Deteksi Serangan Denial Of Service (DoS) pada Cloud Menggunakan Security Onion

Farry Cendekiawan Budi Wicaksono[1], I Made Suartana[2]

[1,3] Program Studi S1 Teknik Informatika Fakultas Teknik Universitas Negeri Surabaya
[1]farry.18070@mhs.unesa.ac.id
[3]imadesuartana@unesa.ac.id

***Abstrak*— *Cloud computing* yang semakin marak digunakan menjadi salah satu layanan yang sering dijadikan sebagai target serangan siber. Peran *Network Security Monitoring* (NSM) semakin besar dalam penggunaan *cloud computing*, karena dapat memberikan informasi terkait adanya penyerangan yang yang tengah terjadi. *Denial of Service* (DoS) sebagai serangan yang paling sering menargetkan *cloud computing*, dipilih sebagai metode serangan pada penelitian ini untuk melihat kemampuan Security Onion sebagai NSM yang di integrasikan dengan *Elastic Cloud Compute* (EC2) pada Amazon Web Service (AWS). Sebagai *Network Intrusion Detection System* (NIDS) di Security Onion, Suricata berhasil mendeteksi serangan DoS yang menargetkan *service* yang dimonitor dan mengirimkan *alert* untuk di tampilkan ke Security Onion. Kesimpulan dari hasil yang diperoleh, Security Onion yang diintegrasikan dengan AWS *Cloud* dapat mendeteksi adanya serangan DoS yang terjadi dan memberikan *alert* berdasarkan *rules* suricata yang telah diterapkan.**

***Kata Kunci*— Cloud Computing, Network Security Monitoring, Detection, Denial of Service, Security Onion, Amazon Web Service.**

I. PENDAHULUAN

Perkembangan teknologi komputasi awan atau yang biasa dikenal sebagai *cloud computing* mengalami peningkatan yang cukup pesat. Banyaknya jenis layanan serta fitur yang ditawarkan oleh *cloud computing* membuat banyak pengguna semakin berminat untuk menggunakan *cloud computing*. Salah satunya sebagai tempat penyimpanan atau storage yang bisa diakses dari mana aja dengan hanya berbekal akses internet. *Cloud computing* menggunakan dasar konsep *multi tenancy* yang memungkinkan satu *hardware* dapat melakukan beberapa pemrosesan layanan *cloud* dari user yang berbeda sekaligus. Hal ini menjadikan keuntungan dari sisi *cloud provider* yang bisa menjual layanan *cloud*-nya dengan sumber daya yang masih sedikit, dengan manajemen yang baik pada perangkat keras *cloud* yang digunakan agar dapat optimal dalam menjalankan layanannya[1].

Namun, selalu terdapat terdapat kekurangan dibalik kelebihan yang ditawarkan oleh *cloud computing*. Keamanan data dan privasi dari pengguna menjadi poin yang sangat krusial untuk diperhatikan dalam infrastruktur *cloud*. *Cloud computing* dibangun dengan teknologi virtualisasi, yang memungkinkan *attacker* untuk memanfaatkan celah keamanan yang tersedia, untuk masuk ke dalam sistem *cloud* dan mengambil keuntungan di dalamnya[2]. Jumlah penggunaan *cloud* yang setiap tahun mengalami peningkatan, tentu tidak terhindarkan dari ancaman *cyberattack* yang juga ikut naik setiap tahunnya. Menurut laporan survei yang dilakukan oleh SOPHOS di dalam *whitepaper* "*The State of Cloud Security 2020*", tercatat sebanyak 70% dari 3.521 reponden telah mengalami serangan pada *public cloud* milik mereka.[3].

Terdapat enam kategori ancaman *cyberattack* yang digunakan untuk mengidentifikasi ancaman dalam sistem tertentu, seperti *spoofing, tampering, repudiation, information disclosure, denial of service* (DoS), dan *elevation of privilage*. Pengkategorian tersebut kemudian disingkat menjadi STRIDE *threat model* yang dikembangkan oleh Praerit Grag dan Loren Kohnfelder di Microsoft. Masing-masing kategori menangkap karakteristik individu dari serangan yang menimbulkan jenis ancaman tertentu, dengan mengkategorikan hasil atau efek dari realisasi ancamannya. Dilansir dalam *Cloud Security Threat Report* (CSRT) 2019 yang dibuat oleh Symnatec, ada bentuk-bentuk baru dari ancaman serangan lintas *cloud* yang kini meningkat seperti serangan *malware* dan DDoS[4].

