



INSTITUT TEKNOLOGI DEL

Centralization Data of Signature Based and Anomaly Based  
on Intrusion Detection System

TUGAS AKHIR II

13320012 Karyn Leoni Manik  
13320023 Paul Martin Parsaulian Nainggolan  
13320039 Antonel Basa Manurung

**FAKULTAS VOKASI  
PROGRAM STUDI DIII TEKNOLOGI KOMPUTER  
LAGUBOTI  
JUNI 2023**



INSTITUT TEKNOLOGI DEL

Centralization Data of Signature Based and Anomaly Based  
on Intrusion Detection System

TUGAS AKHIR II

Diajukan sebagai salah satu syarat untuk memperoleh gelar A.Md.T

13320012	Karyn Leoni Manik
13320023	Paul Martin Parsaulian Nainggolan
13320039	Antonel Basa Manurung

FAKULTAS VOKASI  
PROGRAM STUDI DIII TEKNOLOGI KOMPUTER

## ABSTRAK

Dalam era yang semakin terhubung secara digital, perlindungan terhadap sistem komputer dan jaringan menjadi sangat penting. Untuk melakukan pencegahan terhadap penyerangan tersebut, digunakan tools IDS. IDS (Intrusion Detection System) adalah mekanisme yang digunakan untuk mendeteksi serangan terhadap sistem dan jaringan. Dalam pengerjaan Tugas Akhir ini bertujuan untuk melakukan sistem sentralisasi terhadap dua metode dengan menggunakan dua tools yang berbeda untuk mengetahui bagaimana sistem sentralisasi tersebut dapat bekerja. Metode dan Tools yang kami gunakan dalam sistem central ini adalah metode Anomaly dengan tools suricata dan metode Signature dengan tools snort. Untuk snort dan suricata memiliki tugas yang serupa yaitu perangkat lunak yang digunakan untuk mendeteksi adanya threat yang masuk ke dalam sistem jaringan. Bentuk sentralisasi dari kedua metode tersebut ialah dimana log yang diterima dari kedua metode tersebut akan disatukan ke dalam satu folder kemudian diolah dan disimpan ke database. Dimana hasil dari data pengujian akan ditampilkan Interface aplikasi web.

Kata kunci:

IDS, Snort, Suricata, Anomaly, Signature.

## DAFTAR ISI

ABSTRAK.....	3
DAFTAR ISI .....	4
DAFTAR TABEL .....	7
DAFTAR GAMBAR.....	8
BAB I PENDAHULUAN .....	9
1.1 Latar Belakang .....	9
1.2 Tujuan .....	14
1.3 Research Questions .....	14
1.4 Batasan Penelitian .....	14
1.5 Expected Result .....	15
1.6 Sistematika Penyajian .....	15
BAB II TINJAUAN PUSTAKA .....	16
2.1 Landasan Teori.....	16
2.1.1 Keamanan Jaringan.....	16
2.1.2 Intrusion Detection System .....	18
2.1.3 Arsitektur Intrusion Detection System .....	20
2.1.4 Komponen Intrusion Detection System.....	21
2.1.5 Rules Intrusion Detection System .....	23
2.1.6 Metode Intrusion Detection System .....	26
2.1.7 Snort dan Suricata.....	30
2.1.8 Kelebihan dan Kelemahan Intrusion Detection System .....	33
2.1 Related Work .....	33
BAB III ANALISIS DAN DESAIN .....	35
3.1 Analisis .....	35
3.1.1 Analisis Masalah.....	35
3.1.2 Analisis Pemecahan Masalah .....	37
3.1.3 Analisis Kebutuhan Sistem.....	38
3.1.4 Analisis Sistem .....	40
3.2 Perancangan Pembangunan Sistem.....	41
3.2.1 Perancangan Umum Sistem.....	42
3.2.2 Perancangan Arsitektur Sistem.....	43
3.2.3 Topologi.....	44

3.2.4	Perancangan Web System .....	46
3.3	Flowchart Penelitian .....	48
3.3.1	Flowchart Signature Base .....	49
3.3.2	Flowchart Anomaly Base .....	49
3.3.3	Flowchart Centralization Signature dan Anomaly .....	50
3.4	Eksperimen Pengujian .....	51
3.4.1	Eksperimen Pengujian Denial Of Service .....	52
3.4.2	Eksperimen Pengujian Port Scanning.....	54
3.4.3	Eksperimen Pengujian SQL Injection .....	57
4.1	Instalasi dan Konfigurasi .....	60
4.1.1	Instalasi dan Konfigurasi Snort .....	60
4.1.2	Instalasi dan Konfigurasi Suricata .....	64
4.1.3	Instalasi dan Konfigurasi Web Server.....	67
4.2	Sentralisasi Data.....	70
4.2.1	Sentralisasi Data dari Snort (Signature) & Suricata (Anomaly).....	70
4.3	Visualisasi Monitoring Snort dan Suricata .....	74
4.3.1	Visualisasi Data Monitoring Snort dan Suricata .....	75
4.4	Pengujian.....	77
4.4.1	Pengujian Serangan Denial Of Service .....	77
4.4.2	Pengujian Serangan Port Scanning.....	78
4.4.3	Pengujian Serangan SQL-Injection(DVWA) .....	79
4.5	Hasil Pengujian .....	81
4.5.1	Hasil Pengujian Serangan Denial Of Service .....	81
4.5.2	Hasil Pengujian Serangan Port Scanning .....	82
4.5.3	Hasil Pengujian Serangan SQL-Injection.....	82
BAB V KESIMPULAN .....		83
5.1	Kesimpulan .....	83
5.2	Saran .....	83
Bab II Daftar Pustaka dan Rujukan.....		84
LAMPIRAN .....		88
A.	Kode Program dari Tampilan Monitor .....	88
B.	Kode Program dari Tampilan Rules Snort .....	103
C.	Kode Program dari Tampilan Rules Suricata .....	112



## DAFTAR TABEL

Table 1 Komponen Penyusun IDS .....	23
Table 2 Perbandingan Mekanisme Kerja IDS .....	25
Table 3 Kelebihan dan Kelemahan IDS .....	33
Table 4 Related Work.....	34
Table 5 Kebutuhan Hardware.....	39
Table 6 Kebutuhan Software .....	40
Table 7 Analisis Sistem IDS.....	41
Table 8 Eksperimen Pengujian Denial Of Service .....	54
Table 9 Eksperimen Pengujian Port Scanning .....	57
Table 10 Eksperimen Pengujian SQL-Injection DVWA .....	59

## DAFTAR GAMBAR

Gambar 1 Kategori Persebaran Kebocoran Data Indonesia .....	10
Gambar 2 Jumlah Anomali Nasional Tahun 2021 .....	11
Gambar 3 Intrusion Detection System.....	19
Gambar 4 Arsitektur IDS.....	21
Gambar 5 Komponen Penyusunan IDS.....	22
Gambar 6 Mekanisme Signature Based.....	27
Gambar 7 Mekanisme Anomaly Based .....	29
Gambar 8 Perancangan Umum Sistem.....	42
Gambar 9 Perancangan Arsitektur Sistem.....	43
Gambar 10 Topologi Physical .....	44
Gambar 11 Topologi Logical .....	45
Gambar 12 Web Design Monitoring Suricata & Snort .....	46
Gambar 13 Design Tampilan Rules Suricata .....	47
Gambar 14 Design Tampilan Rules Snort.....	48
Gambar 15 Flowchart Signature Base .....	49
Gambar 16 Flowchart Anomaly Base .....	50
Gambar 17 Flowchart Sentralisasi Data .....	50
Gambar 18 Visualisasi Monitor Snort & Suricata.....	75
Gambar 19 Visualisasi Rules Snort .....	76
Gambar 20 Visualisasi Rules Suricata.....	77
Gambar 21 Penyerangan Denial Of Service.....	78
Gambar 22 Pendeteksian DoS oleh Snort.....	78
Gambar 23 Pendeteksian DoS oleh Suricata .....	78
Gambar 24 Penyerangan Port Scanning .....	79
Gambar 25 Pendeteksian Port Scanning oleh Snort .....	79
Gambar 26 Pendeteksian Port Scanning oleh Suricata.....	79
Gambar 27 Penyerangan SQL-Injection DVWA .....	80
Gambar 28 Pendeteksian SQLInjection DVWA oleh Snort .....	80
Gambar 29 Pendeteksian SQL-Injection DVWA oleh Suricata.....	81



## **BAB I**

### **PENDAHULUAN**

Bab ini menjelaskan latar belakang dari tugas akhir, tujuan, lingkup, pendekatan yang dilakukan, dan sistematika penyajian tugas akhir yang berjudul *Centralization Data of Signature Based and Anomaly Based on Intrusion Detection System*

#### **1.1 Latar Belakang**

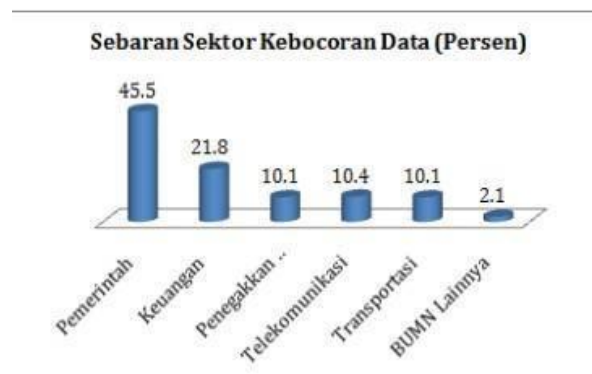
Meningkatnya perkembangan internet dan jaringan komputer terkhusus pada bagian jaringan komputer dan layanan-layanan memiliki dampak positif salah satunya dapat mempermudah pekerjaan-pekerjaan manusia dalam kehidupan, namun terdapat salah satu problem yakni dalam bidang keamanannya, pada saat ini statis peristiwa keamanan meningkat tajam. Terdapat juga serangan yang berusaha menyerang dan berusaha untuk mencari suatu rubrik yang terjadi dalam keamanan jaringan yang digunakan [1]. Terjadinya hal tersebut dikarenakan ketertarikan terhadap pertukaran informasi masih minim. Dengan terjadinya hal tersebut maka dengan itu diperlukan sistem keamanan jaringan yang mampu menemukan dan menangkal suatu tindakan penyusupan di jaringan yang berasal dari pihak- pihak yang tidak bertanggung jawab.

Untuk mencegah berbagai potensi serangan memasuki jaringan sistem dan menyebar, keamanan harus dapat menghentikannya. Intrusion atau serangan yang melanggar hukum adalah apa yang dikenal sebagai serangan atau segala sesuatu yang tidak diinginkan yang mempengaruhi ketersediaan, privasi, atau integritas informasi yang disimpan dalam suatu sistem[2].

Dalam tahun 2020 yang terdapat dalam laporan data anomali trafik BSSN (2021), Negara Indonesia mengalami serangan siber hingga angka yang fantastis sebesar 495,3 juta atau dengan kata lain meningkat 41% dibandingkan pada tahun 2019

dengan jumlah 290,3 juta. Dengan total 7.311.606 anomali, 10 Desember 2020 lalu lintas paling banyak anomali. Trojan tumbuh menjadi outlier dalam hal kuantitas. Negara sumber anomali dengan jumlah serangan terbesar pada tahun 2020 adalah Amerika Serikat. Serta Indonesia adalah negara dengan jumlah serangan terbanyak yang disebabkan oleh anomali yang berasal dari Indonesia (dengan alamat IP Indonesia). Survei tersebut juga mengidentifikasi 79.439 akun yang mengalami pelanggaran data-data sebesar 9.749 kasus deface online, dan 2.549 kasus email phishing, dengan peningkatan kasus tersebut terjadi antara Maret dan Mei 2020. Sektor akademik muncul sebagai sektor dengan kasus tertinggi pada tahun 2020 [4]

Traffic anomali dan serangan siber mencapai 741,4 juta antara Januari dan Juli 2021, dengan malware, penolakan layanan (mengganggu ketersediaan layanan), dan aktivitas trojan menjadi kategori yang paling anomali[5]. Serangan Ransomware (malware yang menuntut tebusan) dan kebocoran data indeks adalah jenis serangan siber (kebocoran data) yang paling umum. Dengan sebaran 45,5%, sektor pemerintah paling banyak mengalami kebocoran data akibat malware pencuri informasi selama ini. Sektor keuangan (21,8%), telekomunikasi (10,4%), penegakan hukum (10,1%), transportasi (10,1%), dan BUMN lainnya (2,1%) adalah sektor yang paling terpengaruh berikutnya. Gambar 1 menampilkan distribusi sektor kebocoran data dan grafik anomali trafik.

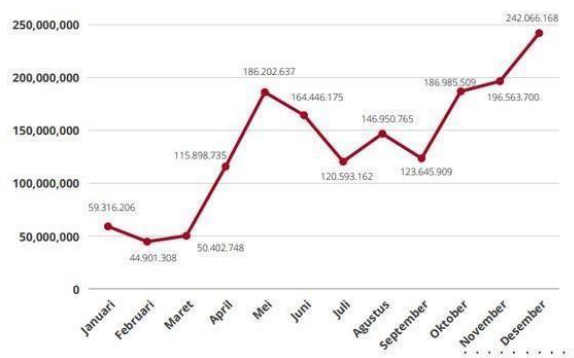


Gambar 1 Kategori Persebaran Kebocoran Data Indonesia

(Sumber : “Laporan Tahunan Monitoring Keamanan Cyber” BSSN, 2021)

Berdasarkan gambar dibawah, anomali mengalami pertumbuhan serangan siber meningkat setiap bulan dari Januari hingga Mei, kecuali Mei dan Juni yang mengalami penurunan sebelum meningkat lagi. Sedangkan komputasi awan adalah target utama serta dapat diyakini bahwa itu akan menyerang sistem yang paling mungkin menampung data sensitif kami. Berdasarkan studi yang dilakukan oleh BSSN pada tahun 2020, yang mengungkapkan bahwa komputasi awan menghadapi ancaman serangan siber sebesar 56% dari eksternal dan 36% dari internal, komputasi awan memiliki kemungkinan yang tinggi untuk menghadapi serangan siber. Terkait tersebut menunjukkan perlunya mempertimbangkan faktor internal dan eksternal saat memikirkan keamanan secara keseluruhan [6]. Selain itu, fungsi kesadaran keamanan siber dari sumber daya manusia atau staf yang mengawasi komputasi awan itu sendiri perlu diperhatikan.

**Jumlah Anomali Nasional pada 2021**



Gambar 2 Jumlah Anomali Nasional Tahun 2021

(Sumber : “Laporan Tahunan Monitoring Keamanan Cyber” BSSN, 2021)

Intrusion detection system (IDS) ditemukan dalam firewall. Dalam proses pencegahan untuk merendahkan terjadinya serangan pada sistem maka hal yang

dilakukan membutuhkan sistem yang dapat mengidentifikasi aktivitas jaringan yang dicurigai dan memberikan sebuah peringatan sehingga dapat dengan cepat mengambil tindakan untuk memberhentikan serangan tersebut. Software atau hardware yang melakukan fungsinya dengan cara otomatis yang dimana untuk memonitor sebuah peristiwa yang ada dalam jaringan komputer serta sehingga dapat menganalisis masalah pada keamanan jaringan merupakan pengertian dari Intrusion Detect System [7]. Suatu kegiatan yang dimana untuk mendeteksi penjahat secara cepat dengan teknik program yang secara otomatis dan dengan respons yang real time[8]. Dalam proses pendeteksian adanya aktivitas yang tidak berjalan seperti apa adanya, maka selanjutnya mengirimkan informasi terhadap administrator secara otomatis dan menampilkan permasalahan yang tersedia. Selain itu, mempunyai sebuah kemampuan untuk mengawasi dan mengontrol integritas file, dikarenakan untuk dapat menaungi kode jahat yang merupakan usaha untuk terjauh dari serangan berbahaya

Oleh karena itu, dibutuhkan suatu system dalam menangani penyalahgunaan jaringan atau ancaman yang terjadi yaitu menggunakan IDS. IDS (Intrusion Detection System) adalah sebuah aplikasi perangkat lunak atau perangkat keras yang dapat mendeteksi aktivitas yang mencurigakan dalam sebuah sistem atau jaringan. IDS digunakan untuk mendeteksi aktivitas yang mencurigakan dalam sebuah sistem atau jaringan. IDS akan memonitor lalu lintas data pada sebuah jaringan atau mengambil data dari berkas log. IDS akan menganalisa dan dengan algoritma tertentu akan memutuskan untuk memberi peringatan kepada seorang administrator jaringan atau tidak[9].

Dalam mendeteksi sebuah serangan yang terjadi dalam jaringan komputer IDS menggunakan dua teknik yaitu dengan menggunakan teknik signature based dan anomaly based. Anomaly based merupakan teknik yang berfokus pada pengamatan perilaku system. Keadaan normal sistem sebelumnya dipantau dan kemudian dibandingkan secara terus-menerus dengan keadaan sistem saat ini untuk mendeteksi

penyimpangan dari keadaan normal. Perilaku atau penyimpangan yang tidak normal dianggap sebagai ketidaknormalan. Kelebihan dari pendekatan anomaly ialah lebih baik dalam mendeteksi serangan baru dan tidak diketahui. Namun teknik ini memiliki kelemahan fatal karena dapat menghasilkan banyak false positive karena aktivitas yang tidak biasa mungkin tidak selalu menunjukkan serangan.

Berbeda dengan IDS berbasis anomaly, Signature based berfokus pada pembuatan aturan dan perilaku yang telah ditentukan sebelumnya dalam database untuk mendeteksi semua serangan yang terekam. Aturan kemudian terus menerus dibandingkan dengan keadaan sistem saat ini untuk menemukan serangan yang ditentukan. Kelebihan dari pendekatan signature ialah lebih baik dalam mendeteksi serangan yang telah diketahui sebelumnya dan memiliki tanda-tanda khusus. Namun memiliki kelemahan utama dari teknik ini adalah ketidakmampuan untuk mendeteksi serangan yang baru ditemukan. Batasan IDS berbasis signature based adalah kegagalan untuk mengidentifikasi serangan baru, dan kadang-kadang bahkan variasi kecil pola yang diketahui [10]

Saat ini, penulis menyadari bahwa belum ada pengembangan sistem yang mengcentralisasi data untuk keamanan jaringan dengan mengintegrasikan pendekatan berbasis Anomaly dan Signature. Centralisasi data antara pendekatan berbasis Anomaly dan Signature dilakukan untuk memanfaatkan kelebihan dan mengatasi kelemahan dari masing-masing pendekatan dalam mendeteksi serangan pada sistem komputer atau jaringan. Dengan demikian, data yang dihasilkan dari teknik berbasis Signature dan Anomaly dapat dicentralisasi sehingga kedua pendekatan tersebut dapat memberikan deteksi yang lebih akurat dan luas. Hal ini memungkinkan penanganan serangan yang lebih kompleks, serta respons yang lebih cepat dengan menggabungkan kekuatan dari kedua pendekatan tersebut.

