JUMANJI | Vol. 02 | No. 01 | Halaman 1 - 16
ISSN (p): 2598-8050 April 2018
ISSN (e): 2598-8069

Perancangan *Intrusion Prevention System* pada Jaringan Software Defined Networks

MUHAMMAD ARIEF NUGROHO[1], NOVIAN ANGGIS SUWASTIKA[2]

[1,2] School of Computing, Telkom University
arif.nugroho@telkomuniversity.ac.id

ABSTRAK

Keamanan jaringan telah menjadi bagian penting dalam implementasi di dalam jaringan Software Defined Network (SDN). Menyelenggarakan jaringan aman di dalam jaringan SDN merupakan tantangan tersendiri karena bagaimana sebuah perangkat jaringan dapat mampu mengenali, mengidentifikasi, dan mencegah serangan di level perangkat jaringan bukan di level host. Intrusion Prevention System (IPS) memberikan kemampuan untuk mengenali, mengidentifikasi, dan mencegah serangan yang terjadi secara otomatis. Integrasi IPS ke dalam jaringan SDN memberikan keuntungan bahwa administrator dapat mengatur dan memonitor keamanan jaringan secara terpusat. Dari hasil pengujian yang telah dilakukan, integrasi IPS ke dalam jaringan SDN mampu mendeteksi dan mencegah serangan yang terjadi dalam jaringan SDN sesuai dengan rule yang terdefinisi dalam IPS. Namun, terjadi penurunan kinerja throughput, delay, dan jitter di dalam jaringan SDN. Hal ini terjadi karena setiap paket yang melewati perangkat jaringan harus melewati proses pengecekan rule di dalam IPS.

Kata kunci: *Keamanan Jaringan, Software Defined Networks, Intrusion Prevention System, Througput, Delay, Jitter.*

ABSTRACT

Network Security become important part when implementing Software Defined Netwowk. Providing secure networks in SDN became more challenging because network devices have to detecting, identifying and preventing network attacks, compared to traditional architecture that need special hardware to do that job. Intrusion Prevention System (IPS) is a method for detect, identify, prevent network attacks automatically. The integration of IPS into SDN networks provides advantage that administrators can organize and monitor network security centrally. Based on experiment result, the integration of IPS into SDN networks able to detect, identify, and prevent attacks that occur in the SDN network based on rule defined in IPS. But, it has drawbacks that network performance including throughput, delay, and jitter decreased. This happens because every packet that passes through the network device must be checked by SDN controller and IPS system.

Keywords*: Network Security, Software Defined Networks, Intrusion Prevention System, throughput, delay, jitter.*

1. PENDAHULUAN

Software Defined Network (SDN) merupakan pendekatan baru di dalam jaringan komputer yang digunakan untuk mengelola jaringan secara terpusat. SDN memungkinkan jaringan komputer dapat dikonfigurasi dengan konfigurasi yang sama meskipun menggunakan vendor/ perangkat yang berbeda, dimana masing masing vendor memiliki konfigurasi dan *syntax* yang berbeda dengan vendor lainnya **(Nadeau & Gray, 2013)**. Implementasi SDN di Telkom University mempermudah administrator jaringan dalam mengelola jaringan. Saat ini jaringan di Telkom University menggunakan perangkat dari cisco, juniper, HP Procurve dan Ryujie. Penerapan SDN di Telkom University dapat mempermudah administrator jaringan dalam mengelola jaringan karena *network administrator* tidak perlu melakukan konfigurasi yang berbeda-beda sesuai dengan vendor, namun cukup satu kali konfigurasi melalui SDN. *Network administrator* dapat mengelola jaringan SDN melalui bantuan *controller.* SDN *controller* menyediakan *interface* antara perangkat dan administrator untuk melakukan konfigurasi/ modifikasi kerja *switch* pada jaringan **(Lim, Ha, Kim, Kim, & Yang, 2014)**. Keamanan jaringan merupakan faktor yang sangat penting dalam menyediakan layanan jaringan kepada *user* selain *network availability*.

Saat ini sering terjadi aktivitas hacking di jaringan SDN seperti *port scanning*, *web deface*, *brute force password*, *sql injection*, *Denial of Service*, *virus*, dll. Jenis serangan tersebut sebagian besar tidak dapat tertangani dengan baik di jaringan SDN. Hal tersebut dikarenakan kemampuan firewall yang terdapat di controller SDN tidak dapat mendeteksi serangan di layer *application*. Kemampuan firewall di SDN hanya mampu mendeteksi serangan hingga di layer *transport*. Kelemahan firewall di jaringan SDN dapat diatasi dengan menggunakan integrase Intrusion Prevention System pada jaringan SDN.. IPS **(Thomas, 2005)** merupakan suatu sistem yang mampu mendeteksi aktifitas mencurigakan pada jaringan dan kemudian melakukan pencegahan dini terhadap serangan atau kejadian yang dapat membuat jaringan menjadi tidak berjalan sebagaimana mestinya. IPS dibagi menjadi dua yaitu *Host-based Intrusion Prevention System* (HIPS) dan *Network-based Intrusion Prevention System* (NIPS). Untuk skala *host,* HIPS dapat di implementasikan dengan mudah, administrator harus menginstall HIPS di masing masing *host*. Namun kompleksitas akan bertambah jika jumlah *host* juga bertambah, karena seorang administrator harus menginstall HIPS di masing masing *host*. Jika dalam satu jaringan terdapat 254 *host*, tentu administrator harus mengkonfigurasi HIPS ke 254 *host* tersebut. HIPS tidak cocok diimplementasikan di level jaringan. Untuk level jaringan, NIPS lebih cocok diimplementasikan. Administrator hanya tinggal mengkonfigurasi NIPS di sisi *gateway/ firewall* jaringan, dan semua jaringan akan dapat termonitor dengan mudah.