Denial of Service atau biasa dikenal DoS adalah serangan siber yang mentargetkan sebuah sistem atau sumber layanan *cloud* dengan *request* yang banyak dan beruntun menggunakan satu komputer melalui jaringan intenet. Adapun juga DDoS atau *Distributed Denial of Service*, versi lanjutan dari DoS dengan menggunakan beberapa device komputer untuk menyerang target sistem. Kedua bentuk serangan ini mengakibatkan sebuah sistem atau layanan tidak dapat digunakan pengguna untuk sementara waktu atau bahkan tanpa batas waktu[5]. Meski termasuk dalam serangan yang tradisional, serangan jenis DoS masih digunakan hingga saat ini oleh *attacker* untuk menyerang sistem di internet. Hal ini terbukti dengan masih banyaknya serangan-serangan yang menargetkan *server* atau layanan menggunakan metode ini. Pada November 2021, salah satu penyedia layanan cloud terbesar, Microsot Azure, berhasil memitigasi serangan DDoS terbesar dalam sejarah, yaitu sebesar 3.47 Tb/s dengan *packet rate* hingga 340 juta *packets per second* (pps)[6]. Volume ini bahkan lebih besar dibandingan dengan 2.3 Tb/s yang dimitigasi oleh *Amazon Web Service* (AWS) pada tahun 2020.

Dengan masih banyaknya penyerangan menggunakan metode DoS terutama pada *cloud network*, tentu menjadi sebuah tugas wajib bagi *cloud provider* untuk menjaga layanan *cloud* yang diberikan agar tetap aman. Dalam hal ini, pihak *cloud provider* biasanya sudah memiliki divisi khusus untuk menangani hal terkait keamanan jaringan di perusahaannya. Namun ada juga beberapa *cloud provider* yang tidak memiliki divisi tersebut dan lebih memilih untuk bermitra dengan perusahaan IT *security consultant* untuk

menangani masalah terkait keamanan cloud milik mereka. Hal ini sangat penting dikarenakan sistem *cloud* yang bersifat *online* dan *open* melalui internet, membuatnya menjadi rentan terhadap serangan yang berpotensi mengakibatkan kebocoran data ataupun *system down* pada layanan *cloud*. Di sisi lain juga, kepercayaan pengguna terhadap layanan *cloud* sangat krusial, karena dalam layanan tersebut pengguna mempercayakan data-data yang dimilikinya untuk disimpan.

SOC *atau Security Operation Center* adalah sebuah divisi yang biasanya menangani masalah keamanan jaringan sebuah perusahaan. Dalam pekerjaannya, SOC adalah divisi yang pertama kali akan mendapatkan informasi terkait adanya serangan yang menyerang sistem yang mereka monitor. *Monitoring* yang dilakukan oleh divisi SOC tentu memerlukan sistem keamanan untuk menunjangnya, yaitu berupa *Intrusion Detection System* (IDS). Sistem ini berupa *software* atau *hardware* yang bekerja dengan mendeteksi akses ilegal yang ditargetkan ke sistem komputer atau jaringan melalui internet. Dengan hal ini, menjadikan taraf keamanan sistem komputer atau jaringan yang ada lebih tinggi daripada umumnya. IDS terdiri dari beberapa tipe yang memiliki fungsi dan karakteristiknya masing-masing, seperti NIDS, PIDS, APIDS, HIDS, dan *Hybrid* IDS. Namun dalam *monitoring* sistem jaringan (*cloud*) atau komputer umumnya menggunakan HIDS atau NIDS. Sistem HIDS (*Host-Based Intrusion Detection System*) ini ditempatkan di dalam *host* untuk mendeteksi adanya *intrusions* yang didasarkan pada *analysis of system callings, application logs, file modification, host statements* dan lain-lainnnya[7]. Sedangkan NIDS (*Network-Based Intrusion Detection System*), ditempatkan di dalam unit jaringan tempat *server* berada atau di *entry point*. Meski memiliki fungsi yang sama, tetapi keduanya memiliki perbedaan utama pada objek target yang deteksi. NIDS umumnya memonitor keseluruhan bagian jaringan, sedangkan HIDS hanya memonitor pada *host* tertentu yang dinilai lebih krusial jika ada serangan terjadi[8].

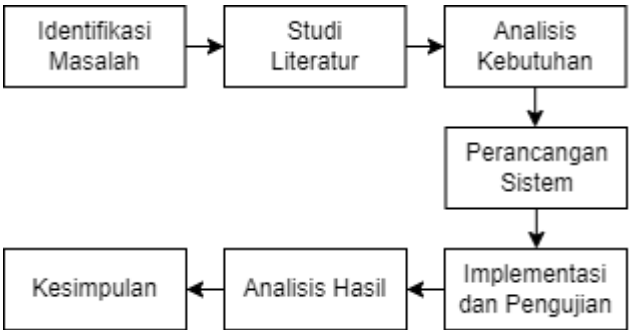
Sistem *cloud* yang bersifat virtual, tentu memiliki sebuah perangkat fisik di pusat data milik *cloud provider* yang menjadi *resource* dari *instance* atau *Virtual Machine* (VM) yang ada di dalamnya. Dalam melakukan *monitoring*, akan sangat sulit untuk melakukannya jika jauh di perangkat kerasnya, bahkan oleh seorang dari divisi SOC *cloud provider* tersebut. Hal ini umumnya berkenaan dengan kebijakan keamanan dan pekerjaan yang diterapkan. Maka dari itu, diperlukan sebuah sistem atau aplikasi yang memudahkan seorang IT *security* untuk memonitor keamanan sistem *cloud* dengan mudah walaupun tidak terhubung langsung ke perangkat keras. NSM (*Network Security Monitoring*), adalah sebuah sistem perlindungan yang bekerja berdasarkan prinsip *monitoring* lalu lintas yang melewati jaringan komputer dengan mengumpulkan data tentang lalu lintas tersebut dan menganalisisnya untuk diambil kesimpulan dan tindakan untuk perlindungan dari akses atau tindakan yang tidak sah[9]. Security Onion adalah salah satu *platform* NSM yang bersifat *open-source*. Dibekali dengan *pre-installed software* HIDS maupun NIDS, menjadikan Security Onion sebagai salah satu aplikasi NSM terpusat yang memudahkan pengguna memonitor sebuah sistem *cloud*. Aplikasi yang buat oleh Doug Burks sejak tahun 2008 ini sudah beberapa kali dipakai dalam penelitian, diantaranya : *Securing Infrastructure-as-a-Service Public Clouds Using* Security Onion[10]. *Intrusion Detection System and Intrusion Prevention System with* Snort *provided by* Security Onion[7]. Analisa dan Perancangan Keamanan Jaringan Lokal Menggunakan Security Onion dan MikroTik[11]. Security Onion Linux *distribucija i njezine primjene*[9].