## **1.2 Tujuan**

Tujuan yang akan diperoleh berdasarkan latar belakang yang diketahui adalah :

1. Membantu administrator dalam memonitoring hasil sentralisasi log Anomaly based dan Signature based pada Intrusion Detection System.

## **1.3 Research Questions**

Berdasarkan latar belakang yang telah dijelaskan sebelumnya, maka dengan itu rumusan masalah ialah sebagai berikut

1. Bagaimana proses sentralisasi data signature dan data anomali dapat dilakukan secara efisien dan akurat dalam sistem deteksi intrusi?

## **1.4 Batasan Penelitian**

Untuk menghindari perluasan materi pembahasan tugas akhir ini, dengan itu terdapat batasan masalah, sebagai berikut:

1. Sistem/jaringan yang akan dilakukan dalam hal ini dengan Virtual Machine dengan menggunakan Operating System Ubuntu baik secara server dan penyerang.
2. Tools yang digunakan dalam Anomaly Intrusion Detection System ialah suricata, dan dalam Signature Intrusion Detection System ialah snort.
3. Attacker, IDS, Target yang digunakan pada penelitian akan dijalankan pada virtual machine.
4. Masing-masing metode Anomaly dan Signature yang akan diuji berdasarkan attack yang sama serta yang dijadikan parameter dalam mendeteksi sebuah serangan

## **1.5 Expected Result**

Hasil yang diharapkan dalam pengerjaan Tugas Akhir ini adalah mendapatkan kesimpulan yang berasal dari analisis hasil dari pengujian pengukuran yang tepat dideteksi serta mengintegrasikan kedua metode Intrusion Detection System.

## **1.6 Sistematika Penyajian**

Secara garis besar dokumen Tugas Akhir ini dibagi menjadi 6 bab yang disusun secara berkesinambungan dan sistematis antara lain sebagai berikut:

### **❖ BAB I PENDAHULUAN**

Pada bab ini berisi penjelasan dari proyek yang akan dibangun, meliputi latar belakang, tujuan, lingkup atau cakupan, pendekatan, dan sistematika penyajian dokumen ini dibuat.

### **❖ BAB II TINJAUAN PUSTAKA**

Pada bab ini berisi uraian setiap dasar teori yang berhubungan dengan Intrusion Detection System, Metode S-IDS (Signature) dan Metode A-IDS (Anomaly)

### **❖ BAB III ANALISIS DAN PERANCANGAN SISTEM**

Pada bab ini berisi tentang analisis yang dilakukan dan perancangan sistem yang sedang dibangun.

### **❖ BAB IV IMPLEMENTASI & HASIL PENGUJIAN**

Pada bab ini berisi implementasi dan pengujian terhadap proyek atau sistem yang sudah dibangun serta pengujian terhadap sistem yang dibangun serta pembahasan mengenai hasilnya.

### **❖ BAB VI KESIMPULAN DAN SARAN**

Pada bab ini berisikan kesimpulan serta saran-saran yang diperlukan dalam pengembangan selanjutnya.

## **BAB II**

### **TINJAUAN PUSTAKA**

Pada bab ini dideskripsikan tinjauan pustaka yang akan digunakan sebagai dasar teori dalam penyusunan Tugas Akhir.

#### **2.1 Landasan Teori**

Elemen Ikhtisar yang diperlukan dalam sentralisasi data Intrusion Detection System (Signature- Anomaly) akan dideskripsikan pada bagian ini.

##### **2.1.1 Keamanan Jaringan**

Mencegah dan mengidentifikasi penggunaan yang tidak baik dari sebuah jaringan komputer. Keamanan jaringan komputer ini memiliki tujuan sebagai teknikantisipasi resiko terhadap jaringan komputer yang seperti ancaman fisik baik secara logika langsung atau tidak langsung. Terdapat 3 konsep keamanan jaringan dalam jaringan komputer [13] yakni sebagai berikut

1. Resiko atau tingkat bahaya (Risk)
2. Ancaman (Threat)
3. Kerapuhan sistem (Vulnerability)

Proses perdana ketika dalam mengembangkan rencana keamanan jaringan yang baik dan efektif adalah dengan cara mengenali ancaman yang akan datang. Pada buku Site Security Handbook, memaparkan bahwasanya terdapat tiga tipe ancaman [14] yaitu

1. Akses tidak sah bagi orang yang terdapat wewenang
2. Kesalahan informasi, segala masalah yang dapat menimbulkan informasi yang sensitif terhadap orang yang salah, dan tidak berhak mendapatkan



informasi tersebut.

3. Penolakan terhadap layanan, setiap masalah berkaitan dengan security yang menimbulkan sistem yang dapat mengganggu pekerjaan-pekerjaan yang produktif

Berdasarkan dari fungsi sistem komputer yang menjadi penyedia informasi, maka ancaman yang terjadi dalam sistem komputer dibedakan menjadi empat yakni:

1. Interruption, ancaman terhadap availability, informasi atau data yang ada di dalam komputer dirusak ataupun dihapus, maka dengan itu jika diperlukan maka sudah tidak ada lagi. Interception, ancaman terhadap kerahasiaan (secrecy). Informasi yang terdapat dalam sistem dapat diakses oleh yang tidak berhak
2. Modification, ancaman terhadap integritas. Berhak menyadap kegiatan lalu lintas informasi yang sedang dikirim kemudian mengubahnya berdasarkan keinginan pribadi
3. Fabrication, ancaman terhadap integritas namun yang membedakan dari modification yakni pada hal ini sukses meniru atau memalsukan informasi dan dengan itu orang yang mendapatkan informasi tersebut menyangka informasi berasal dari orang yang dikehendaki oleh penerima informasi.[15]

Serangan yang terjadi dalam jaringan komputer berupa pengacauan, penyusupan atau serangan lainnya dalam infrastruktur jaringan untuk menganalisa jaringan kemudian mendapatkan informasi secepatnya. Ancaman tersebut seperti objek, manusia ataupun kejadian yang jika tercapai dapat menimbulkan kerusakan pada jaringan. Terdapat sejumlah serangan yang terjadi pada jaringan komputer yakni DOS, Spoofing Attack, port scanning, application based attack serta malware

- DoS (Denial Of Service)

Serangan kepada komputer atau server dalam jaringan. Cara kerja DoS adalah menghabiskan resource yang dimiliki oleh komputer sampai akhirnya komputer tersebut tidak menjalankan fungsinya dengan benar yang secara tidak langsung mencegah penggunaan lain agar mendapatkan akses layanan dari komputer yang diserang tersebut.[16]

- Port Scanning

Kegiatan suatu proses agar dapat mencari dan mendapatkan serta menelusuri port dalam suatu komputer dengan tujuan meneliti kelemahan suatu program yang tampil dalam komputer yang berasal dari port yang terbuka[18]

- SQL Injection

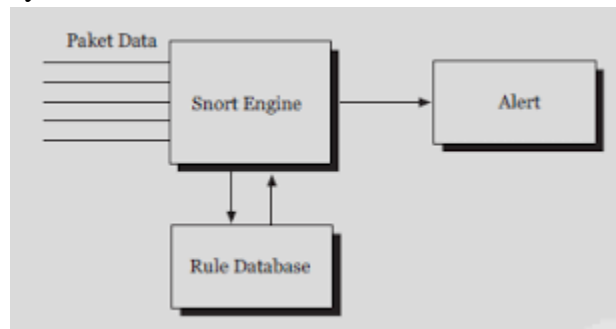
Karena kelemahan ini, penyerang sekarang dapat mengontrol kueri SQL yang dikirimkan dari aplikasi ke database. Penyerang memiliki kapasitas untuk memberikan pengaruh. Sistem operasi yang dialokasikan untuk sistem operasi database dipengaruhi oleh sintaks SQL, kekuatan, dan fleksibilitas database pendukung.

Aplikasi web serta semua program lain yang menggunakan frase SQL dipengaruhi oleh SQL Injection. Semua program menggunakan input dinamis dari luar, yang tidak terpercaya dan rentan terhadap serangan SQL[19].

### **2.1.2 Intrusion Detection System**

Intrusion Detection System (IDS) adalah dasar dalam pemaparan deteksi intrusi ketika proses monitoring komputer atau jaringan dalam suatu kegiatan yang tidak sah dengan konsep yang berbasis cyber pada jaringan komputer dengan

menyediakan kerangka kerja serta didasari oleh hipotesis bahwa pelanggaran keamanan mudah dideteksi dengan teknik melihat hasil penulisan audit sistem dengan tujuan pola abnormal system. [21]



Gambar 3 Intrusion Detection System

Menurut Ariyus bahwasanya beberapa alasan untuk memperoleh serta menggunakan IDS (Intrusion Detection System) yakni sebagai berikut

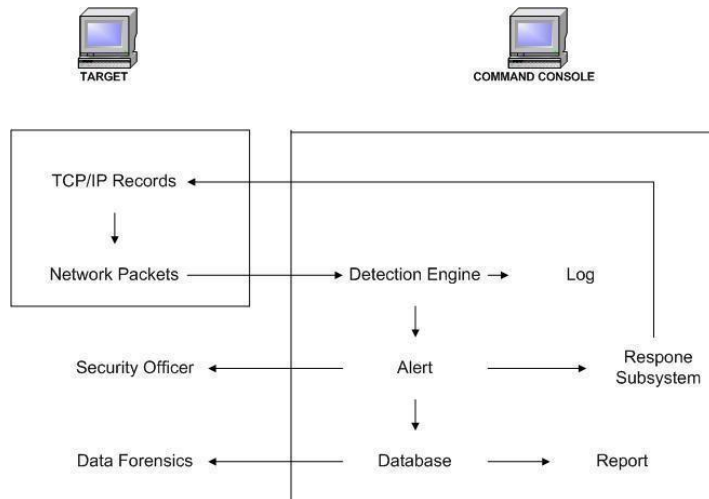
- a. Mencegah resiko keamanan yang sangat naik, hal ini terjadi dikarenakan banyak ditemukan kegiatan ilegal yang dilakukan oleh manusia yang tidak bertanggung jawab
- b. Mendeteksi serangan dan pelanggaran keamanan jaringan yang tidak dapat dicegah oleh sistem umum seperti firewall, oleh karena itu menimbulkan banyak lubang keamanan, seperti :
  - Dalam legacy system, sistem operasi tidak patch maupun update
  - Patch tidak diperhatikan dengan baik, sehingga mendapatkan suatu permasalahan baru dalam hal keamanan.
  - User yang tidak paham terhadap sistem oleh karena jaringan dan protokol yang dipergunakan mempunyai lubang keamanan

Menurut Ariyus terdapat beberapa fungsi Intrusion Detection System yakni sebagai berikut:

- Memantau kegiatan user dan sistem
- Mengobservasi konfigurasi sistem dengan tujuan terhadap kerentanan dan ketidak terhubung konfigurasi
- Menghitung integritas yang diperoleh dari critical system
- Meneliti pola serangan dari kegiatan sistem
- Mengidentifikasi aktivitas yang tidak normal melalui analisis statistik
- Fokus terhadap jejak pelanggaran user dari aturan kebijakan
- Memperbaiki kesalahan dari konfigurasi sistem
- Install dan menjalankan jebakan untuk mengumpulkan dan membukukan informasi terkait attacker

### **2.1.3 Arsitektur Intrusion Detection System**

Sistem deteksi intrusi membuat sebuah pendekatan yang memiliki basis jaringan dan terdapat host yang berguna untuk mengenali maupun menyimpangkan serangan. Kemudian IDS menelusuri tanda serangan dengan pola tertentu, yang pada umumnya untuk mengidentifikasi niat buruk serta mencurigakan. Pada saat sebuah IDS menyaring pada pola-pola tertentu dalam lalu lintas jaringan oleh karena itu IDS berbasis jaringan (NIDS). Pada saat IDS menelusuri tanda serangan yang berada dalam file log, oleh karena itu IDS tersebut basis host. Pada gambar dibawah ini menampilkan sensor berbasis arsitektur jaringan standar yang mendeteksi intrusi. Sebuah sensor berfungsi untuk mencari paket dari jaringan yang pada umumnya mereka di input kedalam mesin deteksi dan hal tersebut menampilkan alarm ketika terdapat salah satu penyelandupan. Kemudian sensor tersebut mengirimkan ke berbagai mission-critical segmen jaringan. Mengumpulkan alarm dari beberapa sensor ialah fungsi dari pusat kontrol [22]



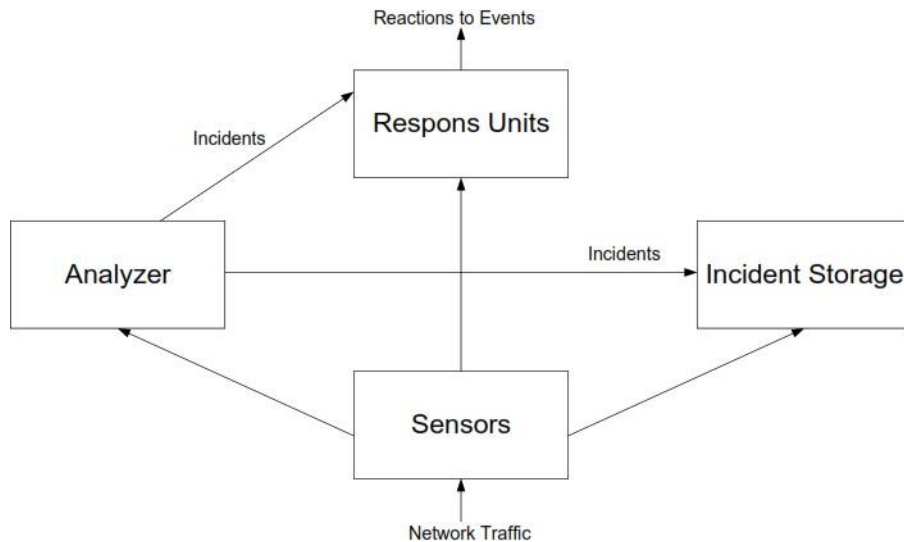
Gambar 4 Arsitektur IDS

Dibuatnya paket jaringan terjadi ketika pada saat komputer berada dalam jaringan

- Sistem ini bertujuan untuk mendeteksi serangan dari intrusi, ketika pola terdeteksi dan kemudian mendapatkan hasil peringatan
- Peringatan keamanan akan ditampilkan dalam penyalahgunaan
- Pendapat terkait intrusi yang ditemukan serta didasari oleh respon yang telah ditentukan dari pihak security
- Peringatan kemudian akan di observasi
- Dilakukan perangkuman akan laporan terkumpul
- Data yang terkumpul digunakan sebagai proses pendeteksian dalam durasi jangka panjang

#### 2.1.4 Komponen Intrusion Detection System

Deteksi intrusi memiliki 4 komponen penyusun yakni sensor, analysers, incident, storages, dan response units. Dimana komponen yang telah disebutkan tersebut saling terhubung dan saling berpengaruh antara satu sama lain.



Gambar 5 Komponen Penyusunan IDS

Untuk lebih lengkapnya berikut adalah tabel penjelasan dari komponen penyusun IDS

	Komponen	Keterangan
1	Sensor	Berfungsi sebagai mendapatkan network traffic serta memberikan output yang digunakan oleh komponen lainnya. Serta dapat memantau dan menghasilkan event berdasarkan traffic network. Terdiri atas network-based sensors dan host-based sensors.
2	Analysers	Menerima inputan dari sensors dan selanjutnya menganalisisnya dan mengeluarkan hasil berupa event yang diolah serta dianalisis. Teknik dari type tersebut signature-based analysers dan anomaly-based analysers

3	Incident Storages	Menyimpan keseluruhan alert dan mendukung proses analisis. Ketika dalam proses analisis, alert disimpan dalam komponen database dan hal itu dapat digunakan sebagai klarifikasi alertnya. Database hanya sebuah komponen IDS dengan tujuan menyimpan alert.
4	Response Units	Menggunakan output dari komponen analysers dan mengarahkannya agar dapat melakukan beberapa tindakan seperti menghentikan proses, reset koneksi, dan mengubah hak akses.

*Table 1 Komponen Penyusun IDS*

### **2.1.5 Rules Intrusion Detection System**

Rules IDS adalah aturan yang telah dirancang untuk menemukan perilaku yang mencurigakan atau membahayakan sistem komputer atau jaringan. Hukum-hukum ini didasarkan pada pola perilaku yang telah diamati atau diajarkan melalui pertemuan sebelumnya.

Set rule memiliki isi rule IDS yang disusun menjadi sets based yang berdasarkan terhadap berbagai properti (seperti aturan yang memiliki sebuah kondisi analisis traffic yang saling bergantung. Kategori dari Rules Intrusion Detection System ialah sebagai berikut

- Signature-Based Detection

Untuk mengidentifikasi ancaman yang dikenali, rules ini menggunakan pustaka signature. Signature mencakup perincian tentang tren atau ciri unik dari serangan yang diakui sebelumnya. Untuk mendeteksi bukti penyerangan yang sama, IDS akan mencocokkan aktivitas jaringan dengan tanda tangan yang sudah ada di database. IDS akan mengaktifkan peringatan jika ada indikasi yang muncul.

- Anomaly Based Detection

Klasifikasi ini mendeteksi aktivitas yang tidak biasa atau mencurigakan di jaringan. IDS akan memantau perilaku sistem dan mencari indikasi aktivitas yang tidak biasa. Sistem akan membuat profil yang terhubung, jaringan, dan aktivitas perangkat biasa. IDS akan mengeluarkan peringatan setiap kali ada perilaku yang tidak sesuai dengan profil tipikal.