2. KAJIAN PUSTAKA

(Zhengyang, 2014) menjelaskan bahwa pengembangan *framework* SDN berbasis *Intrusion Prevention System* (IPS) dalam lingkungan cloud computing. Penelitian ini menggambarkan *framework* pengembangan pertahanan sistem berbasis SDN secara efektif dan efisien menggunakan metode deteksi, modul analisis dan cara memitigasinya.

(Zhiyuan, 2015) Memberikan gambaran bagaimana administrator jaringan dapat mengkonfigurasi sumber daya jaringan secara cepat, agar dapat menyesuaikan diri dengan *traffic flow* jaringan yang luas. Dalam perkembanganya, aspek keamanan menjadi hal yang patut menjadi perhatian. Beragamnya jenis serangan yang berkembang menjadi pendorong bagi dibentuknya arsitektur keamanan bagi sistem berbasis SDN. Penelitian tersebut dilakukan untuk mendesain sebuah solusi keamanan dengan membangun sebuah arsitektur

yang menyediakan layanan keamanan dengan menegakkan *network policy* secara benar dan menyediakan *network policy* yang terjamin keamanannya bagi sistem berbasis SDN.

(Lei, 2013) mengembangkan IPS di dalam jaringan SDN dengan metode sistem penjadwalan keamanan bagi alplikasi untuk keseluruhan jaringan dan juga metode *load balancing* antar IPS yang ada. Dari *test-bed* yang dilakukan, terbukti bahwa skema yang diterapkan akan mereduksi *network latency* secara signifikan.

3. METODOLOGI PENELITIAN

3.1 Dasar Teori

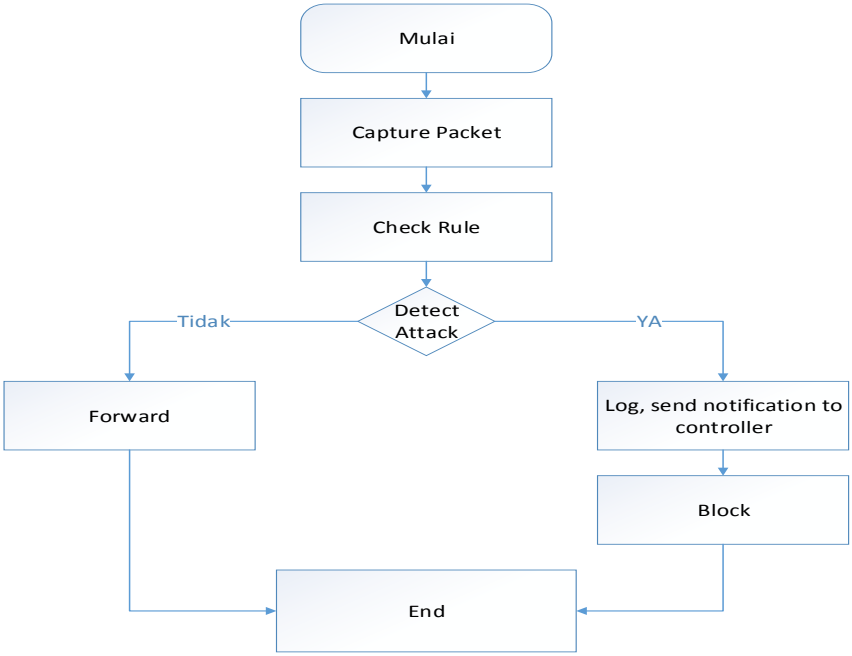
Software Defined Network (SDN) merupakan paradigma baru dalam jaringan yaitu memisahkan perangkat *control plane* dan *data plane*. Pemisahan kedua fungsi tersebut akan mempermudah dalam manajemen perangkat yang heterogen, inovasi, dan evolusi dalam jaringan. SDN memiliki sifat mudah dalam pengaturannya, murah, dan fleksibel serta cocok untuk kebutuhan jaringan saat ini yang membutuhkan *bit rate* yang tinggi untu pertukaran data. Konsep utama dalam SDN terdapat tiga yaitu *programmability*, pemisahan *controller* dengan *data plane,* serta pengelolaan jaringan yang terpusat **(Nadeau & Gray, 2013)**. Secara umum, SDN memisahkan *layer* infrastruktur dan *layer* control. Kecerdasan jaringan secara *logic* terpusat pada *controller* SDN yang mengelola seluruh jaringan. Dengan konsep tersebut, jaringan seakan seperti sebuah *logical switch* yang berisi aplikasi-aplikasi dan *policy*. Keuntungan mengunakan SDN, perusahaan bebas dari *control vendor* perangkat jaringan, sehingga menyederhanakan dan mempermudah perancangan dan pengoperasian perangkat. SDN juga menyederhanakaan perangkat jaringan itu sendiri dikarenakan perangkat tidak perlu mengerti proses dari ribuan standar protokol tetapi hanya menerima perintah dari *controller* SDN

Openflow merupakan protokol yang diciptakan oleh Standord University. Tujuan awal diciptakannya *openflow* adalah untuk mengganti fungsi dari layer 2 dan layer 3 pada *router* dan *switch* **(Nadeau & Gray, 2013)**. Saat ini *openflow* digunakan sebagai basis protokol yang digunakan dalam *Software Defined Network. Openflow* memungkinan *user* untuk mengakses *forwarding plane* pada *switch* maupun *router*. *Openflow* didesain untuk menyediakan aplikasi eksternal yang memiliki akses ke *forwarding plane* yang terdapat pada *switch* maupun *router*. *Openflow* diimplementasikan pada kedua sisi *interface* diantara perangkat infrastruktur jaringan dan SDN *controller software*. Selain itu, protokol *openflow* menyediakan perintah dasar untuk memodifikasi, mengatur rute, serta memblok aliran data yang ada di jaringan **(Lopez & Duarte, 2015)**

