Implementasi *Intrusion Prevention System* Untuk Mencegah Serangan DDOS pada *Software Defined Network*

Ilham Miftakhul Huda[1], I Made Suartana[2]

[1,2] Jurusan Teknik Informatika/Teknik Informatika, Universitas Negeri Surabaya
[1]ilhamhuda16051204006@mhs.unesa.ac.id
[2]imadesuartana@unesa.ac.id

***Abstrak*— *Software defined network* adalah konsep jaringan komputer yang mana memisahkan *control plane* dan *data plane* yang mana akan memudahkan administrator jaringan untuk mengelola jaringan komputer tersebut secara terpusat dengan menggunakan sebuah *controller*. Dalam penerapan SDN administrator jaringan perlu menerapkan sistem keamanan jaringan. Hal ini disebabkan karena tingginnya ancaman yang bertujuan untuk mengganggu bahkan merusak koneksi antar jaringan komputer. Serangan DDOS merupakan serangan yang dilakukan secara terdistribusi dengan cara membajiri lalu lintas jaringan internet pada server, sistem yang mengakibatkan tidak bisa lagi menampung koneksi (*overload*) yang bertujuan agar target tidak dapat diakses. Serangan ini dapat berasal dari dalam jaringan itu sendiri ataupun dari jaringan luar. Maka dari itu dibutuhkan metode untuk mengamankan sumber daya jaringan komputer dengan menggunakan metode *Intrusion Detection Prevention System* yang bermanfaat untuk mendeteksi serangan sekaligus memblokir serangan yang terdapat dalam jaringan komputer. Pada penelitian ini berhasil menerapkan metode *intrusion detection prevention system* pada jaringan *software defined network* menggunakan *snort* sebagai sistem deteksi serangan dan *rest_firewall* dari *ryu* untuk memblokir serangan. *Snort* akan mendeteksi paket yang dianggap berbahaya sesuai dengan *rules* yang sudah ditentukan. Kemudian dari *rules* tersebut akan diambil data seperti paket *source*, paket *destination*, dan protokol paket pada file log. Saat terjadi serangan DDOS pada *host* yang tidak terintegrasi IPS Ping pada *host* tersebut besar, ketika IPS diaktifkan kondisi pada jaringan kembali normal. Kinerja CPU untuk proses pengguna dan sistem lebih tinggi ketika system terintegrasi dengan IPS. Saat sistem terintegrasi dengan IPS penggunaan *memory* lebih banyak daripada saat *system* tidak terintegrasi tanpa IPS.**

***Kata Kunci*— Software Defined Network, Intrusion Detection Prevention System, DDOS, Ryu.**

I. PENDAHULUAN

Pada saat ini jaringan tradisional mulai ditinggalkan karena dirasa sangat lambat serta pengembangannya bergantung pada vendor dan pengembang dari luar seperti komunitas *open source* tidak dapat bebas mengembangkan perangakat jaringan komputer. Pada jaringan tradisional sebuah perangkat jaringan layer 2 (*switch*) dan layer 3 (*router*) menggabungkan antara *control plane* dan *data plane*. Penggabungan *control plane* dan *data plane* ini menyebabkan kompleksitas pada jaringan komputer yang sangat tinggi apabila ada perangkat jaringan baru maka router yang lama akan memberikan informasi ke router sebelah sehingga seluruh *table routing* pada perangkat jaringan tersebut wajib di *update* ulang[1]. Oleh karena itu sangat perlu adanya pembaruan teknologi jaringan komputer yang dapat mengatasi kompleksitas pada jaringan tradisional dengan adanya arsiterktur jaringan SDN(*Software Defined Network*).

Software defined networking (SDN) merupakan suatu konsep jaringan komputer yang mana sistem pengontrol dari arus data dipisahkan dari perangkat kerasnya[2]. Pada biasanya, sistem yang berguna sebagai pembuat keputusan ke mana arus data dikirimkan dibuat menyatu dengan perangkat kerasnya. Konfigurasi *Software Defined Network* dapat menghasilkan sebuah jaringan yang mana perangkat keras pengontrol lalu lintas data secara fisik dipisahkan dari perangkat keras data forwarding plane. Dengan demikian, administrator jaringan dapat mengatur perangkat jaringan dengan melakukan sentralisasi pengaturan pada *control plane*. Penemu dan penyedia konsep jaringan SDN ini mengklaim dapat menyederhanakan jaringan komputer.