Dari penelitian yang telah dilakukan sebelumnya, didapati hasil yang positif berupa kemudahan dengan sudah terinstalnya berbagai macam *software* pendukung *monitoring* keamanan jaringan yang dihimpun dalam satu sistem operasi. Penelitian sebelumnya juga hanya menggunakan Security Onion sebagai NSM dengan terbatasi pada penggunaan di *environment local* VM dan masih menggunakan versi lama (versi 1.6). Maka dari itu, dalam penelitian ini melanjutkan dengan digunakannya versi terbaru Security Onion 2 yang ditempatkan di *environment cloud* dan difokuskan pada *type attack* DoS yang sering menargetkan sistem *cloud*.

Tujuan penelitian ini adalah untuk menguji sistem deteksi dan *monitoring* yang terdapat pada NSM dan efektifitasnya dalam memberikan *alert* serangan DoS di *environment cloud* serta menganalisis hasil paketnya. Dengan ditempatkannya Security Onion pada *cloud* diharapkan dapat memudahkan dalam *monitoring* sistem *cloud* dan analisisnya ketika terjadi serangan DoS.

II. METODOLOGI PENELITIAN

Metode yang diterapkan pada penelitian ini adalah metode *experimental design*. Penerapan metode bertujuan untuk menganalisa Security Onion sebagai *Network Security Monitoring* pada *environment cloud* dalam mendeteksi serangan *denial of service* (DoS).



Gambar . 1 Diagram alur metode penelitian

Berikut merupakan alur tahapan yang dilakukan dalam penelitian ini:

Pada gambar 1, dapat diketahui tahapan penelitian yang akan di terapkan pada badan penelitian, yaitu:

1. Identifikasi masalah

   Tahapan awal untuk melakukan penelitian merupakan identifikasi masalah. Pada tahapan ini

permasalahan yang diidentifikasi mengenai kurangnya pengawasan terhadap serangan DoS pada *cloud*. Peneliti akan membuat rancangan dan implementasi *network security monitoring* yang dideploy pada *cloud* yang bertujuan untuk mendeteksi dan monitoring lalu lintas jaringan pada cloud terhadap serangan DoS secara real-time dengan efektif

2. Studi Literatur

   Peneliti mencari literatur yang relevan untuk menjadi referensi pendukung dalam penelitian. Literatur yang digunakan dalam penelitian ini berhubungan dengan penerapan *network security monitoring* yang didapat dari berbagai macam sumber seperti buku, artikel, jurnal nasional maupun internasional.

3. Analisis kebutuhan

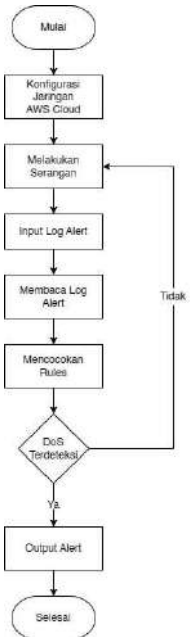
   Tahap ini bertujuan untuk mendapatkan pemahaman yang menyeluruh dan terperinci tentang kebutuhan utama sistem seperti yang didefinisikan agar tujuan tercapai, yang kemudian secara jelas didefinisikan, ditinjau dan disepakti bersama.

4. Perancangan system

   Perancangan dilakukan dengan menerapkan *network security monitoring* yang dideploy pada *cloud* agar dapat mendapatkan alert ketika terjadi tindakan yang tidak biasa dan dikumpulkan ke *log server* yang terdapat pada Security Onion. Berikut *flowchart* proses dari sistem yang akan dibuat:

5. Implementasi dan pengujian

   Pada tahap ini, rancangan sistem yang sudah dibuat akan diimplementasikan sesuai alur kerja sistem yang telah dibuat pada *environment cloud* agar lebih efisien. Instalasi dan konfigurasi dilakukan sesuai referensi dokumentasi dari masing-masing *software* yang digunakan.