Intrusion Prevention System (IPS) merupakan perkembangan dari *Intrusion Detection System* dimana IPS merupakan suatu sistem (*hardware, software*, maupun kombinasi *hardware* dan *software)* yang memiliki kemampuan untuk memonitor jaringan dan kemudian melakukan tidakan pencegahan dari aktivitas mencurigakan di dalam jaringan **(Thomas, 2005)**. Terdapat dua bentuk dari IPS, yaitu *Network Based Intrusion Prevention System* (NIPS) dan *Host Based Intrusion Prevention System* (HIPS). NIPS digunakan untuk memantau aliran data yang keluar masuk jaringan dan biasanya diletakkan di depan atau dibelakang router, *firewall*, maupun VPN *gateway*. Sedangkan HIPS merupakan jenis IPS yang dipasang pada *host* untuk memantau aliran data yang terjadi pada *host* tersebut. Untuk mendeteksi dan mencegah usaha serangan ke dalam jaringan atau *host*, IPS menggunakan beberapa metode yaitu, *signature matching*, *protocol analysis*, dan *anomaly detection*.

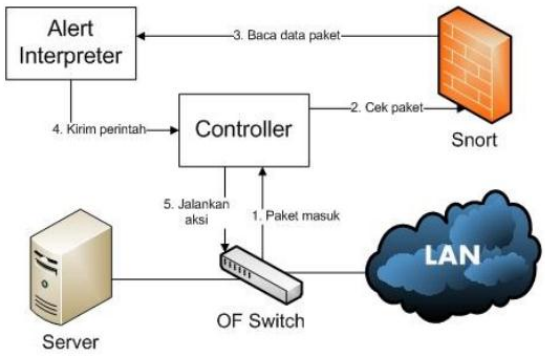
3.2 Perancangan Sistem

Secara umum, IPS akan diiimplementasikan pada jaringan SDN dengan tujuan untuk mendeteksi dan mencegah serangan yang terjadi pada jaringan SDN. IPS menggunakan *Snort* dan ditempatkan di SDN *controller.* Tujuan penempatan *snort* pada *controller* SDN bertujuan agar semua paket yang lewat di *controller* SDN dapat diteruskan dan dibaca oleh *Snort.* Jika terdapat paket data yang dianggap berbahaya, *snort* akan mencatat *log*, memberikan notifikasi ke SDN *controller*, kemudian SDN *controller* akan melakukan *blocking* paket tersebut ke tujuan. Secara umum, alur sistem digambarkan sebagai berikut :



Gambar 1. Flowchart IPS-SDN

Agar dapat melakukan paket, diperlukan *controller* SDN yang memiliki fitur *firewall* yaitu Ryu **(Yamahata, 2014)**. Controller Ryu memiliki fitur *firewall* yang dapat digunakan untuk melakukan pemblokiran dengan merubah *access control list* pada setiap *switch* yang terhubung kedalam jaringan SDN. Namun proses *blocking* tidak dapat berjalan secara otomatis karena *snort* mencatat *log* pada saat terjadi serangan di dalam jaringan. Agar proses *blocking* dapat berjalan secara otomatis, dibuat sebuah program *alert interpreter* yang yang berfungsi sebagai penerjemah *log* yang dihasilkan oleh *snort* dan kemudian memberikan perintah kepada *controller* untuk melakukan *blocking/ droping* paket. Alur program *alert interpreter* untuk *blocking* adalah sebagai berikut :



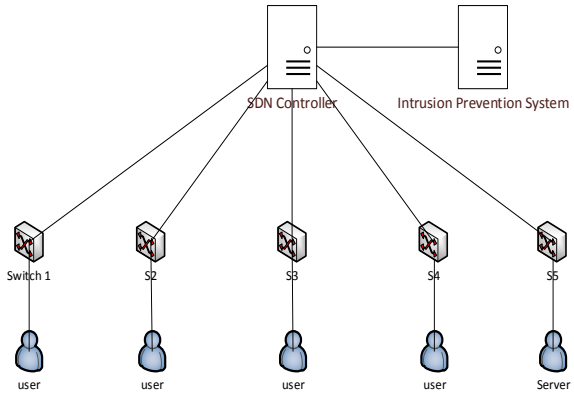
Gambar 2. Alur kerja IPS-SDN

Berdasarkan gambar di atas, fungsi program *alert interpreter* adalah untuk membaca *log* yang berasal dari *snort*, kemudian memberikan perintah *blocking* ke SDN *controller.* Data yang dibaca dari *log snort* yaitu variabel IP *address sender* dan IP *address receiver.* Setelah mendapatkan kedua variabel tersebut, akan dibuat suatu rule yang berisi perintah untuk *blocking* koneksi antara IP *address sender* dan IP *address receiver*. Rule tersebut yang kemudian dimasukkan kedalam *controller*, dan *controller* akan memberikan perintah ke switch untuk melakukan *blocking.* Secara umum, algoritma blocking adalah sebagai berikut :

If attack occured
read log
send notification to controller
create firewall
block incoming connection
Else
Forward packets to destination

3.3 Topologi Jaringan

Topologi jaringan dalam penelitian ini adalah sebagai berikut :



Gambar 3. Arsitektur Jaringan

Gambar di atas merupakan topologi jaringan yang dipakai dalam penelitian ini. Terdapat beberapa komponen dalam topologi jaringan diatas, yaitu :

1. SDN *controller*
   Digunakan untuk melakukan pengaturan seluruh *switch* yang terdapat di dalam jaringan. *Controller* yang digunakan adalah Ryu. Ryu memiliki fitur *access control list* yang dapat dikembangkan agar mampu melakukan *blocking host* yang melakukan serangan.

2. *Intrusion Prevention System* (IPS)
   IPS menggunakan basis IDS yaitu *snort* yang dikembangkan agar mampu melakukan *blocking* terhadap serangan.