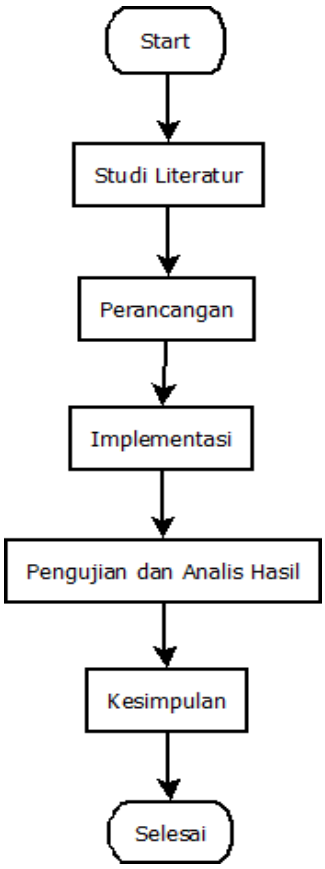
Bukan hanya keuntungan saja yang didapat dari SDN, pemisahan antara *control plane* dan *data plane* pasti mempunyai permasalahan, seperti permasalahan pada keamanan kontroler. Kontroler yang berfungsi sebagai otak dari jaringan komputer bertanggung jawab atas jadwal dan trafik pada jaringan tersebut. Pada umumnya terdapat dua mode yaitu mode proaktif dan mode reaktif yang mana aturan dapat dipasang pada perangkat *switch*. Pada mode proaktif, tanpa melibatkan *controller* dalam memecah kebijakan jaringan menjadi aturan arus serta memasangnya di *switch*. Sebaliknya pada mode reaktif, kontroler menghitung serta meng-*install* aturan yang disaat ada *trigger* tertentu yang telah ditetapkan pada *layer* aplikasi di kontroler. Akan tetapi, pada saat installasi aturan reaktif bisa membuat kontroler SDN dan *switch* sangat rentan dari ancaman *Distributed Denial of Service* (DDoS). Serangan DDoS adalah serangan yang sifatnya terdistribusi. Serangan DDoS dapat melumpuhkan layanan jaringan dengan menghancurkan perangkat jaringan. Pada umumnya, serangan DDOS membanjiri target dengan mengirimkan paket berbahaya [3]. Maka dari itu dibutuhkan sistem yang berfungsi untuk mencegah terjadinya serangan DDOS pada jaringan SDN. Pada penelitian [4] IPS berbasis *Athena* bisa mencegah serangan UDP *flood* dan TCP SYN *flood* yang dibuktikan dengan turunnya nilai throughput dalam kondisi normal. Saat terjadi serangan FTP *server* beban pada CPU meningkat, sedangkan pada saat terjadi serangan UDP dan SYN *flood* beban pada CPU berkurang. Penelitian [5] serangan DDoS UDP *flood* pada perangkat HG553 di jaringan

SDN menyebabkan jaringan tidak bisa bekerja dikarenakan *switch openflow* tidak bisa berkomunikasi dengan *controller* dan ketika serangan UDP *flood* berlangsung pemakaian prosessor menjadi meningkat.

Berdasarkan latar belakang diatas penulis akan mengimplementasikan mekanisme security pada SDN dengan menggunakan konsep IDPS, yang disimulasikan menggunakan *software mininet* sebagai emulator jaringan dengan menerapkan *snort* sebagai *intrusion detection prevention system*(IDS*)* dan akan diintegrasikan dengan modul *rest_firewall Ryu Controller* sebagai *intrusion prevention system* (IPS). Untuk ujicoba mekanisme *security* yang telah dibuat atau diimplementasikan digunakan simulasi serangan TCP Syn *Flood* . Selanjutnya dilakukan analisa kemampuan mekanisme security yang digunakan untuk mendeteksi serangan pada jaringan SDN.

II. METODOLOGI PENELITIAN

Berikut ini merupakan alur dari penelitian implementasi *intrusion prevention system* untuk mencegah serangan DDOS pada *Software Defined Nework*(SDN).