Gambar . 2 Flowchart perancangan sistem

   Kustomisasi dari *software* yang ada mungkin diperlukan bergantung pada kebutuhan penelitian. Parameter pengujian yang akan dilakukan adalah menguji *cloud* dengan serangan DoS yang diterima dan dapat dijadikan sebagai alert notification pada network security monitoring untuk dianalisis.

6. Analisis hasil

   Pada tahap ini hasil dari implementasi dan pengujian sistem akan dianalisis untuk mengetahui jenis serangan DoS yang diterima dan sumber serangan, menggunakan *tools* yang tersedia di Security Onion.

7. Kesimpulan

   Tahapan terkahir dalam penelitian ini akan dilakukan penarikan kesimpulan dan pemberian saran dari seluruh penelitian yang telah dilakukan berdasarkan rumusan masalah yang telah dijelaskan sebelumnya. Hasil dari pengujian sistem dapat menjadi jawaban dari masalah yang telah dijelaskan serta data dari penelitian diharapkan berguna dalam sistem keamanan pada cloud dan dapat menjadi referensi penelitian berikutnya.

A. Analisis kebutuhan

Analisis kebutuhan merupakan analisis yang dibutuhkan untuk menentukan detail kebutuhan pada penelitian deteksi dan *monitoring* serangan DoS pada *cloud*. Penelitian ini mengintegrasi NSM Security Onion untuk memonitor *environment cloud*. Sehingga dibutuhkan sebuah perangkat yang dapat mendukung agar penelitian ini berjalan sesuai tujuan. Berikut merupakan kebutuhan yang dibagi menjadi beberapa bagian, yaitu :

1. Kebutuhan perangkat keras (*Hardware*)

   Perangkat keras yang diperlukan guna tujuan penelitian yaitu Laptop sebagai uji coba dengan spesifikasi berikut :

   Processor : AMD Ryzen 7 5700U
   RAM : 16 GB
   SSD : 256 GB
   Sistem Operasi : Windows 10 Home 64-bit

2. Kebutuhan perangkat lunak (*Software*)

   Perangkat lunak berfungsi untuk pengoperasian sistem pada penelitian ini. Pada penelitian ini *cloud instance* yang digunakan adalah *instance* pada Amazon Web Service (AWS).

Tabel . 1 Kebutuhan perangkat lunak

No	Nama Perangkat Lunak	Keterangan
1	Security Onion 2	Sistem operasi sekaligus aplikasi yang digunakan sebagai *network security monitoring* pada *cloud*
2	Amazon Web Service - Compute EC2	Layanan *cloud* yang digunakan sebagai tempat pengujian
3	Ubuntu WSL Terminal	Aplikasi yang dipergunakan untuk menghubungkan dengan *server*

		AWS melalui SSH *server*
4	Kali Linux	Digunakan sebagai sistem operasi penyerang
5	Amazon Linux	Digunakan sebagai sistem operasi target penyerangan (*victim*)

B. Perancangan system

Perancangan sistem dilakukan untuk membuat desain perencanaan arsitektur sistem yang akan dibangun dapat berjalan sesuai dengan tujuan penelitian.

1. Perancangan Spesifikasi *Cloud Instance*

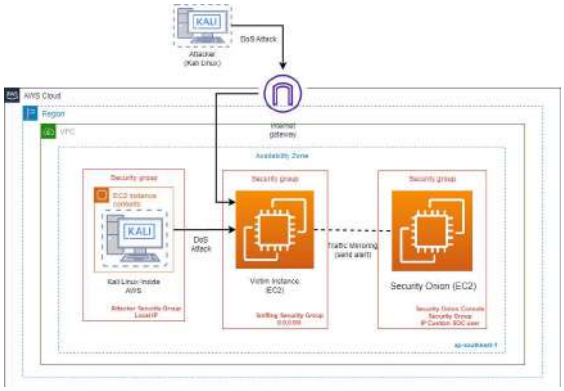
   Perancangan sistem pada penelitian ini secara keseluruhan menggunakan tiga *cloud instance*, yang terdiri dari satu Amazon Linux sebagai *victim* atau target serangan yang terhubung dengan Security Onion sebagai *network security monitoring* dan Kali Linux sebagai *penetration testing*.

Tabel . 2 Spesifikasi cloud instance

Instance	
Sistem Operasi	Security Onion 2
Processor	4
Memory	16 GB
Storage	200 GB
Rules	NSM
Public IP Address	18.136.221.173
Private IP Address	10.0.142.10 & 10.0.142.11
Instance	
Sistem Operasi	Amazon Linux
Processor	2
Memory	1 GB
Storage	8 GB
Rules	Victim
Public IP Address	18.143.211.107
Private IP Address	10.0.142.20
Instance	
Sistem Operasi	Kali Linux
Processor	2
Memory	4 GB
Storage	40 GB
Rules	Penetration Testing
IP Address	10.0.142.30

2. Perancangan Network Security Monitoring

   Perancangan sistem deteksi *network security monitoring* pada penelitian ini menggunakan Security Onion 2 sebagai IDS yang memonitor *network traffic* dan mendeteksi serangan *denial of service* (DoS) yang dilancarkan oleh kali linux sebagai penyerang. Penelitian ini menggunakan infrastruktur lab virtual yang fleksibel dan terpusat, di dalamnya ditempatkan tiga *instance cloud*, yaitu Security Onion (SO), Kali Linux, dan Amazon Linux. AWS *Elastic Cloud Compute* (EC2), sebagai jenis layanan cloud yang digunakan memiliki fleksibilitas tinggi yang dapat disesuaikan untuk menunjang performa tinggi.