Mekanisme Kerja Rules Intrusion Detection System tergantung pada jenis metode yang digunakan namun pada umumnya terdiri dari tahapan berikut yakni:

- Pemantauan  
Dilakukan pemantauan traffic jaringan dengan tujuan mencari tanda-tanda aktivitas yang mencurigakan
- Analisis  
Menentukan apakah aktivitas tersebut ialah serangan atau bukan. Dilakukan dengan perbandingan antara aktivitas yang dicurigai dengan database.
- Tindakan  
Memberikan tindakan sesuai konfigurasi yang tersedia, contoh tindakan berupa alert, block access, notifikasi terhadap admin.
- Pemulihan  
Membantu menyelesaikan permasalahan yang terjadi.

Dalam Tugas Akhir ini, mekanisme kerja dari Rules Intrusion Detection System sebagai berikut

Signature Base		Anomaly Base	
Perekaman Signature	Melakukan perekaman tanda-tanda karakteristik serangan yang dipahami	Pemantauan Anomali	Pemantauan anomaly sistem dan menelusuri kegiatan tidak normal.
Pencocokan Signature	IDS akan membandingkan serangan tersebut dengan database signature tersedia.	Pembuatan Profil Anomaly Normal	IDS akan membuat profil anomaly normal dari user, network, dan device connect.



Tindakan	Memberikan tindakan berdasarkan konfigurasi yang telah ditentukan	Pencocokan Anomali	Membandingkan serangan tersebut dengan profil anomaly serangan
		Tindakan	Tindakan berdasarkan konfigurasi yang telah ditentukan

*Table 2 Perbandingan Mekanisme Kerja IDS*

Contoh Rules Intrusion Detection System digunakan dalam proses pendeteksian serangan atau aktivitas yang mencurigakan pada network. Berikut adalah contoh dan penjelasannya.

- Rules Signature Based Detection

```
alert tcp any any -> any 80 (msg:"Potential HTTP Exploit Attempt";
content:"GET"; nocase; content:"/cmd.exe"; nocase; sid:100001;)
```

Rule ini akan memicu peringatan jika ada paket TCP yang dikirim ke port 80 dan mengandung string "GET" dan "/cmd.exe". Rule ini digunakan untuk mendeteksi serangan HTTP exploit yang mencoba mengambil alih sistem dengan memanfaatkan kerentanan pada aplikasi web.

- Rules Anomaly Based Detection

```
alert tcp any any -> any any (msg:"Large Amount of Outgoing Traffic";
threshold: type both, track by_dst, count 100, seconds 60; sid:100002;)
```

Rule ini akan memicu peringatan jika ada lebih dari 100 koneksi TCP yang dibuat dalam waktu 60 detik dari alamat tujuan yang sama. Rule ini digunakan untuk mendeteksi aktivitas botnet yang mencoba mengirimkan data keluar dari jaringan.

## 2.1.6 Metode Intrusion Detection System

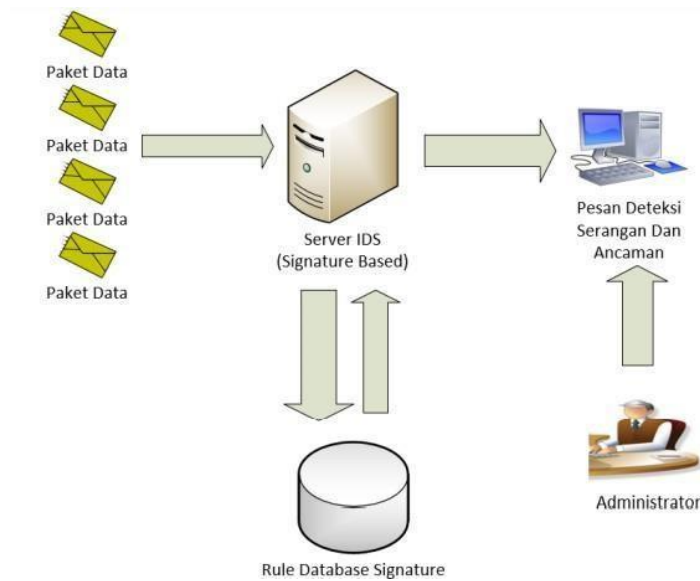
### 2.1.7.1 Signed Based

Metode signature ini adalah proses pendeteksian dengan melakukan pencocokan lalu lintas jaringan dengan basis data yang memiliki isi teknik-teknik penyusupan dan serangan yang merupakan tugas dari intruders. Metode ini juga dikenal dengan missue detection. Dalam komponen IDS, Signature based terdapat pada komponen analysers.

Cara kerja IDS berbasis Signature yakni dengan cara menyadap paket data dan selanjutnya membandingkan dengan basis data rules IDS yang memiliki pola-pola serangan. Terdapat dua perumpamaan dalam basis ini yakni Jika paket data memiliki pola yang sama dengan salah satu pola di basis data rules IDS, maka paket data tersebut ialah serangan. Jika paket data tidak ada hal yang sama dalam pola yang sama di basis data rules IDS, maka hal tersebut dianggap bukan serangan.

IDS akan mengawasi paket-paket tersebut dan membuat perbandingan paket-paket dengan basis data signature yang dimiliki IDS ataupun atribut dimiliki oleh percobaan serangan yang telah diketahui. Suatu proses yang terjadi keterlambatan [23].

Paket data ialah paket data yang ditangkap untuk melewati Ethernet card yang kemudian digunakan oleh metode signature yang digunakan untuk pemeriksaan. IDS akan membandingkan paket data dengan rule yang berada di dalam database IDS yang dipakai. Ketika terdapat persamaan pada database untuk memberikan pesan deteksi ketika terjadi dalam gangguan seperti dalam gambar.



Gambar 6 Mekanisme Signature Based

Metode berguna agar pola ancaman atau gangguan yang sudah dimiliki pola sebelumnya. Metode ini sangat baik digunakan dalam hal mendeteksi semua jenis serangan seperti port scanning, exploit dan DoS.

Terdapat 4 tahapan proses analisa yang terjadi dalam signature base detector yakni

1. Preprocessing

Melakukan proses pengumpulan data yang berkaitan dengan pola dari serangan dan meletakkannya pada skema klasifikasi. Dari skema tersebut maka dibangun sebuah model dan selanjutnya akan diinput dalam format.

2. Analysis

Perbandingan antara data dan format dengan pattern yang tersedia dengan tujuan untuk keperluan analysis engine pattern matching.

### 3. Response

Ketika ada yang sesuai dengan pola serangan maka analysis engine akan mengirimkan alert terhadap server

### 4. Refinement

Melakukan revolusi dari analisis pattern-matching yang diturunkan untuk upgrade signature, dikarenakan IDS hanya memberikan izin signature terakhir untuk diupdate.

Protokol Signature IDS menggunakan signature atau panduan yang telah ditetapkan untuk menemukan serangan atau perilaku mencurigakan. Setiap paket jaringan yang melewati IDS akan dibandingkan dengan katalog signature atau rules yang diterima, dan jika ada kecocokan, IDS akan mengeluarkan peringatan. Algoritma IDS Signature Snort, Suricata, dan Bro adalah beberapa contohnya.

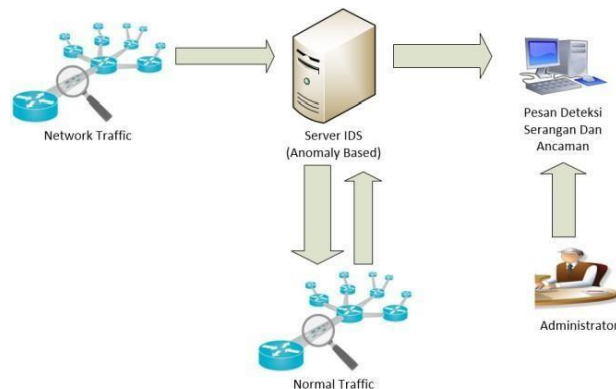
#### 2.1.7.2 Anomaly Based

Metode IDS yang satu ini adalah metode dengan cara deteksinya dengan memperhatikan lalu lintas yang dipantau kemudian membandingkan dengan lalu lintas normal. Detector Anomali kemudian menganalisis perilaku yang aneh dan hal itu terjadi dalam host atau network. Dengan melakukan asumsi bahwasanya serangan yang diterima berbeda dengan aktivitas normal sehingga dapat dideteksi dengan adanya sistem identifikasi perbedaan yang ada.

Pada metode ini digunakan untuk mengawasi trafik dalam jaringan serta melakukan perbandingan terhadap trafik yang tersedia. Metode ini melakukan identifikasi terhadap jaringan normal, jaringan itu dapat ditelusuri dalam penggunaan bandwidth yang terdapat di jaringan, protocol, port serta perangkat yang terhubung dalam jaringan tersebut. Metode ini mendeteksi adanya penyusupan dengan memantau

adanya keanehan pada sistem. Kemudian hal yang aneh tersebut dibandingkan dengan situasi yang normal [24]. Metode ini terdapat dalam komponen Analysers. Cara kerja dapat dilihat dalam gambar berikut ini

Metode ini bersifat dalam perbandingan jaringan normal dengan jaringan tidak normal. Jika dalam jaringan normal maka dapat diartikan maka gangguan tersebut akan menjadi ancaman bagi jaringan tersebut [25]



*Gambar 7 Mekanisme Anomaly Based*

Pada metode ini terdapat 3 tahapan proses analisis yang terdapat dalam anomaly detector

1. Preprocessing

Disusunnya profil-profil yang mendeskripsikan kebiasaan user yang normal, host atau koneksi jaringan. Profil tersebut dibangun dari data historis yang dikumpulkan dalam periode operasi normal. Data ini dikenal data training. Dan selanjutnya data tersebut dalam database dan selanjutnya dilakukan preprocessing. Contohnya pengelompokan berdasarkan source/destination IP dan port, flags & acks, menyimpan data (start time, duration, protocol), menghitung jumlah bytes in dan bytes out

2. Analysis

Terjadi proses pengumpulan data testing dan melakukan proses yang beragam

pada saat kegiatan yang diamati lari dari normal. Setelah data disimpan dalam bentuk tabel selanjutnya perhitungan statistik dengan data record yang tersedia. Contoh data tersebut adalah jumlah koneksi ke host yang sama selama T durasi terakhir, jumlah penolakan koneksi, jumlah services berbeda.

### 3. Response

Ketika terdapat kegiatan yang tidak normal maka menampilkan suatu grafik atau alarm yang berbeda dari kondisi normal.

Protokol Anomaly IDS mencoba mendeteksi serangan atau aktivitas mencurigakan dengan membandingkan perilaku jaringan dengan profil perilaku normal. IDS akan mempelajari dan memantau pola lalu lintas jaringan, kemudian membandingkannya dengan pola lalu lintas yang normal untuk mendeteksi aktivitas yang tidak biasa atau mencurigakan. Jika ada pola aktivitas yang tidak biasa, IDS akan menghasilkan peringatan. Contoh protokol IDS Anomaly adalah Statistical Anomaly-Based IDS dan Neural Network-Based IDS.

#### 2.1.7 Snort dan Suricata

Sebuah aplikasi bernama Snort menawarkan fitur yang dapat menghentikan serangan dan instruksi jaringan. Dalam jaringan berbasis TCP/IP, Snort dapat melakukan analisis lalu lintas dan mencatat paket secara real time. Martin Roesch menulis versi awal Snort, yang sekarang dikelola oleh Sourcefire, dimana Roesch adalah pendiri dan CTO (Chief of Technical Officer). Kombinasi dari sistem analisis protokol dan sistem deteksi intrusi, yang disebut Snort, menjadikannya alat yang sangat efektif untuk mengidentifikasi serangan pada host jaringan. Sementara fungsi Tcp Dump serupa, Snort berfokus pada mengendus paket aman. Pemeriksaan payload adalah fitur utama yang membedakan Snort dari TcpDump. Snort memeriksa seperangkat aturan muatan yang disediakan.

Snort memiliki beberapa keunggulan dalam menjalankan Intrusion Detection system diantaranya sebagai berikut

- Fleksibilitas

Terdapat banyak pemilihan dalam melakukan konfigurasi dan plugin yang dapat diterapkan berdasarkan keperluan jaringan

- Performa

Dalam melakukan identifikasi serangan jaringan dengan menggunakan teknik signature-base atau anomaly based detection

- Integrasi

Snort dapat digabungkan dengan skala kecil atau besar serta dapat dikonfigurasi dalam mengatasi masalah skalabilitas.

Suricata adalah sistem deteksi dan pencegahan intrusi (IDS/IPS) dengan pendekatan berbasis aturan yang memanfaatkan kumpulan aturan yang dibuat oleh pihak ketiga untuk melacak data jaringan yang diendus dan mengirim notifikasi saat aktivitas mencurigakan terjadi. Itu dibuat untuk berintegrasi dengan komponen keamanan jaringan saat ini, seperti mayoritas IDS.

Versi awal Suricata kompatibel dengan banyak tingkat lalu lintas gigabit dan mendukung konfigurasi pelacakan lalu lintas inline dan pasif pada platform Linux 2.6. Suricata bekerja sebagai mesin multithreaded.

Suricata adalah produk dari Open Information Security Foundation (OISF), yang menerima dana dari Space and Naval Warfare Systems Command (SPAWAR), Direktorat Sains dan Teknologi Departemen Keamanan Dalam Negeri, dan inisiatif Teknologi Keamanan Terbuka Dalam Negeri [26].

Suricata bekerja berdasarkan anomaly-based, setiap paket akan diperiksa berdasarkan rules yang ada pada Suricata dengan membandingkan kegiatan yang sedang dipantau dengan kegiatan yang dianggap normal untuk mendeteksi adanya penyimpangan [27].

Beberapa keunggulan Suricata dalam melakukan IDS antara lain:

- Multi-threading: Suricata mendukung pemrosesan multi-threading sehingga dapat mengatasi lalu lintas jaringan yang besar dengan cepat dan efisien.
- Pendeteksian yang akurat: Suricata memiliki pendeteksian yang akurat dan dapat mengidentifikasi serangan yang terjadi pada jaringan seperti scanning, exploit, brute force, dan lain-lain.
- Pengecekan tanda tangan yang efisien: Suricata dapat memeriksa tanda tangan yang kompleks dengan cepat dan efisien. Hal ini memungkinkan Suricata untuk memeriksa banyak serangan yang berbeda.
- Dukungan IPv6: Suricata mendukung IPv6 secara penuh dan dapat memantau lalu lintas jaringan IPv6.
- Sinkhole: Suricata dapat melakukan sinkhole pada IP dan domain yang terdeteksi melakukan serangan, sehingga dapat memblokir akses dari sumber yang merugikan.
- Dukungan protokol: Suricata mendukung banyak protokol seperti HTTP, FTP, DNS, SMTP, dan banyak lagi, sehingga dapat mendeteksi serangan yang terjadi pada protokol tersebut.
- Dukungan OS: Suricata dapat diinstal pada sistem operasi yang berbeda seperti Linux, Windows, dan macOS.



### 2.1.8 Kelebihan dan Kelemahan Intrusion Detection System

Pada tabel dibawah ini adalah kelebihan dan kelemahan dari Intrusion Detection System, berdasarkan dari pendapat J, Gondohanindijo[28] :

INTRUSION DETECTION SYSTEM	
Kelebihan	<ol style="list-style-type: none"><li>1. Tingkat akurasi keamanan yang baik IDS mempunyai kepakaran yang tinggi.</li><li>2. Mampu mendeteksi dan mencegah serangan.</li><li>3. Memiliki cakupan yang luas dalam mengenal proses attacking.</li><li>4. Tingkat forensik yang modern dan mengeluarkan reporting yang bagus</li><li>5. Sensor yang dapat dipercaya dalam pendeteksian dan pencegahan</li></ol>
Kelemahan	<ol style="list-style-type: none"><li>1. Paket data yang terdeteksi sebagai intrusion tidak sesuai dengan rule-rule dibuat</li><li>2. Variasi kode yang banyak dikarenakan modifikasi penyerangnya</li><li>3. False Positives (Adanya aktivitas yang memungkinkan itu ialah serangan)</li><li>4. False Negatives (IDS tidak memberikan alert, ada serangan tetapi tidak mengenal signature)</li><li>5. Data Overload</li></ol>

*Table 3 Kelebihan dan Kelemahan IDS*

### 2.1 Related Work

Pada bagian ini terdapat rangkaian penelitian dengan topik IDS, dengan tujuan untuk berguna dalam tinjauan dan hal utama dalam pengembangan IDS yang diantaranya :

No	Peneliti/Tahun	Judul	Masalah	Metode	Kesimpulan
1	Mercury Fluorida Fibrianda, ,Adhitya Bhawiyuga (2018)[29]	Analisis Perbandingan Akurasi Deteksi Serangan Pada Jaringan Komputer Dengan Metode Naïve Bayes Dan Support Vector	Suatu langkah dengan tujuan untuk mematikan tidak sepenuhnya server	Naïve Bayes, Support Vector Machine (SVM)	Mendapatkan suatu perbandingan dengan nilai yang akurat terhadap confusion, matrix,

		Machine(SVM)	komputer pada jaringan internet dengan begitu komputer tidak berfungsi dengan baik.		precision, recall dan fl score.
2	Nugroho, Dyakso Anindito Rochim, Adian Fatchur Widiyanto, Eko Didi k [2015][30]	Perancangan dan implementasi intrusion detection system di jaringan universitas Diponegoro	Kurangnya fitur IDS berbasis jaringan, terkhusus dalam tingkat serangan yang rendah serta korelasi yang buruk antara deteksi anomali	IPS Cisco	Ada penelitian sebelumnya penerapan IDS dilakukan pada jaringan dengan menggunakan sensor Cisco IPS yang mana dapat mendeteksi.
3.	Indranell, Mohuya, Satyajit[31]	A Comparative Study of Related Technologies of Intrusion Detection & Prevention System	Analisis terhadap jaringan yang menggunakan metode IDPS mengatasi serangan	IDPS	Analisis Jaringan dengan teknik metode NIPS dalam mendeteksi dari serangan DDoS.

*Table 4 Related Work*

## **BAB III**

### **ANALISIS DAN DESAIN**

Pada bab ini dideskripsikan tinjauan pustaka yang akan digunakan sebagai dasar teori dalam penyusunan Tugas Akhir.