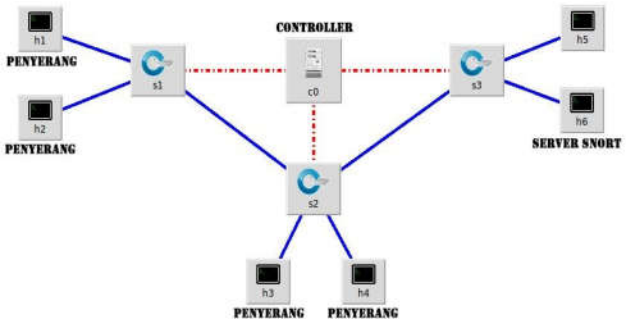


Gambar 1. Alur Penelitian

Tahapan dari alur penelitian seperti digambarkan pada Gambar 1 yaitu dimulai dengan studi literatur yang bertujuan untuk mencari literature relevan yang dapat dijadikan sebagai referensi pendukung dalam penelitian ini. Tahapan selanjutnya yaitu menentukan kebutuhan sistem. Adapun yang dibutuhkan yaitu ubuntu sebagai sistemp operasi, *ryu* sebagai *controller*, *mininet* sebagai emulator topologi jaringan, *snort* sebagai *intrusion detection system*, *hping3* sebagai metode serangan, dan *top* untuk memantau penggunaan CPU, memory, dan proses yang berjalan pada sistem. Selanjutnya yaitu tahapan implementasi dimana tahap ini untuk menginstall semua kebutuhan sistem yang diperlukan tadi secara baik dan benar. Setelah semuanya terinstall dengan baik tahapan selanjutnya yaitu pengujian dan analisis hasil pada tahap pengujian dilakukan secara 3 tahap yaitu pengujian serangan DDOS tanpa IPS, pengujian serangan DDOS dengan IPS, dan pengujian load server meliput penggunaan CPU dan *memory* yang diujikan saat system dalam keadaan normal, tanpa IPS, dan dengan IPS. Dari pengujian serta analisis hasil tersebut bisa dijadikan kesimpulan dari penelitian yang sudah dikerjakan.

A. Topologi Jaringan

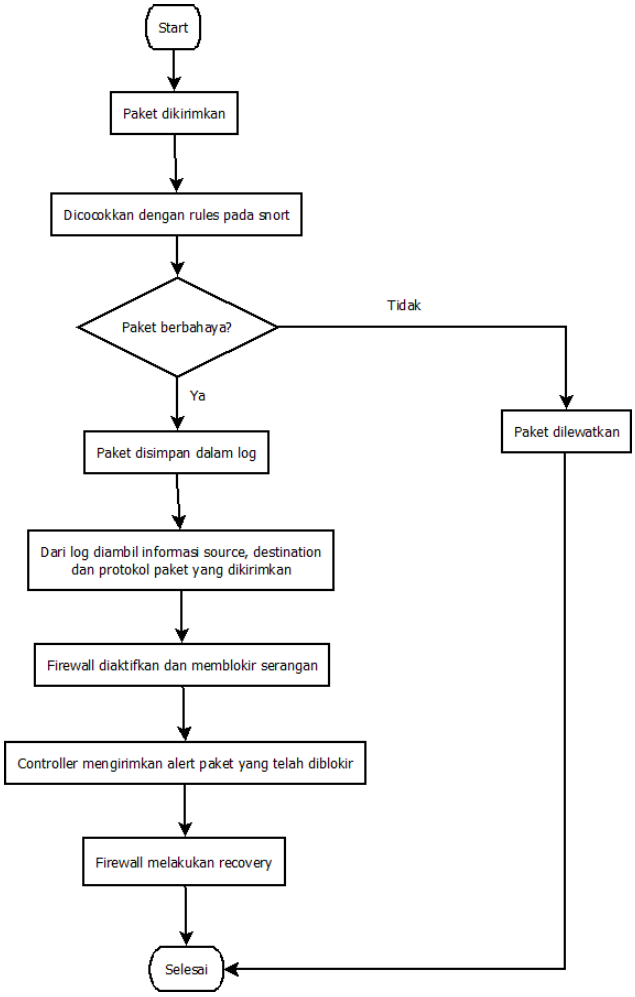
Saat merancang topologi jaringan menggunakan emulator jaringan yaitu *Mininet. Mininet* dapat menjalankan *switch, router, client, dan link.* Pada *mininet* terdapat topologi bawaan seperti topologi *single, tree, dan linear* yang dapat digunakan langsung dengan *command* tertentu. Pada Penelitian ini menggunakan *custom* topologi dengan menggunakan 1 *controller*, 3 *switch*, dan 6 *host*. Adapun yang bertidak sebagai server *snort* yaitu *host 6*. Sedangkan yang bertidak sebagai attacker atau penyerang yaitu *host 1, host 2, host 3,* dan *host 4*.