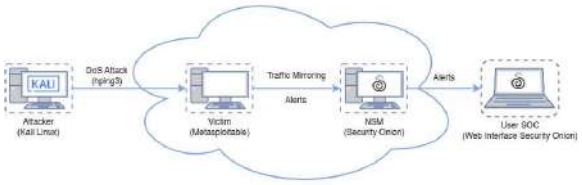


Gambar . 3 Flowchart perancangan sistem

Pada gambar 3 adalah sistem kerja deteksi NSM dengan Security Onion di AWS. Ketika terjadi aktivitas serangan DoS yang menargetkan Amazon Linux, Security Onion sebagai NSM-IDS akan melakukan *monitoring* untuk mendeteksi adanya serangan tersebut. Serangan yang terdeksi akan dikirimkan ke *alert log* yang kemudian akan dianalisis menggunakan Security Onion Console (SOC).

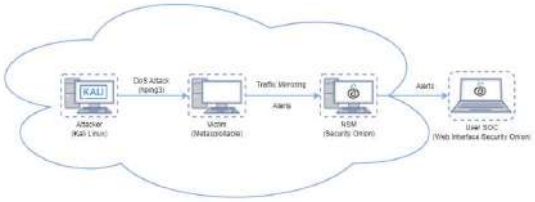
3. Perancangan Pengujian

   Perancangan pengujian pada penelitian ini menggunakan Kali Linux sebagai sistem operasi penyerang. Untuk metode serangan yang digunakan adalah *Denial of Service* (DoS) dengan jenis SYN flood, UDP flood, dan ICMP flood menggunakan tool hping3. Adapun serangan yang dilakukan dari luar jaringan AWS dan dari dalam *environment* AWS.



Gambar . 4 Flowchart perancangan sistem

Pada gambar 4 adalah alur pengujian serangan DoS yang dikirimkan ke *victim instance* dengan menggunakan tool hping3 dari Kali linux di luar *network* AWS. *Traffic* yang diterima oleh *victim instance* kemudian akan dikirimkan dengan *traffic mirroring* ke Security Onion instance. Traffic yang masuk dari *mirroring* inilah yang kemudian akan mendeteksi *alert* serangan dan ditampilkan pada *web interface* dari Security Onion *Console*.



Gambar . 5 Flowchart perancangan sistem

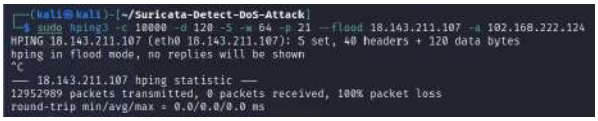
Pada gambar 5 adalah alur pengujian serangan DoS ke *victim instance* dengan hping3 di Kali linux dari dalam *local environment* AWS.

III. HASIL DAN PEMBAHASAN

Tahap pengujian yang dilakukan dalam penelitian ini dimaksudkan untuk mengetahui hasil deteksi *alerts* dari *rules* yang ditambahkan pada suricata pada Security Onion dalam *monitoring victim instance* di AWS. Pengujian menggunakan metode serangan Denial of Service (DoS) dengan *tool* hping3, yang dilakukan dengan serangan dari jaringan luar dan jaringan lokal AWS. Hping3 merupakan *tools* jaringan yang digunakan untuk mengirimkan *traffic* berupa *manipulated packets* melalui jaringan seperti TCP/UDP/ICMP. Jenis DoS yang digunakan adalah SYN *flood*, UDP *flood* dan ICMP *flood*.

1. Deteksi *Denial of Service* (DoS) dari Jaringan Luar AWS

   Pada pengujian ini, serangan dilakukan dengan melakukan DoS dengan hping3 dari jaringan luar AWS, untuk mengetahui kemampuan Security Onion dalam mendeteksi *alerts* dari luar jaringan.



Gambar . 6 Pengujian serangan SYN flood

Jenis serangan DoS yang pertama yaitu SYN *flood* dengan menggunakan *packet count* sebanyak 10.000 SYN *packets* dan 120 *data byte* yang ditujukan pada port 21. Tercatat sebanyak 12.952.989 *packets* yang berhasil ditransmisikan selama 5 menit pengujian seperti gambar 6.

Gambar . 7 Alert SYN flood

Pada gambar 7 merupakan serangan SYN *flood* yang berhasil terdeteksi sesuai dengan *rules* dan bersifat *medium severity. Alerts* serangan akan muncul ketika terjadi *inbound packets* sebanyak 3.000 dalam 10 detik. Pada pengujian pertama SYN *flood*, *alerts* yang berhasil dideteksi oleh suricata sebanyak 2 *alerts*.