#### **3.1 Analisis**

Pada bagian ini dibahas mengenai analisis masalah, analisis pemecahan masalah dan analisis kebutuhan sistem untuk menentukan solusi pemecahan terhadap permasalahan yang terjadi.

##### **3.1.1 Analisis Masalah**

Signature adalah pola atau indikasi unik yang digunakan untuk mengidentifikasi aktivitas jaringan yang mencurigakan atau serangan jaringan dalam catatan pelacakan aktivitas. Dalam hal ini tools untuk signature yakni menggunakan Snort.

Teknis Snort dalam melakukan monitoring log activity yang pertama yakni mengumpulkan data paket jaringan dari interface jaringan yang ditentukan untuk diproses. Setelah itu, data paket tersebut kemudian dianalisis oleh Snort berdasarkan aturan-aturan yang telah ditentukan. Aturan ini terdiri dari pola-pola khusus yang menunjukkan karakteristik dari suatu serangan atau aktivitas yang mencurigakan di dalam jaringan. Setiap kali terdeteksi suatu aktivitas yang mencurigakan, Snort akan mencatat log activity tersebut. Log activity ini berisi informasi tentang waktu terjadinya aktivitas, sumber dan tujuan aktivitas, jenis aktivitas yang terdeteksi, dan lain sebagainya.

Log activity ini kemudian dapat digunakan oleh administrator jaringan untuk melakukan analisis lebih lanjut terhadap aktivitas yang mencurigakan tersebut. Dengan begitu, administrator dapat mengambil tindakan yang tepat untuk melindungi

jaringan dari serangan atau aktivitas yang tidak diinginkan.

Namun, seperti pendekatan signature-based pada umumnya, penggunaan Snort memiliki beberapa kelemahan. Salah satunya adalah keterbatasan dalam mendeteksi serangan yang belum diketahui atau varian serangan yang dimodifikasi secara signifikan seperti update signature dan signature bypass. Snort hanya efektif dalam mendeteksi serangan yang memiliki tanda tangan yang ada dalam database dan tidak dapat secara otomatis mengenali serangan baru yang belum diketahui.

Sementara, Anomaly detection sendiri itu adalah teknik untuk mendeteksi kejadian atau data yang tidak biasa atau tidak sesuai dengan pola yang diharapkan dalam suatu sistem atau lingkungan tertentu. Sedangkan signature detection adalah teknik untuk mendeteksi kejadian yang telah dikenal atau ditandai sebagai serangan atau ancaman keamanan dalam sistem komputer. Dalam hal ini tools untuk signature yanti menggunakan suricata.

Teknik Suricata adalah sebuah Intrusion Detection System (IDS) yang menggunakan pendekatan deteksi anomali untuk mengidentifikasi serangan dalam lalu lintas jaringan. Dalam konteks Suricata, pendekatan deteksi anomali melibatkan pemantauan dan analisis terus-menerus terhadap perilaku jaringan untuk mengidentifikasi aktivitas yang tidak biasa atau mencurigakan yang mungkin merupakan indikasi adanya serangan atau pelanggaran keamanan.

Namun, seperti pendekatan anomaly-based pada umumnya, penggunaan suricata memiliki beberapa kelemahan. Pendeteksian signature akan bergantung pada aturan atau pola yang telah ditentukan sebelumnya. Jika serangan menggunakan metode yang belum dikenal, Suricata mungkin tidak dapat mendeteksinya secara akurat. Ini berarti serangan baru atau serangan yang diubah sedikit mungkin terlewatkan oleh pendekatan ini sebelum ada pembaruan aturan atau pola baru yang

tersedia. Selain itu anomaly juga cenderung menghasilkan tingkat kesalahan positif yang lebih tinggi, dimana ini mengacu pada kemungkinan lebih banyak alarm palsu atau kejadian yang salah diidentifikasi sebagai perilaku aneh atau serangan. Dalam metode anomaly, sistem berusaha mengenali perilaku jaringan yang tidak biasa atau tidak konvensional. Namun, ini juga bisa mengakibatkan beberapa kesalahan dalam mengklasifikasikan aktivitas yang sebenarnya sah sebagai anomali atau serangan.

### **3.1.2 Analisis Pemecahan Masalah**

Berdasarkan permasalahan yang telah dijelaskan dalam sub bab sebelumnya, pada hal tersebut dijelaskan bahwa metode Signature based dengan tools snort dan metode Anomaly based dengan tools suricata itu memiliki tantangan dan batasannya masing-masing dalam melakukan pendeteksian serangan. Maka solusi tepat yang bisa dilakukan dalam permasalahan tersebut yakni melakukan pensentralisasian antara anomaly based dan signature based, dengan tujuan bisa saling melengkapi antara kedua metode tersebut.

Dalam suatu kasus, signature detection sendiri mungkin tidak dapat mendeteksi ancaman keamanan yang baru atau belum dikenal. Dalam hal ini, anomaly detection dapat membantu dengan mendeteksi aktivitas yang tidak biasa atau tidak sesuai dengan pola normal dalam suatu sistem. Dengan demikian, anomaly detection dapat membantu meningkatkan keamanan sistem secara keseluruhan dengan mendeteksi ancaman keamanan yang mungkin tidak dapat dideteksi oleh signature detection. Secara keseluruhan, peran anomaly detection dalam membantu signature detection adalah untuk meningkatkan akurasi dan efektivitas signature detection dengan mendeteksi aktivitas atau perilaku yang tidak biasa atau tidak sesuai dengan pola normal dalam suatu sistem atau jaringan.

Dalam kasus lain, seperti yang kita ketahui bahwa anomaly cenderung

menghasilkan tingkat kesalahan positif yang tinggi. Namun dengan adanya metode signature tersebut dapat membantu anomaly untuk memverifikasi adanya serangan-serangan palsu sehingga dapat mengurangi tingkat kesalahan positif secara keseluruhan dan meningkatkan ke efisiensi sistem deteksi. Selain itu anomaly juga mengalami kesulitan dalam mendeteksi serangan menggunakan teknik evasi yang kompleks. Namun dengan teknik signature, kita dapat mendeteksi serangan menjadi lebih baik lagi dan kelemahan terhadap evasi dapat dikurangi.

Efektivitas, akurasi, menurunkan tingkat kesalahan positif, serta meningkatkan pendeteksian yang lebih kompleks dalam sistem deteksi keamanan merupakan tujuan dilakukannya sentralisasi antara pendekatan metode anomaly dan signature. Ketika melakukan pensentralisasian kedua metode ini, memungkinkan sistem deteksi keamanan dapat memberikan alarm atau notifikasi secara realtime kepada administrator sehingga memungkinkan tindakan respons yang cepat untuk mengatasi serangan.

### **3.1.3 Analisis Kebutuhan Sistem**

Pada saat melakukan testing dalam penyusunan tugas akhir ini, maka diperlukan beberapa perangkat keras dan perangkat lunak, berikut adalah kebutuhan perangkat keras dan perangkat lunak yang dipergunakan dalam pengujian.

### 3.1.3.1 Analisis Kebutuhan Perangkat Keras

Perangkat keras adalah alat yang berada dalam sistem komputer yang secara fisik dapat terlihat dan dapat disentuh semua alatnya Perangkat keras (hardware) yang dibutuhkan yaitu:

Perangkat	Keterangan
Laptop	Digunakan dalam pembuatan dokumen tugas akhir.
	Instalasi Sistem Operasi
	Pengiriman Serangan
	Centralization Data of Anomaly Detection dan Signature Detection

Table 5 Kebutuhan Hardware

### 3.1.3.2 Analisis Kebutuhan Perangkat Lunak

Software yang digunakan dalam penelitian ini diantaranya sebagai berikut:

No	Software	Keterangan
1	Virtual Box	Software yang diperlukan dalam membuat OS Kali Linux dan OS Ubuntu Server
2	Ubuntu	Sistem operasi yang berperan sebagai server yang digunakan sebagai centralisasi anomaly dan signature base yang kemudian menjadi target sistem yang akan diserang
3	Snort	Sebagai IDS yang akan berperan dalam Signature Based
4	Suricata	Sebagai IDS yang akan berperan dalam Anomaly Based
5	PHP	Salah satu bahasa pemrograman berbasis aplikasi web yang didesain dan biasanya digunakan pada sisi server
6	MySQL Server	Sebagai Database yang memuat data serangan
7	Apache Web Server	Server web yang dapat dijalankan untuk memfungsikan dan melayani aplikasi web
8	DVWA (Damn Vulnerable Web Application)	Aplikasi web yang dibuat khusus untuk mempelajari dan memahami bagaimana kerentanan suatu web dapat dieksploitasi

9	NMAP	Tools yang dilakukan untuk melakukan penyerangan dalam Port Scanning
10	HPING3	Tools yang digunakan dalam proses penyerangan dalam Denial Of Service

*Table 6 Kebutuhan Software*

### 3.1.4 Analisis Sistem

Centralization data dari signature ataupun anomaly adalah suatu teknik pengumpulan dan penyimpanan data dari dua pendekatan yang berbeda yakni berbasis signature dan berbasis anomaly, ke dalam sebuah repository atau sebuah database.

Dalam hal signature base, adanya sebuah pendekatan pendeteksian intrusi yang menggunakan aturan atau rule yang telah dikenal dalam proses identifikasi serangan yang telah diketahui sebelumnya. Aturan dikonfigurasi berdasarkan pola serangan yang telah teridentifikasi.

Dalam hal anomaly base, adanya pendekatan intrusi yang berdasarkan pemodelan perilaku normal dari sistem ataupun jaringan. Jika terdapat perbedaan yang signifikan dari pola perilaku normal akan menampilkan suatu kemungkinan serangan.

Dengan dilakukannya centralisasi data, hasil dari pendeteksian kedua tools disimpan dalam satu database. Tujuan dilakukannya memudahkan dalam proses pemantauan yang lebih efektif dalam keamanan jaringan. Data yang dikumpulkan data diakses serta dievaluasi secara baik dan dapat diidentifikasi ancaman yang lebih cepat respons terhadap serangan keamanan.

Analisis Centralisasi A-IDS dan S-IDS	
Faktor	Deskripsi
Sensor Deteksi	Sensor deteksi intrusi adalah komponen yang bertanggung jawab untuk memantau dan mencatat aktivitas jaringan. Dalam sistem ini, sensor dapat berupa sensor signature (Rules Snort) dan sensor berbasis anomali (Rules Suricata).



IDS	Mempertimbangkan kinerja dan efektivitas IDS dalam mendeteksi ancaman keamanan dengan akurasi tinggi.
Centralisasi Data	Memperhatikan pengiriman data deteksi dari sensor IDS ke database, pengolahan data untuk memisahkan data signature dan anomaly, serta penyimpanan data dalam format yang efisien dan mudah diakses. Evaluasi proses ini akan memastikan bahwa data terkumpul dengan benar dan dapat diakses dengan efisien.
Analisis & Respons	Mengamati data yang terkumpul dengan cepat dan efektif untuk mengidentifikasi serangan dan mengambil tindakan yang tepat.
Keamanan & Privasi	Mempertimbangkan aspek keamanan dan privasi dalam proses centralisasi data. Ini melibatkan perlindungan data yang sensitif, pengamanan akses ke sistem, dan pemantauan aktivitas yang mencurigakan.

*Table 7 Analisis Sistem IDS*

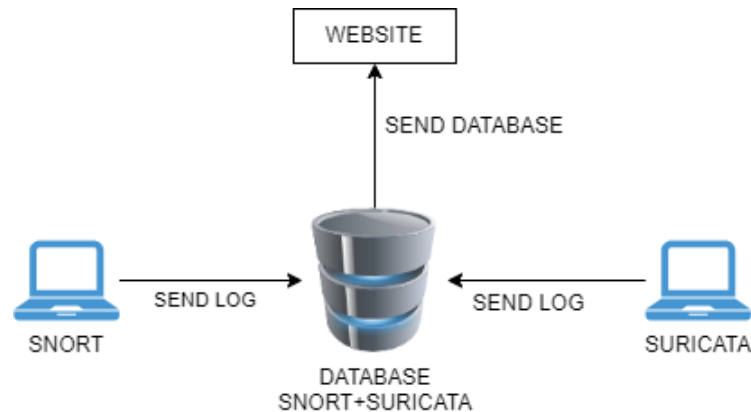
### 3.2 Perancangan Pembangunan Sistem

Pada bagian ini perancangan pembangunan sistem akan dipaparkan pada bagian ini yang akan dibangun yakni dalam “Centralization Data of Signature Based and Anomaly Based on Intrusion Detection System”. Perancangan ini disesuaikan dengan yang dijabarkan pada sub-bab sebelumnya

Perancangan sistem merupakan tahap awal dalam proses melakukan pengujian Anomaly dengan Signature terhadap serangan yang melibatkan beberapa software dan hardware. Perancangan digunakan sebagai dasar dalam mengaplikasikan sistem pada Tugas Akhir. Serta proses perancangan ini, segala kemungkinan yang dapat memperlambat proses pengerjaan Tugas Akhir dapat diminimalkan. Jadi perancangan ini dapat mempermudah dalam merealisasikan pembangunan Tugas Akhir berdasarkan spesifikasi hardware dan software yang diperlukan.

### 3.2.1 Perancangan Umum Sistem

Pada perancangan umum sistem dalam proses pembuatan sentralisasi Data Anomaly Detection dan Signature ini mencakup perancangan dari keterhubungan setiap software yang digunakan. Sistem sentralisasi Data Anomaly dan Signature adalah sebuah pengumpulan data antara deteksi anomali (anomaly detection) dan deteksi signature (signature detection) untuk mengidentifikasi serangan pada jaringan atau sistem komputer dan mempermudah monitoring serangan.



Gambar 8 Perancangan Umum Sistem

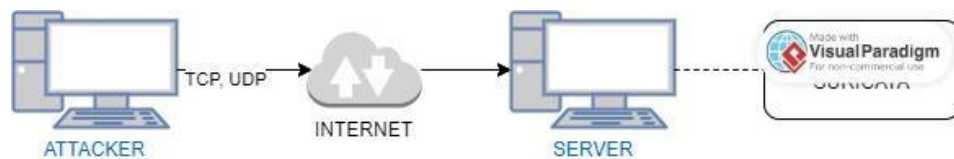
Dalam pengambilan tugas akhir ini, kami membangun IDS berbasis signature dengan menggunakan tools yakni Snort, sedangkan IDS berbasis Anomaly menggunakan tools suricata. Setiap hasil pendeteksian oleh Snort dan Suricata akan dikirimkan dan disimpan dalam database, yang kemudian database akan bertujuan untuk ditampilkan dalam aplikasi web.

Pada gambar diatas, bahwasanya antara Snort dan Suricata harus dapat melakukan sentralisasi. Serangan yang dikirimkan oleh attacker terhadap target akan dideteksi oleh Snort dan Suricata secara bersamaan. Selanjutnya data serangan akan dikelompokkan berdasarkan tingkat severity. Ketika snort dan Suricata berhasil melakukan pendeteksian serangan yang severity high maka akan mengaktifkan alert memberitahukan administrator sistem tentang adanya serangan dan mencatat semua

kejadian serangan dalam log sistem untuk diolah menjadi database dan dikirimkan kepada website. Namun ketika snort dan Suricata mendapatkan hasil detect serangan tetapi berbeda tingkat severity yang diterima maka hal selanjutnya dilakukan yakni melakukan severity mapping sehingga selanjutnya tingkat severity yang diterima sama. Dimana severity mapping adalah melakukan pemetaan dalam snort dan Suricata. Hasil log data yang telah tersimpan di kirimkan terhadap database. Tujuan dikirimkan dalam database maka selanjutnya akan dialihkan menjadi database dalam website

### 3.2.2 Perancangan Arsitektur Sistem

Perancangan Arsitektur Sistem dalam mendirikan sistem ini dari perancangan keseluruhan sistem dibangun dalam lingkungan virtualisasi VM Ware / VirtualBox. Pada subbab ini akan mendeskripsikan bagaimana perancangan arsitektur sistem “Sentralisasi Data Intrusion Detection System (Anomaly & Signature)”. Dapat diperhatikan pada gambar dibawah adalah perancangan arsitektur sistem.



Gambar 9 Perancangan Arsitektur Sistem

Berdasarkan gambar diatas, skema dari perancangan desain sistem tersebut adalah sebagai berikut

1. Dilakukannya penyerangan terhadap aplikasi (target) dengan TCP/UDP/ICMP request oleh Attacker
2. Server menerima paket data yang dikirimkan
3. Snort dan Suricata yang telah diinstall pada server melakukan scan kepada

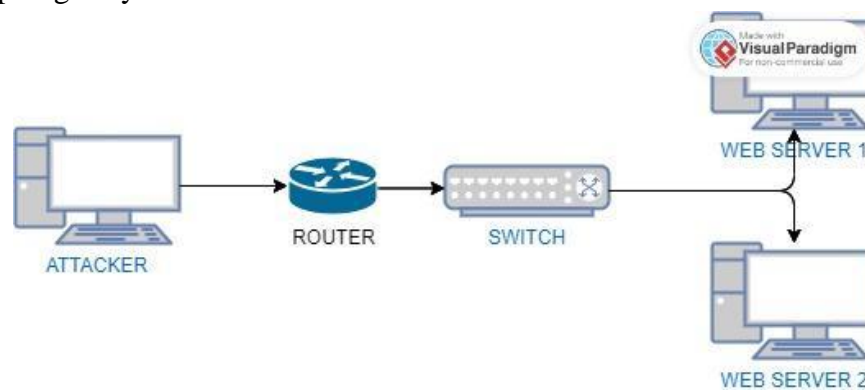
paket yang diterima dengan cara perbandingan paket dengan rules yang ditetapkan oleh snort dan Suricata.

Data yang diterima oleh snort dan Suricata dikirim ke database.