Gambar 2. Rancangan Topologi Jaringan

B. Intrusion Detection Prevention System

Intrusion Detection Prevention System merupakan bagian dari infrastuktur jaringan yang berfungsi untuk meningkatkan keamanan suatu jaringan. Adapun tugas dari keduanya yaitu *Intrusion Detection System* bertugas untuk mendeteksi ancaman atau serangan pada sebuah jaringan. IDS tidak bisa melakukan aksi mitigasi ketika terjadi ancaman atau serangan dalam jaringan. Sebaliknya IPS adalah sistem yang berperan sebagai pengontrol suatu jaringan yang bisa melakukan tindakan mitigasi ketika terdapat ancaman atau serangan dalam suatu jaringan dengan cara melakukan pemblokiran terhadap paket yang dikategorikan ancaman. Adapun alur proses dari IDPS pada penelitian ini bisa dilihat pada Gambar 3.



Gambar 3. Alur proses IPS

C. *Ryu Controller*

Ryu merupakan salah satu jenis *controller* dari *Software Defined Network* yang berbasis *python*. *Controller* merupakan otak dari *Software Defined Network* yang dirancang untuk menpermudah mengatur suatu jaringan.

D. Skenario Pengujian

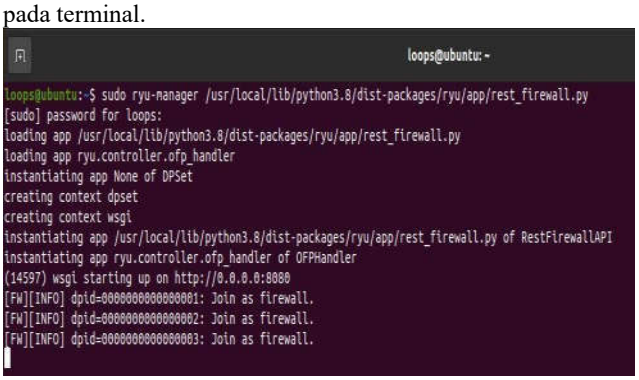
Pada penelitian ini yang bertindak sebagai *attacker* atau penyerang yaitu *host 1, host 2, host 3, dan host 4.* Keempat *host* tersebut akan menyerang secara bersamaan menggunakan jenis serangan TCP *Syn Flood* kepada server *snort* yang berada pada *host 6*. Setelah melakukan penyerangan *host* 5 mencoba ping kepada *host 6* yang bertindak sebagai server *snort* melihat kondisi jaringan apakah menjadi down atau tidak. Adapun pengujian yang dilakukan yaitu :

1. Melakukan serangan DDOS tanpa IPS, ketika serangan DDOS berlangsung dilakukan tes ping untuk melihat efek pada jaringan tersebut.
2. Melakukan serangan DDOS dengan IPS, ketika serangan DDOS berlangsung dilakukan tes ping untuk melihat efek pada jaringan tersebut.
3. Pengujian load server atau resource dilakukan pada server *snort.* Adapun pengujiannya meliputi penggunaan CPU dan *memory* dengan 3 cara yaitu saat sistem dalam keadaan normal tanpa ada serangan, saat sistem terdapat serangan tanpa IPS diaktifkan, dan saat sistem terdapat serangan dengan IPS diaktifkan.

III. HASIL DAN PEMBAHASAN

A. Pengujian Tanpa IPS

Pertama kali yang harus dilakukan adalah menjalankan *controller ryu* dan mengaktifkan *rest_firewall ryu* dengan mengetikkan ***"sudo ryu-manager /usr/local/lib/python3.8/distpackags/tyu/app/rest_firewall.py"*** pada terminal.