Gambar . 8 Pengujian serangan UDP flood

Jenis serangan DoS yang selanjutnya dilakukan yaitu UDP *flood*. Seperti sebelumnya, serangan ini menggunakan setting *packet count* sebanyak 10.000 UDP *packets* dan menggunakan IP 192.168.12.42 sebagai *spoof source address*. Tercatat sebanyak 12.154.760 *packets* yang berhasil ditransmisikan selama 5 menit pengujian seperti gambar 8.

Gambar . 9 Alert UDP flood

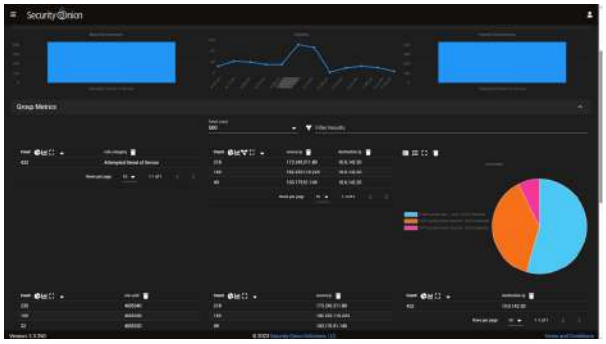
Pada gambar 9 merupakan *alerts* serangan yang terdeteksi pada pengujian pertama UDP *flood*, *alerts* yang berhasil dideteksi oleh suricata sebanyak 30 *alerts* dengan *medium severity*.

Gambar . 10 Pengujian serangan ICMP flood

Jenis serangan DoS terakhir yang dilakukan yaitu ICMP *flood* dengan menggunakan *setting packet count* sebanyak 10.000 ICMP *packets* dan menggunakan IP 102.168.232.122 sebagai *spoof source address*. Tercatat sebanyak 2.956.678 *packets* yang berhasil ditransmisikan selama 5 menit pengujian seperti gambar 10.

Gambar . 11 Alert ICMP flood

Pada gambar 11 merupakan *alerts* serangan yang terdeteksi pada pengujian pertama ICMP *flood*, *alerts* yang berhasil dideteksi oleh suricata sebanyak 74 *alerts* dengan *medium severity*.



Gambar . 12 Dashboard pengujian dari luar AWS

Pada gambar 12 merupakan tampilan *dashboard* dari *recap* pengujian serangan DoS dari luar *environment* AWS. Pada *dashboard* ini peneliti menggunakan *filter* serangan DoS sehingga data yang ditampilkan hanya terkait dengan *rules* DoS yang telah ditambahkan.

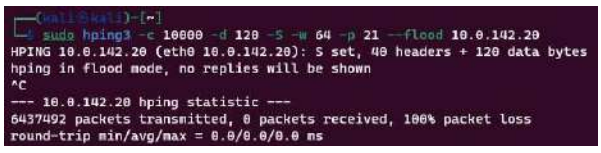
Tabel . 3 Pengujian DoS dari luar AWS

Jenis Pengujian	Pengujian ke-	Packet Transmitted	Jumlah Alert
SYN Flood	1	12.952.989	2
	2	6.553.769	30
	3	7.140.172	0
UDP Flood	1	12.154.760	30
	2	11.863.446	23
	3	11.951.153	24
ICMP Flood	1	2.956.678	74
	2	3.060.371	62
	3	3.609.562	62

Pada pengujian serangan DoS yang dijalankan dari luar jaringan AWS, dilakukan 3 kali pengujian pada tiap jenis DoS, dan menghasilkan hasil yang cukup variatif, seperti pada tabel 3. Dengan *threshold* "*count* 3.000" dan "*seconds* 10" yang diterapkan pada *rules*, Security Onion terbilang baik dalam mendeteksi *alerts* serangan dari luar AWS yang masuk ke *victim instance*.

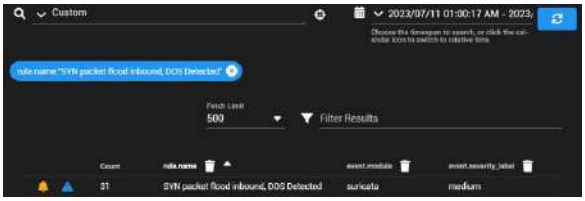
2. Deteksi *Denial of Service* (DoS) dari Jaringan Lokal AWS

Hampir sama dengan pengujian sebelumnya, pada serangan ini dilakukan dari jaringan lokal AWS untuk mengetahui kemampuan Security Onion dalam mendeteksi *alerts* dari dalam jaringan lokal dan melihat perbedaan hasilnya dengan jaringan luar. Tentunya target IP pada pengujian ini adalah *local* IP dari *victim instance*.



Gambar . 13 Local SYN flood

Pada pengujian serangan pertama, SYN *flood local attack* dengan *settings* 10.000 *packets count* dan 120 *data byte* pada *port* 21. Selama 5 menit pengujian, sebanyak 6.437.492 *packets* berhasil ditransmisikan, seperti gambar 13.