3. Switch
   Switch yang digunakan harus mendukung protokol *openflow.*

4. User
   Host yang terhubung ke switch dan melakukan akses data ke server. User juga merepresentasikan *attacker* yang menyerang server.

5. Server
   Host yang membuka beberapa layanan servis seperti HTTP, FTP, dan SSH. Dalam penelitian ini server menjadi target penyerangan yang dilakukan oleh *host*.

Adapun untuk pengalamatan IP Address untuk scenario pengujian adalah sebagai berikut :

Tabel 1. Pengalamatan IP Address

Host	IP Address
Server	172.16.0.5
User 1/attacker	172.16.0.11
User 2	172.16.0.12
User 3	172.16.0.13
User 4	172.16.0.14
Controller	127.0.0.1:6633
IDS	127.0.0.1

Seluruh arsitektur jaringan diatas berjalan pada simulator jaringan mininet, dan controller SDN menggunakan ryu.

3.4 SKENARIO PENGUJIAN

Skenario Pengujian kinerja *QoS*: pengujian ini dimaksudkan untuk mengetahui pengaruh IPS di dalam jaringan SDN. Adapun parameter yang digunakan adalah *delay, throughput, jitter.* Dalam pengujian ini *traffic* yang digunakan adalah *traffic* data. *Traffic* tersebut akan dibangkitkan dengan menggunakan *traffic generator*. Pengujian dilakukan dengan

menambahkan rule secara bertahap kelipatan 50 rule hingga 250 rule, dan dengan menggunakan *traffic background* 0 Mbps, 25 Mbps dan 50 Mbps.

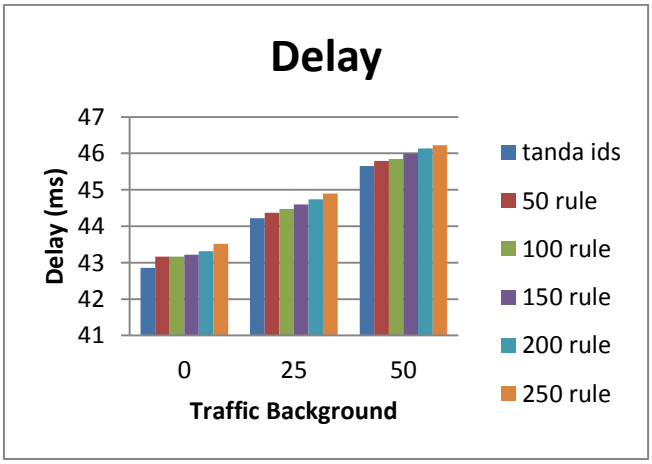
Skenario pengujian serangan: pengujian ini dimaksudkan untuk menguji kemampuan IPS dalam mendeteksi dan memblokir serangan di jaringan SDN. Adapun skenario pengujian serangan adalah sebagai berikut: *SQL Injection, XSS attack, Command Injection, port scanning, DoS Flood Attack, FTP Brute Force attack*. Pengujian dikatakan berhasil jika IPS mampu mendeteksi serangan dan melakukan *blocking* serangan yang berasal dari *attacker*.

4. HASIL PENELITIAN

4.1 Hasil Pengujian Quality of Service

Pengujian performansi jaringan IPS SDN dilakukan dengan cara membandingkan nilai *throughput, delay jitter*, pada jaringan SDN tanpa IPS dan jaringan SDN dengan IPS.

4.1.1 Delay



Gambar 4. Grafik Delay (ms)

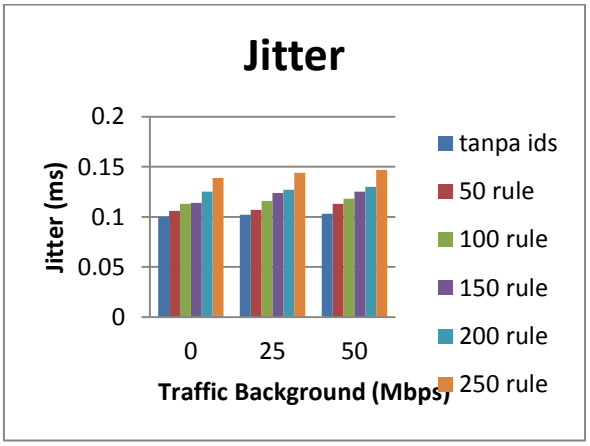
Tabel 2. Tabel Delay (ms)

	Traffic Background		
Jumlah Rule	**0**	**25**	**50**
tanpa ips	42.855	44.216	45.65
50	43.163	44.37	45.789
100	43.163	44.471	45.849
150	43.221	44.596	45.999
200	43.315	44.734	46.139
250	43.52	44.897	46.225

Dari Gambar 4.1 dan Tabel 4.1, nilai *delay* mengalami kenaikan secara signifikan setelah diintegrasikannya IPS ke dalam jaringan SDN. Setiap penambahan 50 rule dalam IPS terjadi kenaikan *delay* rata rata sebesar 0.1 ms. Integrasi IPS ke dalam jaringan SDN memberikan dampak nilai *delay* meningkat. Hal ini dikarenakan arsitektur IPS terdapat komponen *packet decoder*, preprosesor, *Detection engine, logging engine*, dan *alerting engine*. Setiap paket yang masuk ke dalam IPS akan dilakukan penggecekan di *detection engine,* dan IPS

memerlukan waktu lebih lama untuk mencocokkan *traffic* yang lewat dengan rule yang ada. Terlihat bahwa semakin banyak jumlah rule yang dijalankan oleh IPS, nilai *delay* semakin bertambah.