Gambar 4. Menjalankan *controller ryu* dan *rest_firewall*

Setelah berhasil menjalankan *controller ryu* langkah selanjutnya yaitu menjalankan topologi mininet dengan mengetikkan ***"sudo mn --custom /home/loops/tex/topologi.py --topo mytopo --controller remote --switch=ovsk,protocols=OpenFlow13"***.

Gambar 5. Menjalankan topologi pada mininet

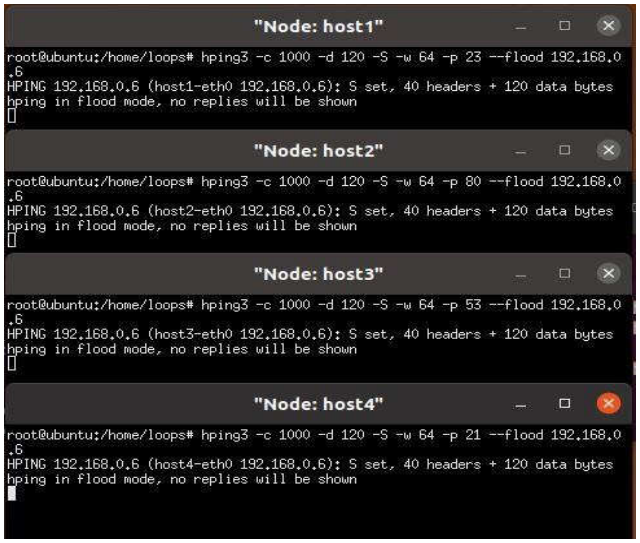
Setelah topologi pada mininet berhasil dijalankan langkah selanjutnya yaitu membuat host dapat berkomunikasi atau terhubung dengan cara mengetikkan ***"cd tex/"*** pada terminal, setelah berada pada direktori **tex** ketikkan ***"./allowFlow.sh"***. Setelah itu kita coba pingall pada mininet. Apabila seperti pada Gambar 6 dibawah ini menandakan semua *host* sudah terhubung satu sama lain.

Gambar 6. Pingall pada mininet

Langkah selanjutnya yaitu menjalankan snort dengan cara mengetikkan ***"sudo snort -i s3-eth2 -c /etc/snort/snort.conf -l /var/log/snort"***. Apabila seperti Gambar 7 dibawah ini menandakan *snort* berhasil dijalankan.

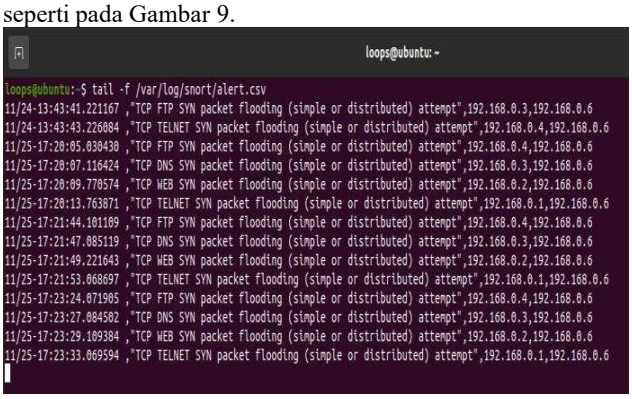
Gambar 7. Snort berhasil dijalankan

Selanjutnya yaitu melakukan serangan DDOS menggunakan *Host1, Host2, Host3, dan Host4* dengan jenis serangan TCP Syn Flood pada *Host6* seperti pada Gambar 8 dibawah ini.



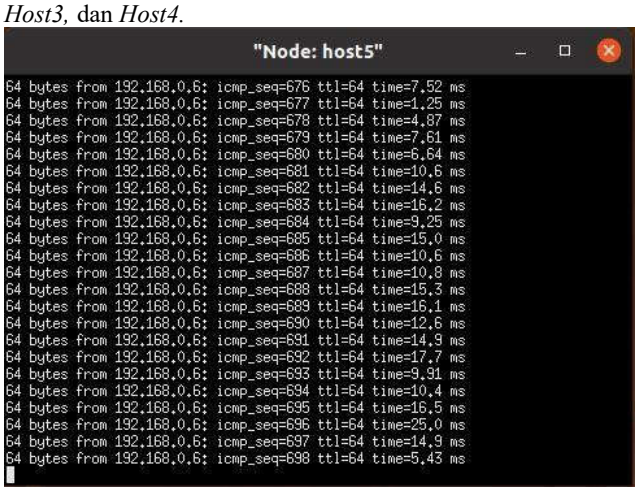
Gambar 8. Host1 – Host4 melakukan serangan