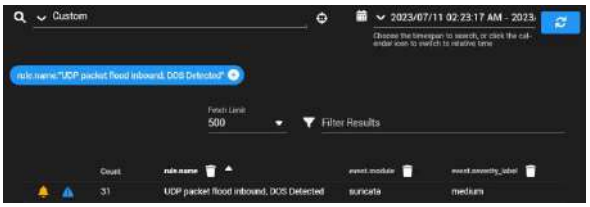


Gambar . 14 Alert local SYN flood

Pada gambar 14 merupakan *alerts* serangan yang terdeteksi pada pengujian pertama SYN *flood local attack*, sebanyak 31 *alerts* dengan *medium severity* berhasil di deteksi oleh suricata.

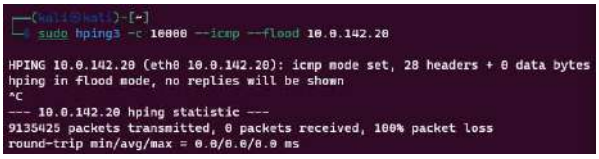
Gambar . 15 Local UDP flood

Jenis serangan DoS yang selanjutnya, yaitu UDP *flood*. Dengan menggunakan *setting* 10.000 *packets count*, sebanyak 8.838.950 *packets* berhasil ditransmisikan selama 5 menit pengujian seperti gambar 15. IP *spoofing* tidak digunakan karena serangan dilakukan dalam 1 jaringan lokal.



Gambar . 16 Alert local UDP flood

Pada gambar 16, *alerts* serangan yang terdeteksi oleh suricata pada pengujian UDP *flood local attack* sebanyak 31 *alerts* dengan *medium severity*.



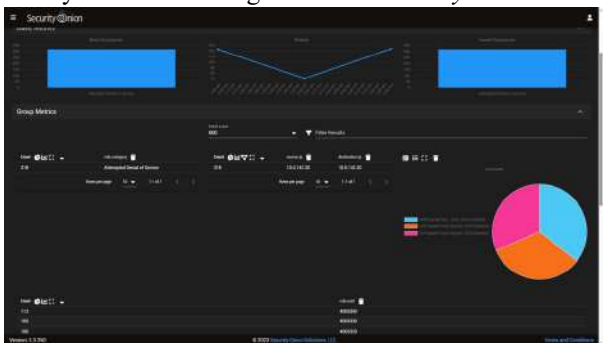
Gambar . 17 Local ICMP flood

Jenis serangan DoS yang terakhir, yaitu ICMP *flood*. Menggunakan *setting* 10.000 *packets count*, sebanyak

9.135.425 *packets* berhasil ditransmisikan selama 5 menit pengujian seperti gambar 17.

Gambar . 18 Alert local ICMP flood

Pada gambar 18, *alerts* serangan yang terdeteksi oleh suricata pada pengujian ICMP *flood local attack* sebanyak 31 *alerts* dengan *medium severity*.



Gambar . 19 Dashboard pengujian dari lokal AWS

Pada gambar 19 merupakan tampilan *dashboard* dari pengujian serangan DoS dari lokal *environment* AWS. Pada *dashboard* ini terlihat data *alerts* yang terdeteksi pada pengujian sangat berbeda jika dibandingkan dengan serangan dari luar.

Tabel . 4 Pengujian DoS dari lokal AWS

Jenis Pengujian	Pengujian ke-	*Packet Transmitted*	Jumlah *Alert*
SYN *Flood*	1	6.437.492	31
	2	6.532.446	31
	3	6.440.111	31
UDP *Flood*	1	8.838.950	31
	2	8.556.995	31
	3	9.287.614	31
ICMP *Flood*	1	9.135.425	31
	2	8.614.741	31
	3	7.994.637	31

Pada tabel 4 terlihat hasil dari ketiga jenis pengujian DoS yang dilakukan pada jaringan lokal memiliki jumlah *alerts* yang konsisten sama.

Tabel . 5 Hasil keseluruhan pengujian

Jenis Pengujian	Pengujian ke-	Parameter	Hasil Pengujian	
			Serangan Luar	Serangan Lokal
SYN Flood	1	Rules terdeteksi ketika terjadi serang-an	Sesuai	Sesuai
	2		Sesuai	Sesuai
	3		Tidak Sesuai	Sesuai
UDP Flood	1		Sesuai	Sesuai
	2		Sesuai	Sesuai
	3		Sesuai	Sesuai
ICMP Flood	1		Sesuai	Sesuai
	2		Sesuai	Sesuai
	3		Sesuai	Sesuai

Dari keseluruhan pengujian yang telah dilakukan, hanya pada pengujian ke-3 dari serangan luar AWS, *rules* tidak terdeteksi. Hal ini bisa terjadi karena efek dari serangan DoS ke *victim instance* membuatnya tidak bisa mengirimkan *logs traffic* untuk di *mirror*.