4.1.2 *Jitter*



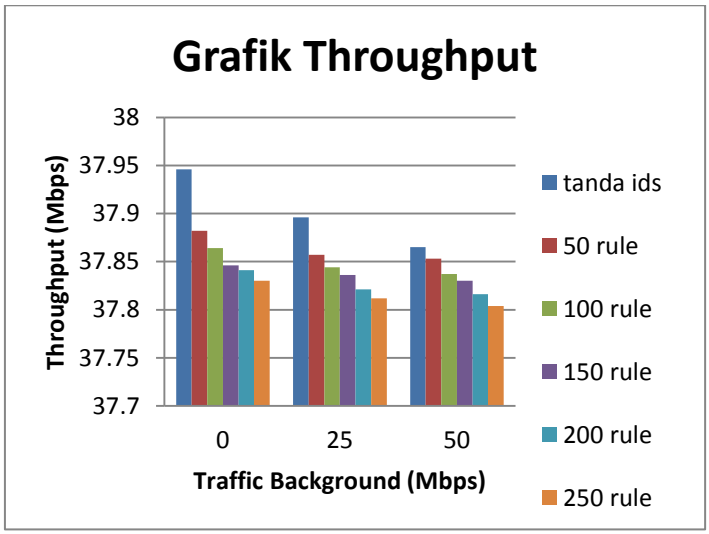
Gambar 5. Grafik Jitter

Tabel 3. Tabel Jitter

	Traffic Background (Mbps)		
Jumlah Rule	**0**	**25**	**50**
tanpa ids	0.1	0.102	0.103
50	0.106	0.107	0.113
100	0.113	0.116	0.118
150	0.114	0.124	0.125
200	0.125	0.127	0.13
250	0.139	0.144	0.147

Dari Gambar 5 dan Tabel 3, nilai *jitter* mengalami kenaikan secara signifikan setelah diintegrasikannya IPS ke dalam jaringan SDN. Setiap penambahan 50 rule dalam IPS terjadi kenaikan *delay* rata rata sebesar 0.01 ms. Integrasi IPS ke dalam jaringan SDN memberikan dampak nilai *jitter* meningkat. Hal ini dikarenakan arsitektur IPS terdapat komponen *packet decoder, preprosesor, Detection engine, logging engine,* dan *alerting engine*. Setiap paket yang masuk ke dalam IPS akan dilakukan pengecekan di *detection engine,* dan IPS memerlukan waktu lebih lama untuk mencocokkan *traffic* yang lewat dengan rule yang ada. Terlihat bahwa semakin banyak jumlah rule yang dijalankan oleh IPS, nilai jitter semakin bertambah.

4.1.3 *Throughput*



Gambar 6. Grafik Throughput

Tabel 4. Tabel *Throughput* (Mbps)

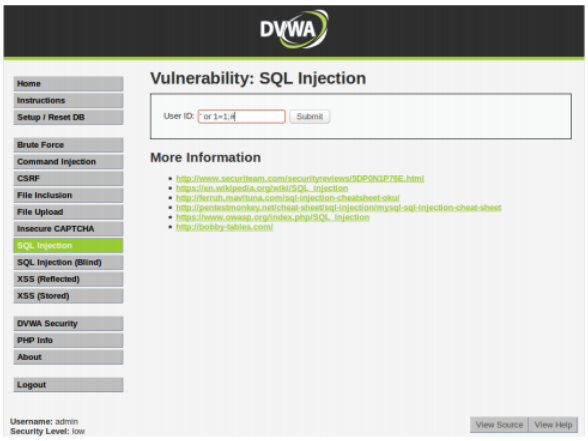
	Traffic Background (Mbps)		
Jumlah rule	**0**	**25**	**50**
tanpa ids	37.946	37.896	37.865
50	37.882	37.857	37.853
100	37.864	37.844	37.837
150	37.846	37.836	37.83
200	37.841	37.821	37.816
250	37.83	37.812	37.804

Nilai *throughput* pada Gambar 6 dan Table 4 secara umum mengalami penurunan pada *background traffic* 0 Mbps, 25 Mbps, dan 50 Mbps. Hal ini dikarenakan adanya proses pengecekan paket yang masuk dari *controller* SDN ke IPS. Setiap paket yang masuk ke dalam IPS akan dilakukan pengecekan di *detection engine,* dan IPS memerlukan waktu lebih lama untuk mencocokkan *traffic* yang lewat dengan rule yang ada.

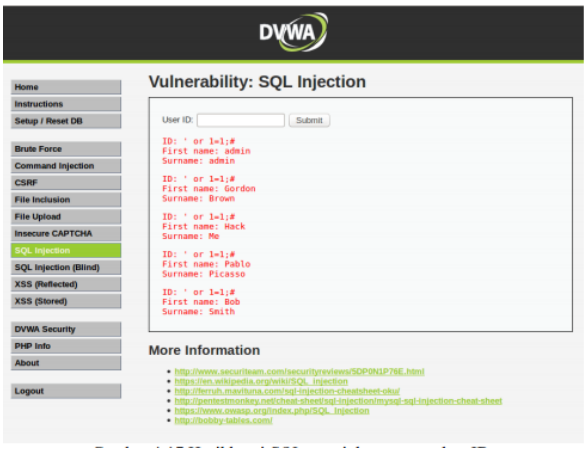
4.2 Pengujian Serangan

4.2.1 *SQL Injection Attack*

Dalam pengujian SQL *injection*, digunakan aplikasi DVWA. Dalam aplikasi tersebut terdapat opsi untuk melakukan simulasi SQL *injection.* Pengujian SQL *injection* dilakukan dengan memasukkan parameter "' or 1=1;#" pada form ID seperti pada Gambar 7. *query* tersebut akan menampilkan seluruh data pengguna yang terdapat di database.