Setelah melakukan serangan pada *Host6* kita lihat apakah snort berhasil mendeteksi adanya paket yang berbahaya atau tidak dengan melihat log file pada *snort*. Cara melihat log file pada *snort* dengan mengetikkan perintah ***"tail -f /var/log/snort/alert.csv"*** pada terminal. Apabila *snort* berhasil mendeteksi paket yang berbahaya muncul informasi seperti pada Gambar 9.



Gambar 9. Log file snort

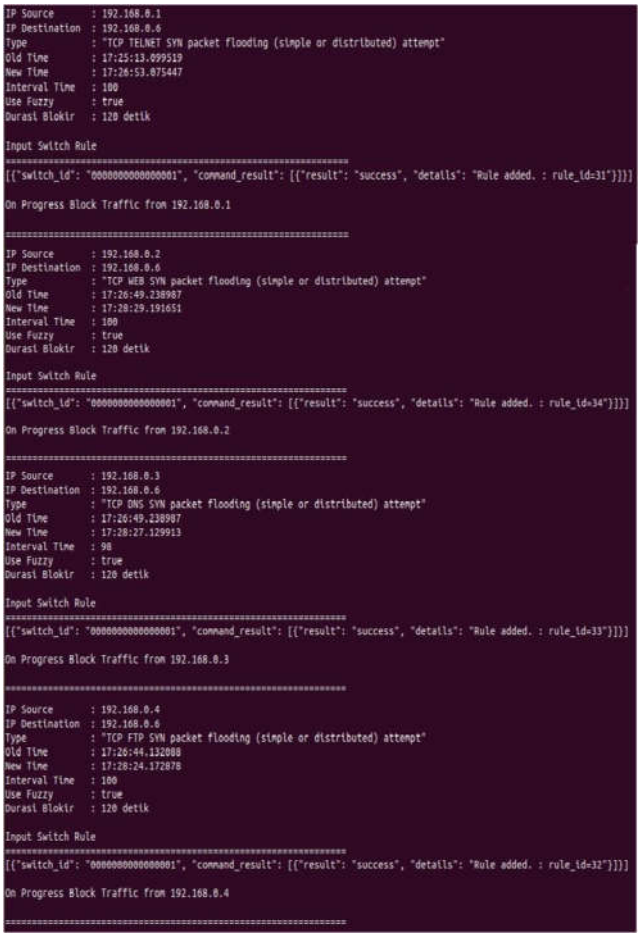
Dan ping pada jaringan akan naik seiring berjalan waktu yang ditunjukkan seperti pada Gambar 10 dikarenakan adanya serangan TCP Syn Flood yang berasal dari *Host1, Host2, Host3,* dan *Host4.*



Gambar 10. Ping pada jaringan besar

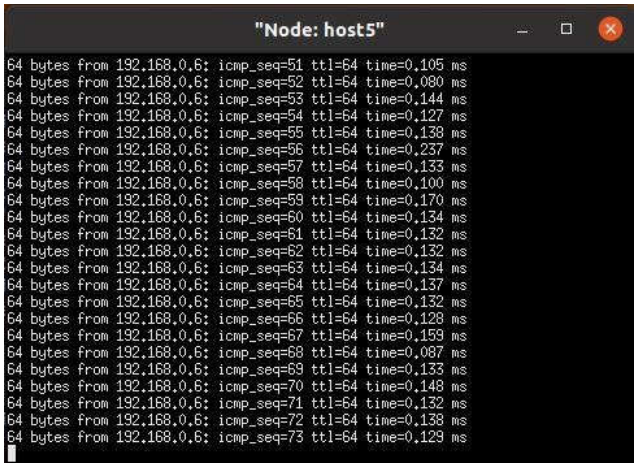
B. Pengujian Dengan IPS

Setelah *snort* berhasil mendeteksi adanya paket yang berbahaya langkah selanjutnya yaitu mengimplementasikan *intrusion prevention system* dengan mengaktifkan *rest_firewall* pada *Ryu Controller* untuk memblokir serangan. Langkah yang harus dilakukan dengan mengetikkan perintah ***"cd tex/"*** pada terminal, setelah berada pada direktori **"tex"** ketikkan ***"./hostInspector.sh"*** *rest_firewall* pada *Ryu Controller* aktif maka akan memblokir serangan seperti pada Gambar 11.