### 3.2.3 Topologi

Pada sub bab ini dipaparkan bagaimana topology sistem keamanan yang didirikan yakni sebagai berikut

#### 3.2.3.1 Topologi Physical



Gambar 10 Topologi Physical

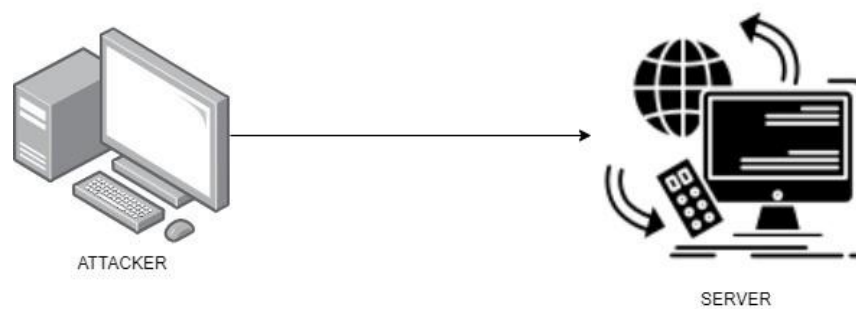
Topologi ini mendeskripsikan peralatan yang diperlukan dalam pengujian. Berikut adalah gambar topology physical yang dipergunakan ketika dalam proses pengujian yakni

Pada gambar diatas ialah dimana kondisi attacker terdapat di luar jaringan dan target sistem berada dalam jaringan yang dipantau dalam administrator. Laptop merupakan hardware yang dipergunakan oleh attacker dan target sistem. Sedangkan fungsi yang berguna untuk menghubungkan jaringan dan meneruskan paket data yang masuk yakni tugas dari router. Lalu yang membagi jaringan ke beberapa target sistem ialah peran dari switch.

Proses pengerjaan yang dilakukan dalam tugas akhir ini menggunakan virtual box tanpa menggunakan router dan switch sebagai komunikasi dikarenakan menjalankan peran virtualisasi. Sistem operasi ubuntu diinstal dalam melakukan serangan ke target sistem serta menjadi target sistem dan web server.

#### 3.2.3.2 Topologi Logical

Topologi ini mendeskripsikan skema komunikasi dalam proses sistem yang diperlukan dalam pengujian. Berikut adalah gambar topology logical yang dipergunakan dalam pengujian yakni:



*Gambar 11 Topologi Logical*

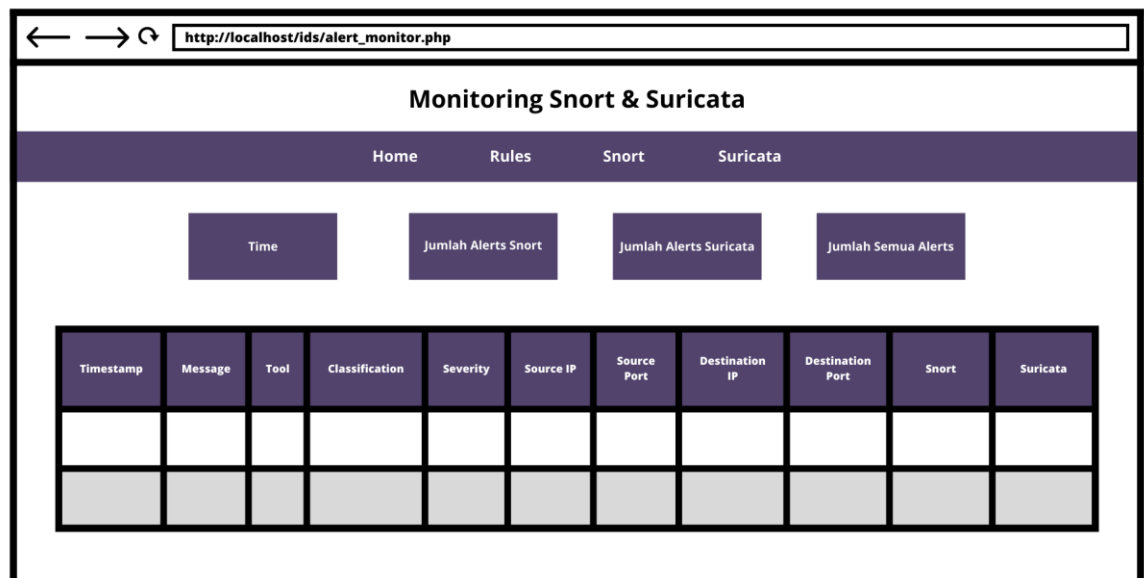
Dari gambar yang dipaparkan di atas, attacker melakukan penyerangan terhadap target server ubuntu dengan menggunakan sistem operasi ubuntu dengan alamat ip yang berbeda dalam mesin virtual. Saat dilakukan proses penyerangan dengan request yang tinggi melalui internet, maka selanjutnya serangan yang masuk disesuaikan dengan rules yang terdapat dalam snort dan Suricata. Snort dan Suricata melakukan pendeteksian threat dan memantau kecepatan/performansi dalam setiap jaringan dan selanjutnya mencari pola yang tidak terdapat dalam rules yang ditentukan. Selanjutnya aplikasi web dibangun dalam web server yang memberikan hasil penampilan berupa hasil pemantauan threat yang dideteksi di jaringan oleh snort dan suricata.

### 3.2.4 Perancangan Web System

Website yang diperlukan dalam menyelesaikan tugas akhir ini dapat dilihat dari desain website berikut. Sistem web ini hanya menampilkan data yang diterima oleh oleh snort dan suricata dalam melakukan monitoring pendeteksian threat.

#### 3.2.4.1 Desain Tampilan Monitor Snort dan Suricata

Desain dari tampilan monitor snort dan suricata berisi tampilan utama yang menampilkan jumlah keseluruhan alert yang masuk, jumlah rules, mencatat ip address dan waktu secara realtime. Pada halaman monitoring ini juga ditampilkan mengenai data serangan yang masuk melalui traffic jaringan dan mencatat klasifikasi tingkat serangan, signature serangan, ip address, port dan timestamp pada database. Pada keterangan di bagian atas, jumlah alert akan menampilkan jumlah keseluruhan alert yang masuk yang terdeteksi oleh Snort, Time akan menampilkan realtime.



Gambar 12 Web Design Monitoring Suricata & Snort

Kolom classification akan berisi angka yang akan diklasifikasikan tingkat severity/serangan berdasarkan hasil deteksi. Kolom signature akan berisi mengenai detail serangan yang masuk (misalnya serangan Sql injection) dan dicocokkan ke signature Snort yang sudah didefinisikan. Apabila terdeteksi threat/serangan maka ip source penyerang akan dicatat langsung ke dalam database pada kolom ip source dan port source. Sedangkan ip address dan target yang diserang akan berisi ip address dari target.

### 3.2.4.2 Desain Tampilan Rules Snort dan Suricata

Desain dari tampilan rules snort dan suricata berisi tampilan yang berisikan dokumentasi bagaimana proses pendeteksian sebuah serangan dengan rules yang telah diklasifikasikan dengan tujuan agar memudahkan admin dalam melihat proses pendeteksian.

Action	Protocol	Source IP	Source Port	Direction	Destination IP	Destination Port	Message	Classtype	ID Rules

Gambar 13 Design Tampilan Rules Suricata

Dalam desain tampilan rules suricata mendokumentasikan hasil pendeteksian dan berisikan informasi rules yang dipakai dalam proses pendeteksian seperti action,

protocol, source ip, source port, direction, destination ip, message, classtype serta id rules.

Rules Snort									
Home	Rules	Snort	Suricata						
Time		Jumlah Rules							
Action	Protocol	Source IP	Source Port	Direction	Destination IP	Destination Port	Message	Classtype	ID Rules

Gambar 14 Design Tampilan Rules Snort

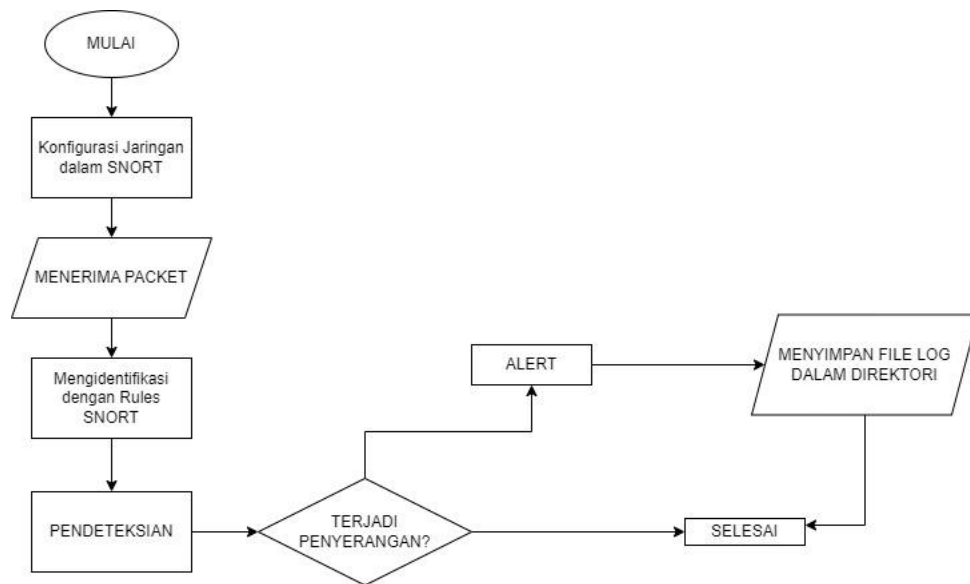
Dalam desain tampilan rules snort mendokumentasikan hasil pendeteksian dan berisikan informasi rules yang dipakai dalam proses pendeteksian seperti action, protocol, source ip, source port, direction, destination ip, message, classtype serta id rules.

### 3.3 Flowchart Penelitian

Pada bagian ini akan dipaparkan mengenai diagram alur yang diperlukan dalam penelitian ini. Diagram alur ini berfungsi untuk menjelaskan bagaimana proses teknis sistem yang akan dibuat. Dalam flowchart ini akan dijelaskan mekanisme teknik sistem sentralisasi snort suricata dalam melakukan pengumpulan logs, mendeteksi serangan.



### 3.3.1 Flowchart Signature Base



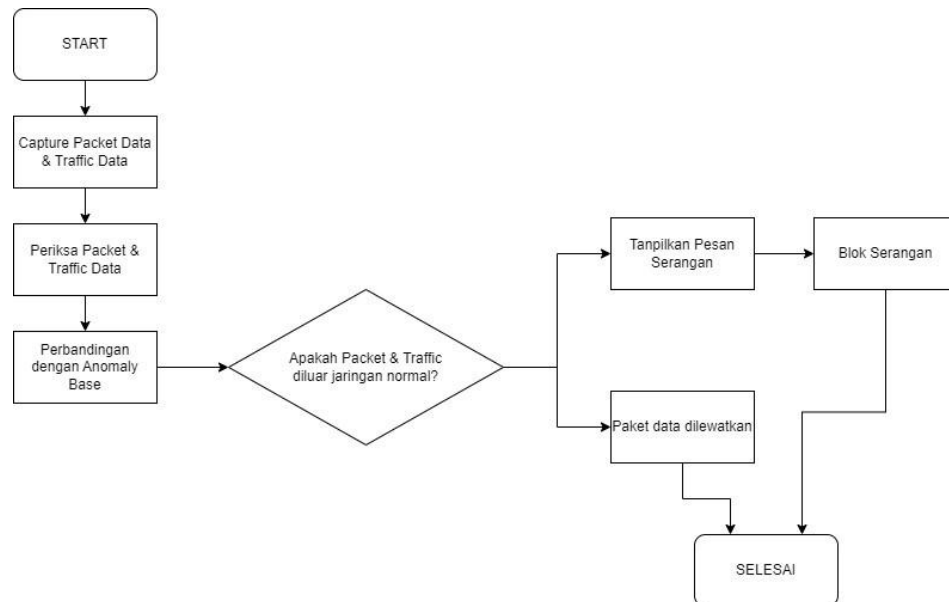
Gambar 15 Flowchart Signature Base

Pada gambar diatas adalah teknis kerja SNORT dalam signature base, dimana ketika attacker mengirimkan attack ke server/host dan langkah selanjutnya sistem akan mengkonfigurasi snort sebagai IDS dan akan melakukan capture packet dan langkah selanjutnya mengkonfigurasi terhadap rules yang telah ditetapkan, setelah itu sistem melakukan pendeteksian. Ketika sistem mendeteksi bahwa tersebut adalah attacker maka akan muncul alert dan alert itu akan disimpan dalam log, dengan disimpannya itu maka administrator mendapatkan peringatan adanya penyusupan. Dan begitulah teknis kerja dari snort

### 3.3.2 Flowchart Anomaly Base

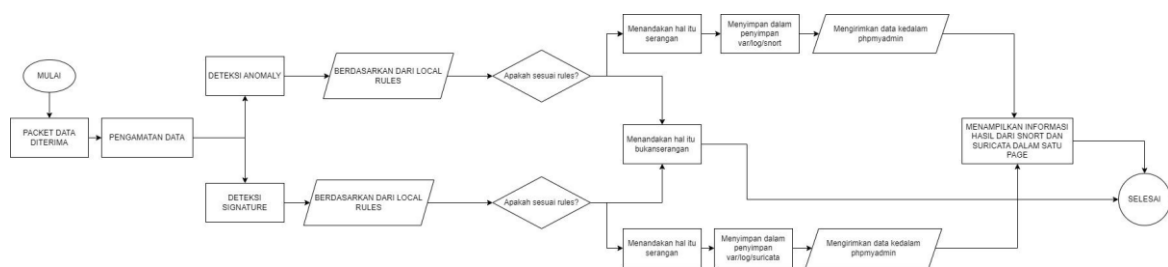
Pada gambar dibawah ini adalah teknis kerja SNORT dalam signature base, dimana ketika attacker mengirimkan attack ke server/host dan langkah selanjutnya sistem akan mengkonfigurasi snort sebagai IDS dan akan melakukan capture packet dan

langkah selanjutnya mengkonfigurasi terhadap rules yang telah ditetapkan, setelah itu sistem melakukan pendeteksian. Ketika sistem mendeteksi bahwa tersebut adalah attacker maka akan muncul alert dan alert itu akan disimpan dalam log, dengan disimpennya itu maka administrator mendapatkan peringatan adanya penyusupan. Dan begitulah teknis kerja dari snort.



Gambar 16 Flowchart Anomaly Base

### 3.3.3 Flowchart Centralization Signature dan Anomaly



Gambar 17 Flowchart Sentralisasi Data

Pada gambar diatas adalah teknis kerja sentralisasi antara anomaly dan signature base, dimana ketika attacker mengirimkan attack ke server/host dan langkah selanjutnya

sistem akan menganalisis data yang diterima dan akan melakukan capture packet dan langkah selanjutnya mengkonfigurasi terhadap rules yang telah ditetapkan baik secara anomaly ataupun signature, setelah itu sistem melakukan pendeteksian. Ketika sistem mendeteksi bahwa tersebut adalah attacker maka akan muncul alert dan alert itu akan disimpan dalam log, dengan disimpannya itu maka administrator mendapatkan peringatan adanya penyusupan. Selanjutnya log data antara signature dan anomali di kolaborasi menjadi sebuah database system. Setelah terbentuk database system maka selanjutnya akan dikirim dalam database web server.

### **3.4 Eksperimen Pengujian**

Pengujian ini dibutuhkan 2 buah laptop yakni 1 laptop untuk target dan IDS, 1 laptop untuk attacker. IDS Signature dan Anomali terdapat dalam satu laptop yang sama. Kedua laptop tersebut berada dalam satu jaringan yang sama. Dalam pengujian ini hanya terdapat 2 komputer yang real yakni 1 komputer attacker lalu 1 komputer untuk IDS dan target berupa virtual machine.

Pengujian yang dilakukan adalah dengan melakukan percobaan minimal 20 kali. Pengambilan sampel data dalam tugas akhir ini berupa pengujian serangan yang terdeteksi diambil dalam jumlah batas minimal sebanyak 30 kali dikarenakan jika hendak melakukan pengujian yang baik dan akurat itu ketika melakukan jumlah percobaan lebih banyak. Fungsi dilaksanakan sentralisasi antara anomaly dan signature ialah yaitu meningkatkan efektivitas sistem deteksi intrusi dalam mengenali serangan dan mengurangi tingkat kebocoran data yang mungkin terjadi serta juga dapat meningkatkan kecepatan deteksi, mengurangi jumlah false positif dan false negatif yang terjadi.

Dalam mengamati keadaan snort dan suricata dalam beberapa penyerangan yang dilaksanakan maka proses pendeteksian serangan ditinjau dari performa dengan adanya Confusion Matriks yakni

1. True Positive, Terdapat aktivitas serangan yang diprediksi dan benar itu adalah serangan. Contohnya berhasil mendeteksi serangan SQL Injection.
2. True Negative, Tidak terdapat aktivitas serangan dan benar itu tidak serangan. Contohnya tidak mendeteksi adanya serangan padahal tidak ada serangan.

False Positive, Terdapat adanya serangan yang diprediksi namun hal tersebut bukan serangan. Contohnya mendeteksi traffic network sebagai serangan tetapi hanya data normal yang tidak membahayakan

3. False Negative, Terdapat tidak adanya serangan namun ternyata itu adalah serangan. Contohnya serangan hacker berhasil melewati sistem deteksi tanpa terdeteksi.

Dalam sentralisasi Data Anomaly dan Signature, performa dari deteksi intrusi ditinjau berdasarkan banyak false positive, false negative, true negative, true positive, true negative. Semakin sedikit false positif dan false negatif dan true positive dan true negative maka performa sistem deteksi semakin baik.