IV. KESIMPULAN DAN SARAN

Berdasarkan penelitian deteksi serangan *Denial of Service* (DoS) pada *cloud* menggunakan Security Onion yang telah berhasil dilakukan, kesimpulan yang didapatkan menunjukkan bahwa metode integrasi *monitoring* untuk mendeteksi serangan *Denial of Service* (DoS) yang diterapkan di AWS *cloud* dengan *traffic mirroring*, berhasil mengirim *alert* kepada Security Onion. Berdasarkan penerapan *rules* pada Suricata yang telah dibuat, Security Onion dapat mendeteksi *alert* sesuai dengan format kondisi yang ditentukan. *Alert* yang diterima Security Onion ketika terjadi serangan dari luar lebih variatif, dibandingkan serangan dari lokal yang menghasilkan *alert* lebih konsisten. Terdapat jeda waktu saat menampilkan *alert* deteksi serangan di Security Onion setelah pengujian pertama pada serangan dari luar maupun lokal.

Secara keseluruhan hasil yang didapatkan dalam pengujian menunjukkan bahwa Security Onion berfungsi dengan baik dalam mendeteksi serangan, namun masih terdapat beberapa poin yang bisa dikembangkan. Beberapa saran yang dapat diterapkan untuk pengembangan sistem deteksi serangan *Denial of Service* (DoS) pada *Cloud* menggunakan Security Onion, yaitu dengan mengembangkan limitasi pada *rules* yang diterapkan pada Suricata untuk mendapatkan hasil *alert* serangan yang lebih kompleks, menonaktifkan *rules* yang tidak terpakai untuk menghindari adanya *alerts flooding* atas serangan dinilai *false-positive*, mengkombinasikan notifikasi dari *alerts* dengan menerapkan fitur *Simple Notification Service* (SNS) pada AWS, yang diintegrasikan dengan Security Onion agar dapat menerima *alerts* melalui *email* atau SMS. Adapun VPN juga dapat digunakan pada tiap jenis serangan untuk mendapatkan IP berbeda dari perangkat penyerang, agar tidak terjadi *delay* atau pemblokiran IP oleh AWS ketika melakukan serangan.

REFERENSI

[1] D. Satrinia, S. N. Yutia, dan I. M. M. Matin, "Analisis Keamanan dan Kenyamanan pada Cloud Computing Dwina," *J. Informatics Commun. Technol.*, vol. 4, no. 1, hal. 1–7, 2022.

[2] Muqorobin, Z. Hisyam, M. Mashuri, Hanafi, dan Y. Setiyantara, "Implementasi Network Intrusion Detection System (NIDS) Dalam Sistem Keamanan Open Cloud Computing," *Maj. Ilm. Bahari Jogja*, vol. 17, no. 2, hal. 1–9, 2019, doi: 10.33489/mibj.v17i2.205.

[3] Sophos, "THE STATE OF CLOUD SECURITY 2020," *Sophos Whitepaper 2020*, no. July, Juli 2020.

[4] H. Tabrizchi dan M. K. Rafsanjani, "A survey on security challenges in cloud computing: issues, threats, and solutions," *J. Supercomput.*, no. 76, hal. 9493–9532, 2020.

[5] S. Mahjabin, "Implementation of DoS and DDoS Attacks on Cloud Servers," *Period. Eng. Nat. Sci.*, vol. 6, no. 2, hal. 148–158, 2018, doi: 10.21533/pen.v6i2.170.

[6] A. Toh, "Azure DDoS Protection-2021 Q3 and Q4 DDoS attack trends," 2022. https://azure.microsoft.com/en-us/blog/azure-ddos-protection-2021-q3-and-q4-ddos-attack-trends/

[7] B. Sergey, "Intrusion Detection System and Intrusion Prevention System with Snort provided by Security Onion .," *Bachelor's Thesis Inf. Technol. MAMK Univ. Appl. Sci.*, no. May, 2016.

[8] D. Santoso, A. Noertjahyana, dan J. Andjarwirawan, "Implementasi dan Analisa Snort dan Suricata Sebagai IDS dan IPS Untuk Mencegah Serangan DOS dan DDOS," *J. Infra*, vol. 10, no. 1, hal. 1–6, 2022, [Daring]. Tersedia pada: https://publication.petra.ac.id/index.php/teknik-informatika/article/view/12033

[9] D. Bošnjak, "Security Onion Linux distribution and its applications," JOSIP JURJA STROSSMAYER UNIVERSITY, 2019. [Daring]. Tersedia pada: https://urn.nsk.hr/urn:nbn:hr:200:263185

[10] A. Mikail dan B. Pranggono, "Securing infrastructure-as-a-service public clouds using security onion," *Appl. Syst. Innov.*, vol. 2, no. 6, hal. 1–17, 2019, doi: 10.3390/asi2010006.

[11] F. Yohaness dan Haerudin, "Analisa Dan Perancangan Keamanan Jaringan Lokal Menggunakan Security Onion Dan Mikrotik," *J. Inf. Syst. Technol.*, vol. 01, no. 02, hal. 37–61, 2020, [Daring]. Tersedia pada: https://journal.uib.ac.id/index.php/joint/article/view/4309

.