Gambar 11. *Rest_firewall ryu controller* berhasi memblokir serangan.

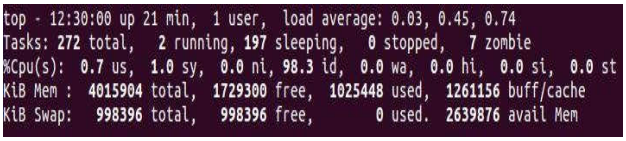
Setelah rest_firewall pada ryu controller berhasil memblokir serangan maka kondisi pada jaringan kembali normal seperti yang ditunjukkan pada Gambar 12.



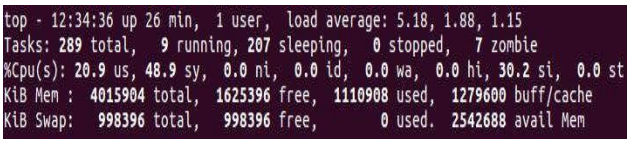
Gambar 12. Kondisi pada jaringan kembali normal

C. Pengujian Load Server

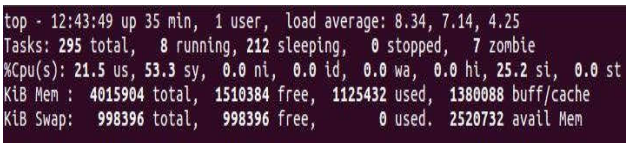
Pada pengujian ini peneliti menggunakan *software* yang bernama ***top*** untuk mengukur pemakaian *cpu* dan memory. Pada pengujian ini sistem diuji secara normal, tanpa IPS, dan dengan IPS. Berikut merupakan hasil dari pengujian load server:



Gambar 13. Sistem keadaan normal

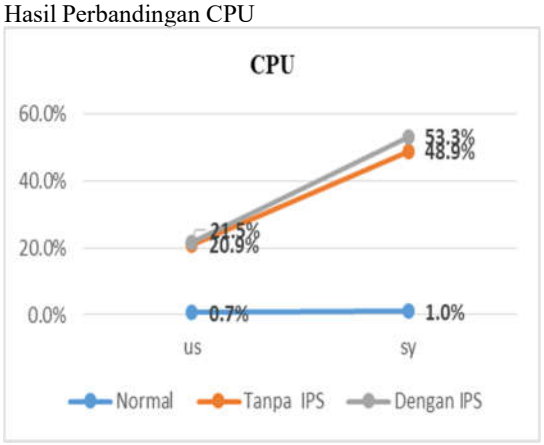


Gambar 14. Sistem tanpa IPS



Gambar 15. Sistem dengan IPS

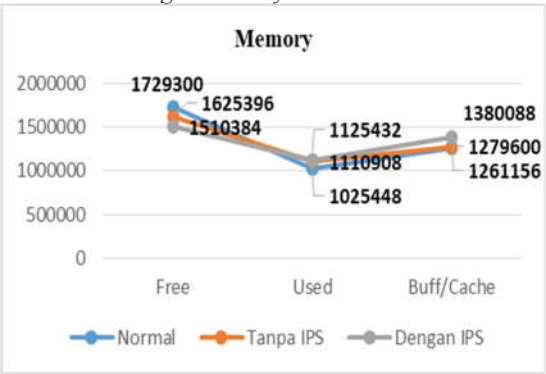
1. Hasil Perbandingan CPU



Gambar 16. Grapik hasil pemakaian CPU

Persentase pemakaian CPU saat sistem keadaan normal yaitu 0.7% untuk proses pengguna (us) dan 1.0% untuk proses CPU yang digunakan oleh sistem (sy). Pada saat sistem berjalan tanpa IPS pemakaian CPU untuk proses pengguna (us) yaitu dengan nilai 20.9% dan untuk pemakaian CPU yang digunakan oleh sistem (sy) dengan nilai 48.9%. Ketika sistem terintegrasi IPS pemakian CPU naik pada nilai 21.5% untuk proses pengguna (us) dan besaran proses CPU yang digunakan sistem naik dengan nilai 53.3%. Kenaikan persentase yang dialami CPU disebabkan oleh bertambahnya proses pada *system*.