Gambar 7. Input ID pada aplikasi DVWA



Gambar 8. Hasil Query SQL

Agar IPS dapat mendeteksi serangan tersebut, ditambahkan rule pada IPS-SDN yang berisi aturan untuk mencari karakter yang digunakan untuk melakukan serangan SQL *Injection*. IPS berhasil mendeteksi serangan SQL *injection* dan menuliskan *log* serangan tersebut seperti pada gambar di bawah ini :

```
08/01-19:48:49.077654 ,"SQL Injection attempt",TCP,172.16.0.11,172.16.0.5
```

Gambar 9. log IPS serangan SQL injection

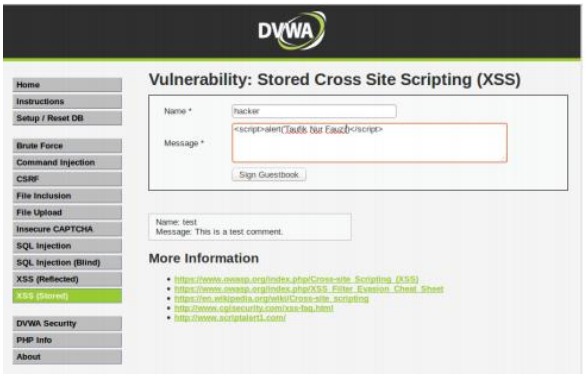
Log firewall yang dihasilkan oleh controller SDN adalah sebagai berikut :

```
curl http://127.0.0.1:6633/firewall/rules/0000000000000005
 [
   {
    "access_control_list": [
     {
      "rules": [
       {
        "priority": 1,
        "dl_type": "IPv4",
        "nw_dst": "172.16.0.5",
        "nw_src": "172.16.0.11",
        "rule_id": 3,
        "actions": "DENY"
       } ]                                                    ]
```

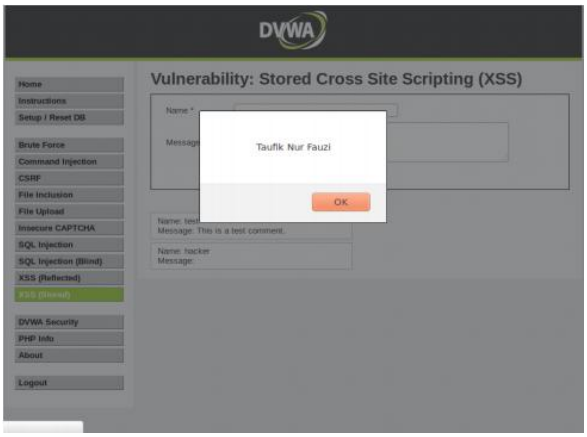
```
      }
    ],
    "switch_id": "0000000000000005"
  }
]
```


4.2.2 *Cross Site Scripting (XSS) Attack*

Simulasi serangan XSS dilakukan pada aplikasi DVWA dengan cara memasukkan karakter <script>alert(Taufik Nur Fauzi)</script>. Sebelumnya, IPS ditambahkan rule untuk mendeteksi serangan XSS Tersebut.



Gambar 10. Penulisan kode javascript pada aplikasi



Gambar 11. Serangan XSS Berhasil

IPS-SDN mampu mendeteksi serangan IPS seperti pada log di bawah ini:

```
08/01-19:52:59.354521 ,"Cross-site scripting
attempt",TCP,172.16.0.11,172.16.0.5
```

Gambar 12. Log XSS

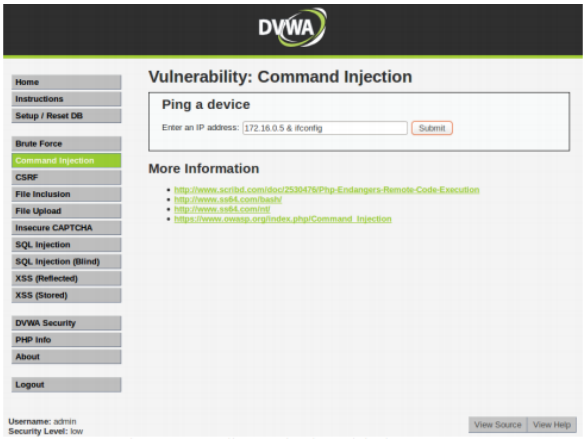
Log firewall yang dihasilkan oleh controller SDN adalah sebagai berikut :


```
curl http://127.0.0.1:6633/firewall/rules/0000000000000005
 [
   {
    "access_control_list": [
     {
      "rules": [
       {
        "priority": 1,
```

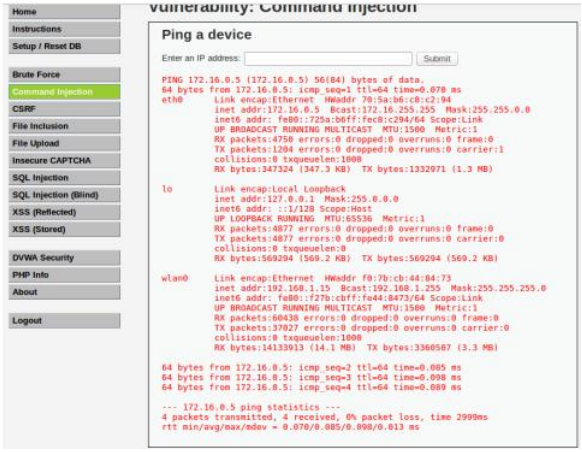
```
            "dl_type": "IPv4",
            "nw_dst": "172.16.0.5",
            "nw_src": "172.16.0.11",
            "rule_id": 4,
            "actions": "DENY"
          } ]                                                    ]
      }
    ],
    "switch_id": "0000000000000005"
  }
]
```


4.2.3 *Command Injection Attack*

Serangan *command injection* dilakukan dengan cara menjalankan perintah ifconfig pada aplikasi DVWA. Pada form Enter an IP Address, *attacker* mengisi form dengan parameter 172.16.0.5 & ifconfig & ping 172.16.0.5. Hasil serangan ditunjukkan pada gambar di bawah ini:



Gambar 13. Serangan *Command Injection* pada aplikasi



Gambar 14. Hasil Serangan Command Injection

Log yang dihasilkan IPS-SDN dari serangan tersebut adalah sebagai berikut :

```
08/01-19:55:19.018315 ,"command line injection
(ifconfig)",TCP,172.16.0.11,172.16.0.5
```

Gambar 15. Output Log IPS Serangan Command Injection

Log firewall yang dihasilkan oleh controller SDN adalah sebagai berikut :

```
curl http://127.0.0.1:6633/firewall/rules/0000000000000005
 [
   {
    "access_control_list": [
     {
      "rules": [
       {
        "priority": 1,
        "dl_type": "IPv4",
        "nw_dst": "172.16.0.5",
        "nw_src": "172.16.0.11",
        "rule_id": 5,
        "actions": "DENY"
       } ]                                                    ]
     }
    ],
    "switch_id": "0000000000000005"
   }
 ]
```

4.2.4 *Port Scanning*

Pengujian *port scanning* dilakukan dengan menggunakan *tools* NMAP*. Port scanning* dugunakan *attacker* untuk memetakan *device* yang terhubung ke dalam jaringan. Proses pengujiannya seperti pada gambar di bawah ini :

```
nmap -v -n -sP --send-ip 172.16.0.0/28
```

Gambar 16. Perintah port scanning

```
Nmap scan report for 172.16.0.5
Host is up (0.00068s latency).
MAC Address: 70:5A:B6:C8:C2:94 (Compal Information (kunshan) CO.)
Nmap scan report for 172.16.0.6 [host down]
Nmap scan report for 172.16.0.7 [host down]
Nmap scan report for 172.16.0.8 [host down]
Nmap scan report for 172.16.0.9 [host down]
Nmap scan report for 172.16.0.10 [host down]
Nmap scan report for 172.16.0.12
Host is up (0.00031s latency).
MAC Address: 00:00:00:00:00:02 (Xerox)
Nmap scan report for 172.16.0.13
Host is up (0.013s latency).
MAC Address: 00:00:00:00:00:03 (Xerox)
Nmap scan report for 172.16.0.14
Host is up (0.00075s latency).
MAC Address: 00:00:00:00:00:04 (Xerox)
Nmap scan report for 172.16.0.15 [host down]
Nmap scan report for 172.16.0.11
Host is up.
Read data files from: /usr/bin/../share/nmap
Nmap done: 16 IP addresses (5 hosts up) scanned in 1.67 seconds
           Raw packets sent: 97 (3.656KB) | Rcvd: 9 (300B)
```

Gambar 17. Output Port Scanning

IPS-SDN berhasil mendeteksi serangan *port scanning* ditunjukkan dengan log dibawah ini :

```
08/01-19:57:59.166005 ,"nmap ping scan",ICMP,172.16.0.11,172.16.0.5
08/01-19:57:59.361387 ,"nmap ping scan",ICMP,172.16.0.11,172.16.0.14
08/01-19:57:59.362637 ,"nmap ping scan",ICMP,172.16.0.11,172.16.0.13
```

Gambar 18. Log IPS port scanning

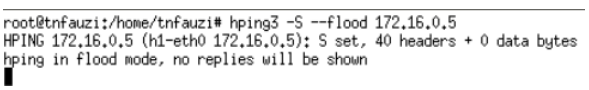
Log firewall yang dihasilkan oleh controller SDN adalah sebagai berikut :

```
curl http://127.0.0.1:6633/firewall/rules/0000000000000005
 [
   {
    "access_control_list": [
```

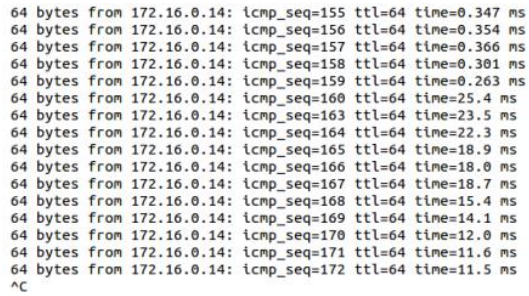
```
    {
      "rules": [
        {
          "priority": 1,
          "dl_type": "IPv4",
          "nw_dst": "172.16.0.5",
          "nw_src": "172.16.0.11",
          "rule_id": 6,
          "actions": "DENY"
        } ]                                                    ]
    }
   ],
   "switch_id": "0000000000000005"
  }
]
```


4.2.5 *DoS Flooding*

Denial of Service adalah jenis serangan yang dilakukan untuk mengurangi kinerja dari suatu target dengan cara membanjiri *request*/ paket yang dialamatkan ke target. Dalam pengujian DoS, menggunakan teknik *Syn Flooding* dengan menggunakan *tools* hping3. Pada saat terjadi *Syn Flooding*, *response time* target menjadi turun seperti ditunjukkan pada Gambar 19 di bawah ini.

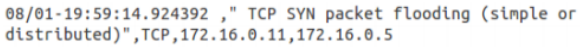


Gambar 19. Perintah Syn Flooding



Gambar 20. Output Syn Flooding

Hasil deteksi IPS-SDN ditunjukkan pada log di bawah ini



Gambar 21. Log IPS syn Flooding

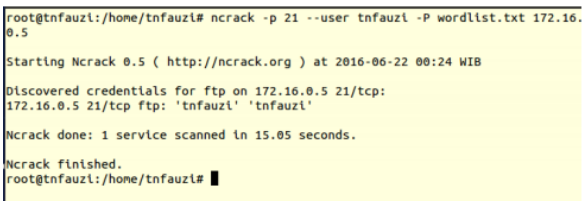
Log firewall yang dihasilkan oleh controller SDN adalah sebagai berikut :


```
curl http://127.0.0.1:6633/firewall/rules/0000000000000005
 [
   {
    "access_control_list": [
     {
      "rules": [
       {
        "priority": 1,
        "dl_type": "IPv4",
        "nw_dst": "172.16.0.5",
        "nw_src": "172.16.0.11",
```

```
          "rule_id": 7,
          "actions": "DENY"
         } ]                                                    ]
      }
    ],
    "switch_id": "0000000000000005"
  }
]
```