Eksperimen Pengujian dilakukan dengan 3 macam studi kasus yakni

### **3.4.1 Eksperimen Pengujian Denial Of Service**

DOS adalah jenis serangan yang membuatnya dengan cara menghabiskan resource yang dimiliki pada server yang dituju. Dalam tugas akhir ini DoS yang dipergunakan adalah jenis TCP SYN Flood Attack, yakni type serangan yang dimana mengirim paket SYN secara terus-menerus hingga target hanya melayani request SYN dari attacker serta dapat mengakibatkan layanan yang terjadi dalam target

berhenti. Eksperimen pengujian penyerangan dengan Denial Of Service dapat dilihat sebagai berikut

Kode Pengujian	SK01
Kondisi Awal	<ul style="list-style-type: none"> <li>• Snort dan Suricata telah diinstall pada ubuntu server dan siap dipakai dalam host server</li> <li>• Menginstall Hping3 dalam Attacker</li> <li>• Contoh Pembuatan Rules Denial Of Service di Suricata dan Snort</li> </ul> <pre> alert tcp any any -&gt; \$HOME_NET 80 (msg: "DoS"; classtype: attempted-dos; sid:1000004; flags:S; flow:stateless; threshold:type_both, track by_dst, count 70; seconds 10;)  alert tcp any any -&gt; \$HOME_NET 80 (msg: "DoS"; classtype: attempted-dos; sid:1000005; flags:S; flow:to_server; threshold:type threshold, track by_src, count 70; seconds 10;) </pre> <p>Kedua aturan ini bertujuan untuk mendeteksi aktivitas yang mencurigakan terkait serangan Denial of Service (DoS). Aturan-aturan ini memeriksa lalu lintas TCP menuju port 80 di jaringan internal dan melacak jumlah paket yang memenuhi kriteria tertentu dalam periode waktu yang ditentukan. Jika ambang batas tercapai, pesan "DoS" akan ditampilkan, dan aktivitas akan diklasifikasikan sebagai upaya serangan DoS. ID aturan yang digunakan adalah 1000004 dan 1000005.</p>
Alur Skenario	<ul style="list-style-type: none"> <li>• Antara attacker dan target saling terhubung dengan cara saling ping melalui ip address antara target dan attacker</li> </ul>

	<ul style="list-style-type: none"> <li>• Pada attacker menjalankan command sebagai berikut Perintah ini untuk mengirimkan paket SYN ke target “sudo hping3 -S -flood -V -p 80 &lt;target_IP&gt;”.</li> <li>• Pada snort dan suricata menjalankan command Snort  sudo snort -q -l /opt/lampp/htdocs/ids -i enp0s3 -A fast -c /etc/snort/snort.conf  Suricata  sudo systemctl start suricata  Melakukan pemeriksaan log dari snort dan suricata dengan tujuan melihat apakah serangan DoS terdeteksi. Jika sudah sesuai rules yang telah dibuat.</li> <li>• Database hasil log penggabungan antara snort dan suricata dikirimkan terhadap website. Dengan tujuan untuk menampilkan data hasil serangan.</li> </ul>
Hasil Akhir	<p>Pada hal ini diharapkan dapat menguji kemampuan Pengiriman data Snort dan Suricata dalam mendeteksi serangan Denial of Service serta memperoleh informasi untuk melakukan peningkatan performa sistem deteksi serangan tersebut.</p>

*Table 8 Eksperimen Pengujian Denial Of Service*

### **3.4.2 Eksperimen Pengujian Port Scanning**

Sebuah serangan yang memiliki tujuan yakni berhasil mengumpulkan informasi dengan jumlah yang besar pada sebuah jaringan, seperti ipscan, portscan, penggalan

informasi terkait sistem operasi dipakai. Port scanning menggunakan tools yakni Nmap. Serangan dilakukan oleh komputer attacker dengan membuat suatu percobaan serangan port scanning. Dengan fungsi melihat port yang terbuka serta melihat sistem operasi yang digunakan oleh komputer target. Eksperimen pengujian penyerangan dengan Port Scanning dapat dilihat sebagai berikut

Kode Pengujian	SK02
Kondisi Awal	<ul style="list-style-type: none"> <li>● Snort dan Suricata telah diinstall pada ubuntu server dan siap dipakai dalam host server</li> <li>● NMAP telah berhasil diinstall dan dapat dipergunakan</li> <li>● Contoh Pembuatan Rules Port Scanning di Suricata dan Snort alert tcp \$EXTERNAL_NET any -&gt; \$HOME_NET 161 (msg: "Port Scanning"; classtype: attempted-recon; sid:1000006; flow:stateless;)</li> </ul> <p>Penjelasan Rules:</p> <p>alert tcp: Mendefinisikan aturan ini hanya berlaku untuk protokol TCP.</p> <p>\$EXTERNAL_NET any -&gt; \$HOME_NET 161: Menunjukkan arah lalu lintas yang akan diperiksa. \$EXTERNAL_NET merujuk pada jaringan eksternal, sedangkan \$HOME_NET merujuk pada jaringan internal. Aturan ini mendeteksi lalu lintas yang menuju ke port 161 di jaringan internal.</p> <p>(msg: "Port Scanning"; classtype: attempted-recon; sid:1000006; flow:stateless;): Bagian ini memberikan informasi tambahan tentang aturan ini:</p>

	<p>msg: "Port Scanning": Pesan yang akan ditampilkan jika aturan ini dipicu. Pesan ini akan memberitahu pengelola IDS bahwa terdeteksi aktivitas pemindaian port.</p> <p>classtype: attempted-recon: Mengklasifikasikan aktivitas yang dicurigai sebagai "attempted-recon" (upaya pengintaian/rekognisi).</p> <p>sid:1000006 dan sid 100007: ID aturan unik yang digunakan untuk mengidentifikasi aturan ini.</p> <p>flow:stateless: Menunjukkan bahwa aturan ini tidak bergantung pada status koneksi dan diperiksa secara stateless (tidak melacak koneksi).</p>
Alur Skenario	<ul style="list-style-type: none"> <li>• Antara attacker dan target saling terhubung dengan cara saling ping melalui ip address antara target dan attacker</li> <li>• Pada attacker menjalankan command sebagai berikut  “nmap -p- [IP TARGET]”  Perintah ini untuk melakukan deteksi dan menyampaikan kepada attacker port yang sedang terbuka yang ada di komputer korban.</li> <li>• Pada snort dan suricata menjalankan  command Snort  sudo snort -q -l /opt/lampp/htdocs/ids -i enp0s3 -A fast -c /etc/snort/snort.conf  Suricata</li> </ul>



	<p>sudo systemctl start suricata</p> <ul style="list-style-type: none"> <li>● Melakukan pemeriksaan log dari snort dan suricata dengan tujuan melihat apakah serangan Port Scanning terdeteksi. Jika sudah sesuai rules yang telah dibuat maka akan terdapat pesan/alert yang dihasilkan.</li> </ul> <p>Database hasil log antara snort dan suricata dikirimkan terhadap website. Dengan tujuan untuk menampilkan data hasil serangan</p>
Hasil Akhir	Pada hal ini diharapkan dapat menguji kemampuan Pendeteksian Snort dan Suricata dalam mendeteksi serangan Port Scanning serta mengirimkan hasil deteksi kepada website dengan tujuan mempermudah monitoring.

*Table 9 Eksperimen Pengujian Port Scanning*

### 3.4.3 Eksperimen Pengujian SQL Injection

Metode serangan dunia maya yang disebut SQL Injection mengeksploitasi kelemahan keamanan di aplikasi online yang menggunakan database. Pada dasarnya, data pengguna pada formulir atau URL digunakan untuk menyuntikkan kode berbahaya (pernyataan SQL) dalam serangan ini. Jika kelemahan keamanan dieksploitasi secara efektif, penyerang akan dapat mengakses database dan melihat, menambah, mengubah, atau bahkan menghapus data tanpa izin. Eksperimen pengujian penyerangan dengan SQL Injection dapat dilihat sebagai berikut

Kode Pengujian	SK03
Kondisi Awal	<ul style="list-style-type: none"> <li>● Snort dan Suricata telah diinstall pada ubuntu server dan siap</li> </ul>

	<p>dipakai dalam host server</p> <ul style="list-style-type: none"> <li>● DVWA telah berhasil diinstall dan dapat dipergunakan</li> <li>● Pembuatan Rules Denial Of Service di Suricata dan Snort</li> </ul> <p>alert tcp any any -&gt; \$HOME_NET 80 (msg:"SQL Injection Detected"; classtype:web-application-attack; sid:100003; content:"'%25%27"; )</p> <p>Aturan ini bertujuan untuk mendeteksi serangan SQL Injection terhadap aplikasi web yang berjalan di jaringan dengan alamat 192.168.56.0/24 pada port 80. IDS akan mencocokkan konten yang sesuai dengan pola serangan SQL Injection yang ditentukan dalam aturan tersebut. Jika pola serangan terdeteksi, pesan "SQL Injection Detected" akan ditampilkan, dan aktivitas akan diklasifikasikan sebagai serangan terhadap aplikasi web. ID aturan yang digunakan adalah 100002</p>
Alur Skenario	<ul style="list-style-type: none"> <li>● Antara attacker dan target saling terhubung dengan cara saling ping melalui ip address antara target dan attacker</li> <li>● Pada attacker menjalankan command sebagai berikut dalam payload  <pre>'% or '0' = '0</pre> <p>Perintah ini untuk menyerang web vulnerability dengan menggunakan payload</p> </li> <li>● Pada snort dan suricata menjalankan command</li> </ul> <p>Snort</p> <pre>sudo snort -q -l /opt/lampp/htdocs/ids -i enp0s3 -A fast -c /etc/snort/snort.conf</pre>

	<p>Suricata</p> <p>sudo systemctl start suricata</p> <ul style="list-style-type: none"> <li>● Melakukan pemeriksaan log dari snort dan suricata dengan tujuan melihat apakah serangan SQL Injection terdeteksi. Jika sudah sesuai rules yang telah dibuat maka akan terdapat pesan/alert yang dihasilkan.</li> <li>● Database hasil log penggabungan antara snort dan suricata dikirimkan terhadap website. Dengan tujuan untuk menampilkan data hasil serangan</li> </ul>
Hasil Akhir	Pada hal ini diharapkan dapat menguji kemampuan Sentralisasi Data Snort dan Suricata dalam mendeteksi serangan SQL Injection serta memperoleh informasi untuk melakukan peningkatan performa sistem deteksi serangan tersebut.

*Table 10 Eksperimen Pengujian SQL-Injection DVWA*

## **BAB IV IMPLEMENTASI**

Pada bab berisi uraian mengenai tahapan implementasi setiap tools yang akan digunakan dalam pengerjaan tugas akhir. Adapun implementasi yang dilakukan mencakup tahap instalasi, konfigurasi, integrasi dan sentralisasi serta pengujian sistem sesuai dengan skenario pengujian yang telah dijelaskan pada bab sebelumnya

### **4.1 Instalasi dan Konfigurasi**

Pada sub bab ini dijelaskan mengenai proses instalasi dan konfigurasi yang digunakan dalam tahap pengerjaan tugas akhir. Instalasi dan konfigurasi dilakukan pada tools snort, suricata, web server pada virtual box.

#### **4.1.1 Instalasi dan Konfigurasi Snort**

Pada bagian ini, menjelaskan mengenai tahap-tahap dalam melakukan instalasi dan konfigurasi snort dalam mode ids. Menyiapkan snort di ubuntu VM dari source code perlu melakukan beberapa langkah yaitu mengunduh code, melakukan konfigurasi, menjalankan code, serta instalasi ke direktori yang sesuai, dan terakhir mengkonfigurasi rules. Berikut cara yang perlu dilakukan dalam instalasi dan konfigurasi snort pada virtual machine ubuntu.

1. Persiapan pertama yaitu mengupdate sistem terlebih dahulu dengan menjalankan perintah berikut

```
sudo apt-get update
```

2. Kemudian melakukan cek jaringan dengan menjalankan perintah ifconfig dan mencatat nama interface serta ip address yang dibutuhkan saat mengkonfigurasi

snort nantinya

```
sudo ifconfig
```

3. Setelah itu lanjutkan dengan menginstal snort sebagai IDS dengan menggunakan perintah

```
sudo apt-get install snort -y
```

Selama proses penginstalan, akan diminta untuk memasukkan network ip yang akan dimonitoring diikuti dengan range ip. Isi bagian tersebut dengan network ip yang dimiliki oleh Ubuntu VM

4. Perlu juga untuk mengaktifkan promiscuous pada interface agar memungkinkan untuk menangkap dan memproses semua lalu lintas jaringan dengan perintah

```
sudo ip link set enp0s3 promisc on
```

5. Agar direktori file program snort bisa di akses, berikutnya adalah memberi hak akses pada direktori /etc/snort/ dengan menjalankan perintah berikut

```
sudo chmod 777 /etc/snort
```

6. Lakukan konfigurasi jaringan dan membuat rules. Untuk melakukan konfigurasi terhadap jaringan dan rules pada snort bisa dilakukan pada file snort.conf yang berada pada direktori /etc/snort untuk mengubah parameter sesuai yang dibutuhkan. Kita perlu mengkonfigurasi file utama snort yaitu /etc/snort/snort.conf. masuk ke file tersebut dengan perintah

```
sudo nano /etc/snort/snort.conf
```

7. Kemudian matikan file rules yang tidak dibutuhkan, karena kita ingin membuat file rule dengan rule baru maka kita harus mematikan file rules lain yang pada file /etc/snort/snort.conf dengan cara memberi tanda pagar di depan path file rule

```
#include $RULE_PATH/icmp-info.rules  
#include $RULE_PATH/icmp.rules
```

8. Selanjutnya adalah melakukan validasi. Snort harus siap dijalankan dengan menguji konfigurasinya menggunakan parameter -T dan untuk mengaktifkan mode uji. Jalankan perintah berikut

```
sudo snort -T -c /etc/snort/snort.conf
```

9. Selanjutnya mengatur letak file log dari snort dengan cara mengganti default path direktori file log ke /opt/lampp/htdocs/ids yaitu path web server agar kode program mudah dalam membaca file log dan mudah untuk mengimport nya ke dalam database

#### 4.1.1.1 Rules Snort

Jika kita menginstall snort dengan baik maka semua aturan snort akan disimpan di direktori /etc/snort/rules. Bagi kita yang ingin mencoba membuat rules sendiri, dapat mengedit file yang berada di /etc/snort/rules/local.rules.

Aturan penulisan snort antara lain sebagai berikut:

## 1. Penulisan aturan snort secara umum

```
action protocol source_ip source_port direction  
destination_ip destination_port (rule option)
```

Penjelasan sebagai berikut

- Pada bagian action yang dipakai adalah alert yang berfungsi menghasilkan sebuah output alert ke sebuah file
- Pada bagian protokol mempunyai pilihan seperti “tcp”, “udp” atau “icmp”. Sesuai serangan apa yang kemungkinan terdeteksi atau ingin di deteksi
- Pada bagian ip dapat berisi ip sumber atau tujuan, bisa ip yang ingin di monitoring keamanan nya atau ip yang ingin dipantau karena mungkin berbahaya
- Pada bagian port dapat berisi port sumber atau tujuan, bisa berisi ip yang ingin di monitoring keamanan nya atau ip yang menjadi sumber serangan
- Pada bagian direction atau operator arah bisa berisi “->”, “<-” atau “<>” untuk lalu lintas bi-directional antara dua address

alert icmp any any -> 192.168.1.1/24 80 (rule option)

Ketika saat kita ingin membuat rules pendeteksi serangan SQLI terlebih dahulu kita harus mengetahui port berapa biasanya SQLI bekerja. Pada umumnya sqli menyerang pada port web server 80 yang sedang terbuka atau digunakan ini bisa di check menggunakan tool nmap

- ## 2. Setelah mengetahui nomor port, maka kita dapat membuat aturan snort sederhana untuk mendeteksi telnet, sebagai berikut

```
alert tcp any any -> 192.168.148.0/24 80 (msg:"ada sqli";  
content:"%27"; sid:1000001;)
```

Berdasarkan rules di atas, dapat dijelaskan bahwa:

- Action: akan memberikan sebuah alert
- Content : semua payload dengan code ascii %27 yang melewati ip 192.168.114.0/24 dan port 80 akan di pantau
- Message : untuk membantu pengguna deskripsi dari sebuah alert yang dihasilkan oleh snort
- Sid: sid merupakan id alert yang dimiliki setiap rules dan harus bersifat unik dimulai dari 1000001

#### 4.1.2 Instalasi dan Konfigurasi Suricata

Pada bagian ini, menjelaskan mengenai tahap-tahap dalam melakukan instalasi dan konfigurasi suricata dalam mode ids anomaly based. Menyiapkan suricata di ubuntu VM dari source code perlu melakukan beberapa langkah yaitu mengunduh code, mengkonfigurasinya, mengcompile code, menginstal-nya ke direktori yang sesuai, dan terakhir mengonfigurasi rules. Berikut cara yang perlu dilakukan dalam penginstalan dan konfigurasi suricata pada ubuntu vm

1. Persiapan pertama yaitu mengupdate sistem terlebih dahulu dengan menjalankan perintah berikut

```
sudo apt-get update
```

2. Setelah itu lanjutkan dengan menginstal suricata sebagai IDS dengan menggunakan



perintah

```
sudo apt-get install software-properties-common
sudo add-apt-repository ppa:oisf/suricata-stable
sudo apt-get install suricata
```

3. Selanjutnya harus mengkonfigurasi ip address yang akan dimonitoring oleh suricata dan memasukkan interface yang benar pada file konfigurasi /etc/suricata/suricata.yaml

```
HOME_NET: "[192.168.114.0/24]"

af-packet:
    interface: enp0s3

pcap:
    interface: enp0s3
```

4. Selanjutnya aktifkan mode anomaly based suricata pada file /etc/suricata/suricata.yaml dengan cara

```
anomaly:
    enabled: yes
```

5. Kemudian kita mengkonfigurasi file rules yang baru dan yang tidak dipakai dengan cara memberi “#” pada parameter jika ingin memaatikannya dan menambah file local.rules sebagai tempat rules baru nanti pada file /etc/suricata/suricata.yaml

```
rule-file:
    #-suricata.rule
    local.rules
```

6. Lalu mengganti direktori file log suricata pada file `/etc/suricata/suricata.yaml` dengan cara

```
fast:
  filename:
    /opt/lampp/htdocs/ids/fast.log
```

7. Setelah melakukan konfigurasi maka sekarang harus melakukan validasi untuk mengecek ada atau tidak kesalahan dalam konfigurasi dengan cara

```
sudo suricata-update
sudo suricata -T -c
/etc/suricata/suricata.yaml -v
```

8. Lalu untuk menjalankan suricata cukup dengan menjalankan perintah

```
sudo systemctl start suricata
sudo systemctl status suricata
```

#### 4.1.2.1 Rules Suricata

Jika kita menginstall suricata dengan baik maka semua aturan suricata akan disimpan di direktori `/etc/suricata/rules`. Bagi kita yang ingin mencoba membuat rules sendiri, dapat mengedit file yang berada di `/etc/suricata/rules/local.rules`.