2. Hasil Perbandingan *Memory*



Gambar 17. Grapik hasil pemakaian *memory*

Penggunaan *memory* saat sistem keadaan normal adalah 1025448. Pada saat sistem berjalan tanpa IPS *memory* yang digunakan dengan nilai 1110908. Saat melakukan serangan DDOS dengan sistem terintegrasi IPS penggunaan *memory* lebih besar dengan nilai 1125432.

IV. KESIMPULAN

Berdasarkan analisis dari hasil pengujian pada penelitian ini dapat disimpulkan sebagai berikut:

Arsitektur jaringan *software defined network* yang menggunakan *controller ryu* dapat diintegrasikan dengan *snort* sebagai sistem deteksi serangan dan mitigasi serangan. *Intrusion prevention system* dapat memblokir serangan DDOS dengan cara mengambil log informasi berupa paket *source*, paket *destination*, dan protokol paket. Saat terjadi serangan DDOS tanpa IPS ping jaringan menjadi besar. Ketika IPS diaktifkan dan berhasil memblokir serangan kondisi pada jaringan kembali normal. Saat sistem terintegrasi IPS kinerja CPU untuk menjalankan proses pengguna dan sistem lebih berat daripada saat sistem tanpa terintegrasi IPS. Saat sistem terintegrasi IPS penggunaan *memory* lebih besar daripada saat sistem tidak terintegrasi IPS.

UCAPAN TERIMA KASIH

Dengan selesainya penelitian ini penulis panjatkan Puji syukur kepada Allah SWT yang telah memberikan rahmat serta hidayah-Nya sehingga penulis bisa menuntaskan penelitian serta menulis artikel ilmiah dengan baik. Tidak lupa juga penulis mengucapkan terimakasih kepada dirinya sendiri yang telah berusaha sampai saat ini, kedua orang tua serta adik saya yang selalu memberikan do'a dan dukungan, dosen pembimbing, teman dan semua pihak yang telah mendukung, membantu, serta membimbing penulis dalam mengerjakan dan menyelesaikan penelitian ini.

REFERENSI

[1] A. Heryanto and Afrilia, "Software Defined Network Menggunakan Simulator," no. 33, pp. 5–8, 2016.

[2] V. D. Chakravarthy, Y. J. Visishta, and N. Veeramalla, "Detection and prevention of ddos attacks in software defined networking," *Int. J. Adv. Sci. Technol.*, vol. 29, no. 6, pp. 3529–3535, 2020, doi: 10.21090/ijaerd.030620.

[3] B. V Baiju, S. M. Yahiya, P. A. Raj, and S. S. Farooq, "Ddos Attack Detection Using SDN Techniques," vol. 12, no. 10, pp. 326–335, 2021.

[4] M. F. Muntaha, P. H. Trisnawan, and R. Primananda, "Implementasi Intrusion Prevention System (Ips) Berbasis Athena Untuk Mencegah Serangan Ddos Pada Arsitektur Software-Defined Network (Sdn)," *Pengemb. Teknol. Inf. dan Ilmu Komput. Vol.*, vol. 3, no. 7, pp. 6847–6855, 2019.

[5] A. Yasin and I. Mohidin, "Dampak Serangan DDoS pada Software Based Openfow Switch di Perangkat HG553," *J. Technopreneur*, vol. 6, no. 2, p. 72, 2018, doi: 10.30869/jtech.v6i2.206.

[6] P. Putu, K. Adi, R. M. Negara, D. D. Sanjoyo, F. T. Elektro, and U. Telkom, "Integrasi Intrusion Prevention System Dan Analisa Performansi Pada Software Defined Network Intrusion Prevention System Integration and Performance," vol. 5, no. 3, pp. 5046–5053, 2018, [Online]. Available: https://libraryeproceeding.telkomuniversity.ac.id/index.php/engineering/article/view/7922.

[7] A. D. Ummah Izzatul, "Perancangan Simulasi Jaringan Virtual Berbasis Software-Define Networking," *Indones. J. Comput.*, vol. 1, no. 1, pp. 95–106, 2016, doi: 10.21108/indojc.2016.1.1.20.