4.2.6 *FTP Brute Force*

Serangan *FTP brute force* merupakan serangan yang ditujukan untuk masuk ke layanan FTP secara tidak sah. Caranya adalah dengan melakukan *brute force username* dan *password* dengan menggunakan *wordlist attack*. Tools yang digunakan untuk pengujian ini menggunakan Ncrack. Untuk menjalankan serangan ini, dibutuhkan *wordlist username* dan *wordlist password.*



Gambar 22. Serangan FTP Brute Force

Gambar di atas menunjukkan bahwa serangan FTP *brute force* berhasil menembus FTP server. IPS-SDN mampu mendeteksi serangan tersebut seperti ditunjukkan pada log dibawah ini :


```
08/01-20:08:31.854471 ,"FTP Brute Force",TCP,172.16.0.11,172.16.0.5
08/01-20:08:47.777142 ,"FTP Brute Force",TCP,172.16.0.11,172.16.0.5
08/01-20:08:47.838501 ,"FTP Brute Force",TCP,172.16.0.11,172.16.0.5
08/01-20:09:04.238524 ,"FTP Brute Force",TCP,172.16.0.11,172.16.0.5
08/01-20:09:20.266268 ,"FTP Brute Force",TCP,172.16.0.11,172.16.0.5
08/01-20:09:20.266327 ,"FTP Brute Force",TCP,172.16.0.11,172.16.0.5
08/01-20:09:20.266403 ,"FTP Brute Force",TCP,172.16.0.11,172.16.0.5
08/01-20:09:20.266567 ,"FTP Brute Force",TCP,172.16.0.11,172.16.0.5
08/01-20:09:20.266765 ,"FTP Brute Force",TCP,172.16.0.11,172.16.0.5
08/01-20:09:20.279491 ,"FTP Brute Force",TCP,172.16.0.11,172.16.0.5
08/01-20:09:20.285875 ,"FTP Brute Force",TCP,172.16.0.11,172.16.0.5
08/01-20:09:20.328369 ,"FTP Brute Force",TCP,172.16.0.11,172.16.0.5
08/01-20:09:20.331917 ,"FTP Brute Force",TCP,172.16.0.11,172.16.0.5
```


Gambar 23. **Log IPS terhadap serangan *FTP Bruteforce***

Log firewall yang dihasilkan oleh controller SDN adalah sebagai berikut :

curl http://127.0.0.1:6633/firewall/rules/0000000000000005


```
[
  {
    "access_control_list": [
      {
        "rules": [
          {
            "priority": 1,
            "dl_type": "IPv4",
            "nw_dst": "172.16.0.5",
            "nw_src": "172.16.0.11",
            "rule_id": 8,
            "actions": "DENY"
          } ]                                                   ]
      }
    ],
    "switch_id": "0000000000000005"
  }
]
```

5. KESIMPULAN

Dari hasil pengujian yang telah dilakukan dalam penelitian ini, dapat diambil beberapa kesimpulan yaitu :

IPS di dalam jaringan SDN dapat meningkatkan tingkat keamanan di dalam jaringan. Hal ini dibuktikan dengan pengujian serangan dari layer *network, transport*, dan *application* mampu di deteksi dan diblok oleh IPS. Integrasi IPS dalam jaringan SDN memberikan pengaruh terhadap *throughput, delay* dan *jitter.* Kinerja ketiga parameter tersebut menurun setelah integrasi IPS ke jaringan SDN. Untuk setiap kenaikan 50 rule dalam IPS, nilai *throughput* berkurang rata rata 100 kbs, *delay* naik rata rata 0.1 ms, dan *jitter* naik rata rata 0.02 ms.

DAFTAR PUSTAKA

Rujukan Buku:

Morreale, P. A., & Anderson, J. A. (2015). *Software Defined Networking Design and Development.* Florida: CRC Press .

Nadeau, T. D., & Gray, K. (2013). *Software Defined Networks.* California: O'Reilly Media.

Rahman, R. U. (2003). *Intrusion Detection System with Snort, Advanced IDS Techniques Using Snort, Apache, MySQL, PHP, and ACID.* New Jersey: Prentice Hall PTR .

Thomas, T. (2005). *Network Security First-Step.* Yogyakarta: Penerbit ANDI .

Rujukan Prosiding:

Lei, Z. (2013). Deployment of Intrusion Prevention System ased on Software Defined Networking. *15th International Conference on Communication Technology.* IEEE.

Lim, S., Ha, J., Kim, H., Kim, Y., & Yang, S. (2014). A SDN-Oriented DDoS Blocking Scheme for Botnet-Based Attacks. *International Conference on Ubiquitious and Future Networks*, (hal. 63-68).

Lopez, M., & Duarte, O. (2015). Providing Elasticity to Intrusion Detection System in Virtualized Software Detection System. *2015 IEEE International Conference on Communications*, (hal. 7120-7125). London.

Nunes, B., Mendonca, M., Nguyen, N., Obraczka, K., & Turletti, T. (2013). A Survey of Software Defined Network : Past, Present, and Future. *IEEE Communication Survey & Tutorial*, (hal. 1617-1631). 2014.

Yamahata, I. (2014). *Ryu: SDN Framework and Python Experience.*

Zhengyang, X. (2014). *An SDN-Based IPS Development Framework in Cloud Networking Environment.* Arizona: Arizona State University.

Zhiyuan, H. e. (2015). A Comprehensive security architecture for SDN. *18th International Conference on Intelegence in Next Generation Networks.* 2015: IEEE.