Aturan penulisan suricata antara lain sebagai berikut:

1. Penulisan aturan suricata secara umum:

```
action protocol source_ip source_port direction
destination_ip destination_port (rule option)
```

Penjelasan format:

- Pada action yang dipakai adalah alert yang berfungsi menghasilkan sebuah output alert ke sebuah file
- Pada protokol mempunyai pilihan seperti “tcp”, “udp” atau “icmp”. Sesuai serangan apa yang kemungkinan terdeteksi atau ingin di deteksi
- Pada ip dapat berisi ip sumber atau tujuan, bisa ip yang ingin di monitoring keamanan nya atau ip yang ingin dipantau karena mungkin berbahaya
- Pada port dapat berisi port sumber atau tujuan, bisa berisi ip yang ingin di monitoring keamanan nya atau ip yang menjadi sumber serangan
- Pada direction atau operator arah bisa berisi “->”, “<-” atau “<>” untuk lalu lintas bi-directional antara dua address

```
alert icmp any any -> 192.168.114.0/24 80 (rule option)
```

#### 4.1.3 Instalasi dan Konfigurasi Web Server

Pada bagian ini dilakukan instalasi dan konfigurasi web server sebagai salah satu server yang diinstal pada ubuntu. Web server ini nantinya akan digunakan untuk menjalankan tool sqli yaitu DVWA. Pada web server diperlukan untuk menginstal apache

web server, database mysql dan php. Langkah-langkah instalasi web server antara lain sebagai berikut

#### A. Menginstal Apache

1. Sebelum melakukan instalasi web server, update terlebih dahulu seluruh packages jalankan perintah berikut

```
sudo apt-get update
```

```
sudo apt install apache2
```

2. selanjutnya pada ubuntu lakukan instalasi apache web server dengan perintah di bawah sebagai berikut.

#### B. Menginstal mysql

1. Setelah apache web server sudah berjalan, selanjutnya adalah melakukan instalasi terhadap MySQL. MySQL adalah manajemen database yang digunakan dalam lingkungan PHP

```
sudo apt install mysql-server
```

2. Selanjutnya ketika sudah selesai, lakukan pengujian apakah dapat masuk ke terminal MySQL dengan input perintah di bawah ini.

```
sudo mysql
```

#### C. Menginstal xampp

1. Download installer xampp dari internet

2. Buka direktori download lalu ubah hak akses kemudian jalan installer pada terminal

```
cd Downloads
chmod 777 xampp-linux-*-installer.run
sudo ./xampp-linux-*-installer.run
```

3. Jika ada jendela baru muncul maka klik next tunggu sampai proses instalasi selesai
4. Kemudian jalankan xampp dengan perintah

```
sudo /opt/lampp/./manager-linux-x64.run
```

5. Setelah jendela xampp terbuka jalankan mysql service dan apache service

#### D. Menginstal PHP

1. Setelah berhasil menginstal web server Apache dan MySQL, selanjutnya adalah menginstall PHP. PHP adalah salah satu kode program untuk menampilkan data secara dinamis. Jalankan perintah berikut untuk menginstall PHP. Dependensi yang perlu diinstal yaitu paket php, php-mysql untuk memungkinkan PHP berkomunikasi dengan database MySQL, serta menginstall libapache2-mod-php untuk mengaktifkan Apache dalam menangani file PHP. Paket inti PHP akan secara otomatis diinstal sebagai dependensi. Langkah pertama jalankan perintah berikut ini

```
sudo apt install php libapache2-mod-php php-mysql
```

2. Pada pengaturan DirectoryIndex di Apache, file bernama index.html akan selalu didahulukan dari file index.php. Edit file `/etc/apache2/mods-`

enabled/dir.conf file agar index.php lebih didahulukan agar menjadi halaman awal. Setelah proses edit selesai, index.html namanya diganti atau dihapus dari root dokumen. Jalankan perintah berikut:

```
sudo nano /etc/apache2/mods-enabled/dir.conf
```

Apabila berhasil masuk ke konsol maka akan terhubung ke server MySQL sebagai root pengguna database administratif, yang disimpulkan dengan penggunaan sudo saat menjalankan perintah ini.

3. Setelah menyimpan dan menutup file, lanjutkan dengan memuat ulang Apache agar perubahan disimpan.

```
sudo systemctl reload apache2
```

## 4.2 Sentralisasi Data

Pada sub-bab ini dipaparkan terkait mekanisme sentralisasi data yang difungsikan pada tahap pengerjaan tugas akhir ini Sentralisasi data yang letak filenya diubah menjadi satu direktori dalam opt/lampp/htdocs/ dengan folder ids dan selanjutnya dikirimkan ke dalam phpMyAdmin dan setelah dikirimkan diolah menjadi sebuah database suatu sistem.

### 4.2.1 Sentralisasi Data dari Snort (Signature) & Suricata (Anomaly)

Pada bagian ini dipaparkan bagaimana proses pengiriman atau sentralisasi data dari Snort yang berperan sebagai IDS-Signature menjadi sebuah database sebuah system dalam website

Kode yang dipergunakan dalam pengiriman data dari opt/lampp/htdocs/ids

menjadi database dalam phpmyadmin menggunakan bahasa pemrograman PHP yang mengelola data dari file "alert" untuk snort dan file "fast.log" untuk suricata dan memasukkannya ke dalam database MySQL. Berikut adalah klasifikasi dan penjelasan untuk setiap bagian kode tersebut:

#### 1. Mengatur Koneksi ke Database:

```
$host = 'localhost';
$username = 'antonel';
$password = '1234underworld1234';
$database = 'ids';
$conn = new mysqli($host, $username, $password, $database);
```

Penjelasan dari kode tersebut sebagai berikut :

Bagian ini berisi kode yang digunakan untuk membuat koneksi ke database MySQL. Koneksi ini digunakan untuk melakukan operasi database seperti pengambilan data, penambahan data, dan pembaruan data. Kode ini menggunakan objek mysqli untuk melakukan koneksi.

#### 2. Membaca Isi File "alert":

```
$data1 = file_get_contents('alert');
$lines1 = explode("\n", $data1);

foreach ($lines1 as $line) {
    // Pemrosesan dan Ekstraksi Data dari $line
    $data = explode(",", $line);
    $timestamp = $data[0];
    $sourceIP = $data[1];
    $destinationIP = $data[2];
    // ...

    // Penyimpanan Data ke Tabel "snort" dengan query INSERT
    $query = "INSERT INTO snort (timestamp, source_ip,
destination_ip) VALUES ('$timestamp', '$sourceIP',
'$destinationIP')";
    $conn->query($query);
}
```

Bagian ini membaca file alert dan mengolah setiap barisnya. Setiap baris

dipisahkan menjadi kolom-kolom menggunakan karakter pemisah spasi. Nilai-nilai kolom yang diperoleh kemudian digunakan untuk melakukan operasi pada database. Jika data dengan timestamp yang sama belum ada di tabel snort, maka data akan dimasukkan ke tabel snort menggunakan perintah SQL INSERT.

### 3. Pengolahan data dari file fast.log dalam tabel suricata

```
$data2 = file_get_contents('fast.log');
$lines2 = explode("\n", $data2);

foreach ($lines2 as $line) {

    $data = explode(" ", $line);
    $timestamp = $data[0];
    $sourceIP = $data[1];
    $destinationIP = $data[2];
    // ...

    $query = "INSERT INTO suricata (timestamp, source_ip,
destination_ip) VALUES ('$timestamp', '$sourceIP',
'$destinationIP')";
    $conn->query($query);
}
```

Bagian ini mirip dengan bagian sebelumnya, namun menggunakan file fast.log dan tabel suricata. Data yang diperoleh juga dimasukkan ke database menggunakan perintah SQL INSERT. Jika terdapat data dengan timestamp yang sama di tabel suricata, status2 pada data tersebut akan diperbarui.

### 4. Pemrosesan data dari tabel snort dan menampilkan dalam monitor



```

$snortQuery = "SELECT * FROM snort";
$snortResult = $conn->query($snortQuery);

if ($snortResult->num_rows > 0) {
    while ($row = $snortResult->fetch_assoc()) {
        // Pemrosesan Data dari Tabel "snort"
        $timestamp = $row['timestamp'];
        $sourceIP = $row['source_ip'];
        $destinationIP = $row['destination_ip'];
        // ...

        // Penyimpanan Data ke Tabel "monitor" dengan query INSERT
        $query = "INSERT INTO monitor (timestamp, source_ip,
destination_ip) VALUES ('$timestamp', '$sourceIP', '$destinationIP')";
        $conn->query($query);
    }
}

```

Bagian ini mengambil data dari tabel snort menggunakan perintah SQL SELECT dan melakukan pengolahan data untuk dimasukkan ke tabel monitor. Data yang sudah ada di tabel monitor tidak akan dimasukkan kembali.

## 5. Pemrosesan Data dari Tabel Suricata

```

$suricataQuery = "SELECT * FROM suricata";
$suricataResult = $conn->query($suricataQuery);

if ($suricataResult->num_rows > 0) {
    while ($row = $suricataResult->fetch_assoc()) {
        // Pemrosesan Data dari Tabel "suricata"
        $timestamp = $row['timestamp'];
        $sourceIP = $row['source_ip'];
        $destinationIP = $row['destination_ip'];
        // ...

        // Penyimpanan Data ke Tabel "monitor" dengan query INSERT
        $query = "INSERT INTO monitor (timestamp, source_ip,
destination_ip) VALUES ('$timestamp', '$sourceIP', '$destinationIP')";
        $conn->query($query);
    }
}

```

Bagian ini mirip dengan bagian sebelumnya, namun menggunakan tabel suricata dan juga melakukan pengolahan data untuk dimasukkan ke tabel monitor.

## 6. Menampilkan data dalam bentuk tabel di page web

```
<!DOCTYPE html>
<html lang="en">
<head>
  <!-- Pengaturan Meta dan Link Eksternal -->
  <title></title>
  <style>
    /* Pengaturan CSS */
  </style>
</head>
<body>
  <!-- Bagian Header dan Navbar -->
  <header>
    <!-- Isi Header -->
  </header>
  <main>
    <!-- Bagian Utama Halaman Web -->
  </main>
  <!-- Bagian JavaScript dan Footer -->
  <script>
    // Pengaturan JavaScript
  </script>
</body>
</html>
```

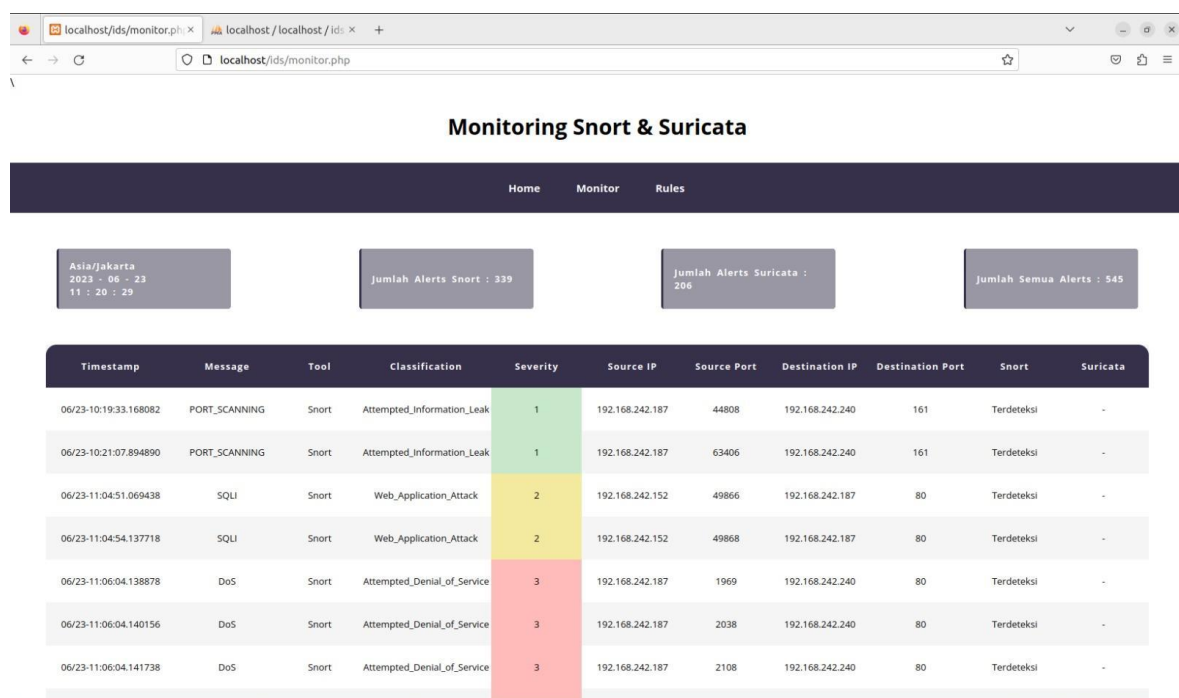
Bagian ini menggunakan HTML dan CSS untuk membuat halaman web yang menampilkan data dari tabel monitor. Tampilan data menggunakan elemen tabel dan berbagai elemen lainnya seperti header, navbar, dan bagian tambahan. Data ditampilkan dalam tabel dengan kolom yang sesuai.

### 4.3 Visualisasi Monitoring Snort dan Suricata

Pada sub-bab ini dipaparkan terkait bagaimana visualisasi data yang difungsikan untuk mempermudah administrator melihat jenis serangan yang terdeteksi secara anomaly ataupun signature. Data yang sudah diterima dalam phpMyAdmin akan diolah menjadi sebuah database dalam sistem.

### 4.3.1 Visualisasi Data Monitoring Snort dan Suricata

Pada visualisasi data serangan Snort dan suricata ditampilkan dalam satu page yang sama. Dalam page tersebut menampilkan jumlah keseluruhan alert yang masuk, jumlah rules, mencatat alamat IP dan timestamp. Tampilan halaman ini akan mencatat hasil pendeteksian serangan yang masuk melalui traffic jaringan dengan data timestamp, message, klasifikasi tingkat serangan, severity, protocol, source IP, source port, Destination IP, Destination Port. Untuk menampilkan data serangan Snort, pada menu monitor, pilih dan klik monitor Snort maka hasil deteksi penyerangan akan tampak pada halaman tersebut.



Timestamp	Message	Tool	Classification	Severity	Source IP	Source Port	Destination IP	Destination Port	Snort	Suricata
06/23-10:19:33.168082	PORT_SCANNING	Snort	Attempted_Information_Leak	1	192.168.242.187	44808	192.168.242.240	161	Terdeteksi	-
06/23-10:21:07.894890	PORT_SCANNING	Snort	Attempted_Information_Leak	1	192.168.242.187	63406	192.168.242.240	161	Terdeteksi	-
06/23-11:04:51.069438	SQLI	Snort	Web_Application_Attack	2	192.168.242.152	49866	192.168.242.187	80	Terdeteksi	-
06/23-11:04:54.137718	SQLI	Snort	Web_Application_Attack	2	192.168.242.152	49868	192.168.242.187	80	Terdeteksi	-
06/23-11:06:04.138878	DoS	Snort	Attempted_Denial_of_Service	3	192.168.242.187	1969	192.168.242.240	80	Terdeteksi	-
06/23-11:06:04.140156	DoS	Snort	Attempted_Denial_of_Service	3	192.168.242.187	2038	192.168.242.240	80	Terdeteksi	-
06/23-11:06:04.141738	DoS	Snort	Attempted_Denial_of_Service	3	192.168.242.187	2108	192.168.242.240	80	Terdeteksi	-

Gambar 18 Visualisasi Monitor Snort & Suricata

### 4.3.2 Visualisasi Rules

Data serangan yang dilakukan berlandaskan pada rules yang sudah di konfigurasi snort dan suricata. Visualisasi Rules Snort dan Suricata dapat dilihat dari gambar berikut

#### 4.3.2.1 Visualisasi Rules Snort

Pada visualisasi rules Snort menampilkan data keseluruhan rules Snort yang sudah dikonfigurasi. Pada visualisasi ini ditampilkan action berupa alert, protocol, ip source, port source, ip destination, port destination, classtype, ID Rules dan message yang berisi pesan peringatan dan informasi paket yang diserang tersebut. Visualisasi rules Snort ini berguna untuk administrator dalam memantau daftar rules apa saja yang sudah berhasil dikonfigurasi. Untuk melihat rules yang sudah terdaftar pada /etc/snort/rules/local.rules, maka kita dapat mengeceknya pada halaman rules Snort.

Snort Rules									
<div>Home Snort Suricata Rules</div> <div>Asia/Jakarta 2023 - 08 - 21 10 : 07 : 12</div> <div>Jumlah Rules : 5</div>									
Action	Protocol	Source IP	Source Port	Direction	Destination IP	Destination Port	Message	Classtype	ID Rules
alert	icmp	any	any	->	any	any	PING	icmp-event	1000001
alert	tcp	any	any	->	\$HOME_NET	80	SQL_injection	web-application-attack	1000002
alert	tcp	any	any	->	\$HOME_NET	80	DoS	attempted-dos	1000003
alert	tcp	any	any	->	\$HOME_NET	80	DoS	attempted-dos	1000004
alert	tcp	any	any	->	\$HOME_NET	161	Port_Scanning	attempted-recon	1000005

Gambar 19 Visualisasi Rules Snort

#### 4.3.2.2 Visualisasi Rules Suricata

Pada visualisasi rules Suricata menampilkan data keseluruhan rules Suricata yang sudah dikonfigurasi. Pada visualisasi ini ditampilkan action berupa alert, protocol, ip source, port source, ip destination, port destination, classtype, ID Rules dan message yang berisi pesan peringatan dan informasi paket yang diserang tersebut. Visualisasi rules Suricata ini

berguna untuk administrator dalam memantau daftar rules apa saja yang sudah berhasil dikonfigurasi. Untuk melihat rules yang sudah terdaftar pada /etc/suricata/rules/local.rules, maka kita dapat mengeceknya pada halaman rules Suricata.

