Improving Intrusion Detection System Based on Snort Rules for Network Probe Attacks Detection with Association Rules Technique of Data Mining," *Journal of ICT Research and Applications*, vol. 8, no. 3, pp. 234-250, 2015.

[12] K. K. A. Marta, I. N. B. Hartawan and I. K. S. Satwika, "Analisis Sistem Monitoring Keamanan Server Dengan SMS Alert Berbasis Snort," *Information System and Emerging Technology Journal*, vol. 1, no. 1, 2020.