Suricata Rules									
Home Snort Suricata Rules									
Asia/Jakarta 2023 - 08 - 21 10 : 05 : 50					Jumlah Rules : 5				
Action	Protocol	Source IP	Source Port	Direction	Destination IP	Destination Port	Message	Classtype	ID Rules
alert	icmp	any	any	->	any	any	PING	icmp-event	1000001
alert	tcp	any	any	->	\$HOME_NET	80	SQL_injection	web-application-attack	1000002
alert	tcp	any	any	->	\$HOME_NET	80	DoS	attempted-dos	1000003
alert	tcp	any	any	->	\$HOME_NET	80	DoS	attempted-dos	1000004
alert	tcp	any	any	->	\$HOME_NET	161	Port_Scanning	attempted-recon	1000005

Gambar 20 Visualisasi Rules Suricata

## 4.4 Pengujian

Pada bagian ini akan dipaparkan terkait pengujian yang ditugaskan dalam setiap serangan yang tersedia.

### 4.4.1 Pengujian Serangan Denial Of Service

Dalam sub bab ini dideskripsikan terkait pengujian sentralisasi dalam sistem yang dibuat. Hal pertama yang diperhatikan yakni bahwa antara attacker dan server dalam jaringan yang sama dan dapat melakukan saling komunikasi. Selanjutnya melakukan uji coba penyerangan dengan denial of service yakni dengan tools hping3 kemudian rules IDS baik suricata dan snort dipaparkan.

```

antonel@antonel:~$ sudo hping3 -S --flood -V -p 80 192.168.56.240
using enp0s3, addr: 192.168.56.187, MTU: 1500
HPING 192.168.56.240 (enp0s3 192.168.56.240): S set, 40 headers + 0 data bytes
hping in flood mode, no replies will be shown
^C
--- 192.168.56.240 hping statistic ---
41238 packets transmitted, 0 packets received, 100% packet loss
round-trip min/avg/max = 0.0/0.0/0.0 ms
antonel@antonel:~$

```

Gambar 21 Penyerangan Denial Of Service

Setelah melakukan serangan DoS, maka dilakukan pengamatan yakni jika snort dan suricata berhasil mendeteksi serangan DoS, maka kedua sistem dapat berfungsi dengan baik dalam melindungi jaringan atau server dari serangan DoS. Kemudian alert hasil pendeteksian itu akan dikirimkan dalam database system dengan tujuan untuk menjadi database system.

06/23-11:06:04.138878	1:1000005:0 DoS	Snort	Attempted_Denial_of_Service 3	TCP	192.168.242.187 1969	192.168.242.240 80	Terdeteksi -
06/23-11:06:04.140156	1:1000005:0 DoS	Snort	Attempted_Denial_of_Service 3	TCP	192.168.242.187 2038	192.168.242.240 80	Terdeteksi -

Gambar 22 Pendeteksian DoS oleh Snort

06/23/2023-11:07:06.862557 1:4:0	DoS	Suricata	Attempted_Denial_of_Service 3	TCP	192.168.242.187 2780	192.168.242.240 80	-	Terdeteksi
06/23/2023-11:07:06.865545 1:5:0	DoS	Suricata	Attempted_Denial_of_Service 3	TCP	192.168.242.187 2850	192.168.242.240 80	-	Terdeteksi

Gambar 23 Pendeteksian DoS oleh Suricata

#### 4.4.2 Pengujian Serangan Port Scanning

Dalam sub bab ini dideskripsikan terkait pengujian sentralisasi dalam sistem yang dibuat. Hal pertama yang diperhatikan yakni bahwa antara attacker dan server dalam jaringan yang sama dan dapat melakukan komunikasi. Selanjutnya melakukan uji coba penyerangan dengan Port Scanning yakni dengan bantuan dari nmap kemudian rules IDS baik suricata dan snort untuk proses pendeteksian. Proses penyerangan dilakukan dalam host atau jaringan yang terhubung dengan uji coba untuk melihat apakah snort dan suricata dapat mendeteksi serangan tersebut.

```

antonel@antonel:/etc/snort/rules$ sudo nmap -p- 192.168.56.240
Starting Nmap 7.80 ( https://nmap.org ) at 2023-06-05 22:20 WIB
Nmap scan report for 192.168.56.240
Host is up (0.0055s latency).
Not shown: 65531 closed ports
PORT      STATE SERVICE
21/tcp    open  ftp
80/tcp    open  http
443/tcp   open  https
3306/tcp  open  mysql
MAC Address: 08:00:27:56:7D:EE (Oracle VirtualBox virtual NIC)

Nmap done: 1 IP address (1 host up) scanned in 18.69 seconds
antonel@antonel:/etc/snort/rules$

```

Gambar 24 Penyerangan Port Scanning

06/23-10:19:33.168082	1:1000006:0	PORT_SCANNING	Snort	Attempted_Information_Leak	1	TCP	192.168.242.187	44808	192.168.242.240	161	Terdeteksi	-
06/23-10:21:07.894890	1:1000006:0	PORT_SCANNING	Snort	Attempted_Information_Leak	1	TCP	192.168.242.187	63406	192.168.242.240	161	Terdeteksi	-

Gambar 25 Pendeteksian Port Scanning oleh Snort

06/23-2023-10:48:34.255216	1:6:0	PORT_SCANNING	Suricata	Attempted_Information_Leak	1	TCP	192.168.242.187	41787	192.168.242.240	161	-	Terdeteksi
06/23-2023-10:48:36.799249	1:7:0	PORT_SCANNING	Suricata	Attempted_Information_Leak	1	TCP	192.168.242.187	41787	192.168.242.240	705	-	Terdeteksi

Gambar 26 Pendeteksian Port Scanning oleh Suricata

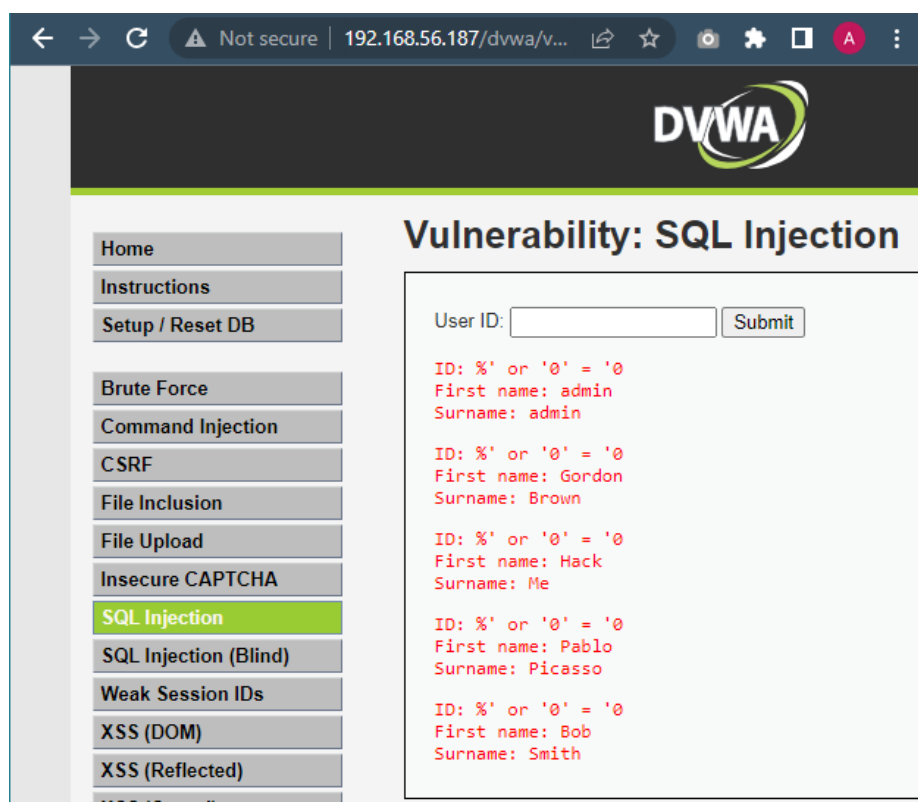
Setelah melakukan serangan Port Scanning maka dilakukan pendeteksian yakni jika snort dan suricata berhasil mendeteksi serangan Port Scanning, maka kedua sistem dapat menghasilkan log dengan berisikan informasi terkait penyerangan. Kemudian alert yang muncul akan terkirimkan ke dalam phpMyAdmin dan menjadi sebuah database.

#### 4.4.3 Pengujian Serangan SQL-Injection(DVWA)

Dalam sub bab ini dideskripsikan terkait pengujian sentralisasi dalam sistem yang dibuat. Hal pertama yang diperhatikan yakni bahwa antara attacker dan server dalam jaringan yang sama dan dapat melakukan komunikasi. Selanjutnya melakukan uji coba penyerangan dengan Port Scanning yakni dengan bantuan dari DVWA kemudian rules IDS baik suricata dan snort untuk proses pendeteksian. Proses penyerangan dilakukan dalam sebuah webserver DVWA, pada bagian navigasi terdapat bagian SQL-Injection .Setelah melakukan serangan SQL-I maka dilakukan analisis yakni jika snort dan

suricata berhasil mendeteksi serangan Port Scanning, maka kedua sistem dapat menghasilkan log dengan berisikan informasi terkait penyerangan. Kemudian alert yang muncul akan dikirimkan ke dalam phpMyAdmin dan menjadi sebuah database.

Berikut adalah pengujian menggunakan serangan sql injection menggunakan aplikasi web DVWA, Pertama dilakukan adalah uji coba serangan SQL Injection. Pada aplikasi web DVWA di bagian SQL Injection, terdapat form yang perlu di input untuk memasukkan User ID. Pada bagian ini kami melakukan uji coba dengan menginput '5' or '0' = 0 yang akan menghasilkan output seperti gambar di bawah ini.

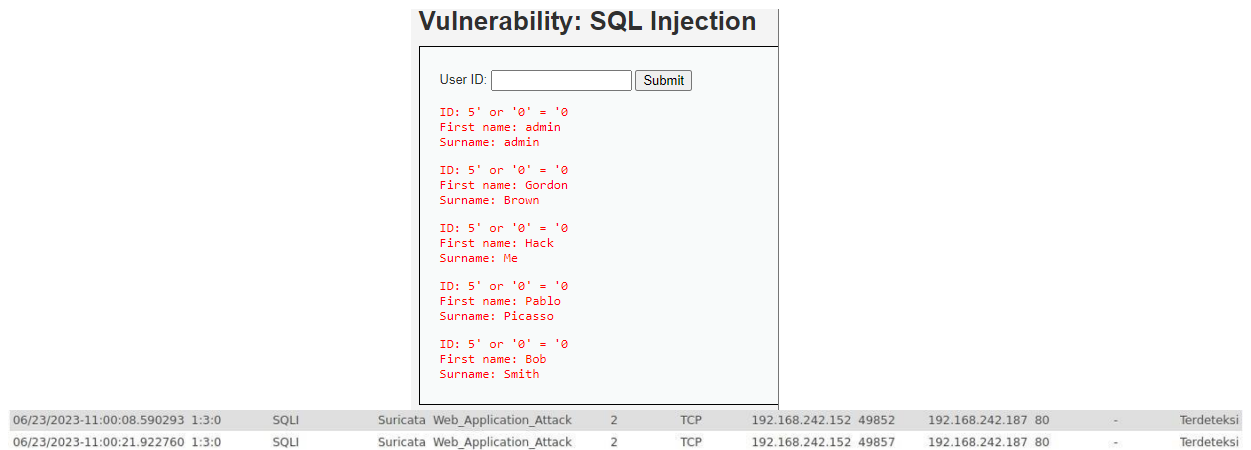


Gambar 27 Penyerangan SQL-Injection DVWA

06/23-11:04:51.069438	1:1000003:0	SQLI	Snort	Web_Application_Attack	2	TCP	192.168.242.152	49866	192.168.242.187	80	Terdeteksi -
06/23-11:04:54.137718	1:1000003:0	SQLI	Snort	Web_Application_Attack	2	TCP	192.168.242.152	49868	192.168.242.187	80	Terdeteksi -

Gambar 28 Pendeteksian SQLInjection DVWA oleh Snort





Gambar 29 Pendeteksian SQL-Injection DVWA oleh Suricata

Setelah melakukan serangan SQL-Injection DVWA, maka dilakukan pengamatan yakni jika snort dan suricata berhasil mendeteksi serangan DoS, maka kedua sistem dapat berfungsi dengan baik dalam melindungi jaringan atau server dari serangan DoS. Kemudian alert hasil pendeteksian itu akan dikirimkan dalam database system dengan tujuan untuk menjadi database system.

## 4.5 Hasil Pengujian

Pada bagian ini akan dipaparkan terkait hasil pengujian yang ditugaskan dalam setiap serangan yang tersedia.

### 4.5.1 Hasil Pengujian Serangan Denial Of Service

Secara keseluruhan, hasil pengujian Denial Of Service dilakukan sebanyak 25 dengan klasifikasi/level serangan diuji sebanyak 15 kali untuk high dan masing-masing 10 kali untuk klasifikasi lainnya. Dalam 25 kali percobaan yang dilakukan, pengujian dibedakan berdasarkan fungsi nmap pada terminal ubuntu.

Berdasarkan masing-masing klasifikasi, kondisi yang seharusnya didapatkan adalah sebagai berikut:

1. Apabila berada pada klasifikasi “HIGH”, maka penyerangan yang dideteksi seharusnya berada pada klasifikasi “HIGH” dan tidak mampu sentralisasi data secara otomatis.

#### **4.5.2 Hasil Pengujian Serangan Port Scanning**

Secara keseluruhan, hasil pengujian Port Scanning dilakukan sebanyak 25 dengan klasifikasi/level serangan diuji sebanyak 15 kali untuk high dan masing-masing 10 kali untuk klasifikasi lainnya. Dalam 25 kali percobaan yang dilakukan, pengujian dibedakan berdasarkan fungsi hping3 pada terminal ubuntu.

Berdasarkan masing-masing klasifikasi, kondisi yang seharusnya didapatkan adalah sebagai berikut:

1. Apabila berada pada klasifikasi “low”, maka penyerangan yang dideteksi seharusnya berada pada klasifikasi “low” dan mampu sentralisasi data secara otomatis.

#### **4.5.3 Hasil Pengujian Serangan SQL-Injection**

Secara keseluruhan, hasil pengujian SQL-Injection dilakukan sebanyak 25 dengan klasifikasi/level serangan diuji sebanyak 15 kali untuk high dan masing-masing 10 kali untuk klasifikasi lainnya. Dalam 25 kali percobaan yang dilakukan berdasarkan User ID yang diinput pada aplikasi web DVWA. User ID tersebut dibedakan tiap percobaan uji untuk mendapatkan data yang lebih bervariasi.

Berdasarkan masing-masing klasifikasi, kondisi yang seharusnya didapatkan adalah sebagai berikut:

1. Apabila berada pada klasifikasi “medium”, maka penyerangan yang dideteksi seharusnya berada pada klasifikasi “medium” dan mampu sentralisasi data secara otomatis.

## **BAB V**

### **KESIMPULAN**

Pada bab ini dideskripsikan tinjauan pustaka yang akan digunakan sebagai dasar teori dalam penyusunan Tugas Akhir.

#### **5.1 Kesimpulan**

Penulisan Tugas Akhir ini bermaksud untuk melihat bagaimana hasil sentralisasi data dari tools IDS yakni Snort dan Suricata. Berdasarkan hasil pengujian yang telah dilakukan, dapat diambil kesimpulan antara lain sebagai berikut:

1. Sistem sentralisasi Data Snort dan Suricata berhasil mengirimkan data direktori ke sebuah pusat database mysql.
2. Setelah data serangan berhasil diterima, data serangan tersebut otomatis ditampilkan ke dalam website.

#### **5.2 Saran**

Terdapat beberapa saran untuk penelitian selanjutnya terhadap Tugas Akhir ini antara lain sebagai berikut:

1. Sistem yang dibangun hanya mampu memonitor serangan Port Scanning, Denial Of Service dan SQL Injection.
2. Pengujian berikutnya diharapkan dapat dilanjutkan dengan melakukan testing penyerangan dari jaringan luar.
3. Pada visualisasi aplikasi web hanya menampilkan data serangan dan diharapkan penelitian selanjutnya dapat menambahkan fitur lain seperti alert dan notifikasi agar dapat mempermudah administrator jaringan dalam memonitor serangan yang masuk.

## Daftar Pustaka dan Rujukan

- [1] I. Anugrah and R. H. Rahmanto, "Sistem Keamanan Jaringan Local Area Network Menggunakan Teknik De-Militarized Zone," PIKSEL Penelit. Ilmu Komput. Sist. Embed. Log., vol. 5, no. 2, pp. 91–106, 2018, doi: 10.33558/piksel.v5i2.271.
- [2] Q. D. and Yudiastuti, "MONITORING KEAMANAN JARINGAN KOMPUTER MELALUI NOTIFIKASI DI SMARTPHONE DENGAN SNORT PADA DINAS KEBUDAYAAN DAN PARIWISATA PALEMBANG," Univ. Bina, [Online]. Available: <http://repository.binadarma.ac.id/id/eprint/1352>
- [3] W. D., "Sistem Keamanan Jaringan Komputer Menggunakan SNORT," J. Teknol. Inf. dan Ter., vol. 02, pp. 171–175, 2016.
- [4] "BSSN: Terjadi 1,6 Miliar Serangan Siber Sepanjang 2021 | Neraca.co.id." <https://www.neraca.co.id/article/169222/bssn-terjadi-16-miliar-serangan-siber-sepanjang-2021> (accessed Nov. 10, 2022).
- [5] "BSSN catat 741 juta kali serangan siber periode Januari-Juli 2021 - ANTARA News." <https://www.antaranews.com/berita/2347446/bssn-catat-741-juta-kali-serangan-siber-periode-januari-juli-2021> (accessed Nov. 10, 2022).
- [6] A. Hasnul, Jaringan Komputer & Koneksi Internet, MediaCom. Jakarta, 2011.
- [7] R. U. Rehman, Intrusion Detection Systems with Snort. New Jersey: Prentice Hall PTR, 2003.
- [8] D. Ariyus, Intrusion Detection System. Yogyakarta: Andi Yoga, 2007.
- [9] J. Gondohanindijo, "Sistem Untuk Mendeteksi Adanya Penyusup (IDS : Intrusion Detection System)."
- [10] P. Agarwal and S. Rani Satapathy, "Integration of Signature Integration of Signature Integration of Signature B B B Based and Anomaly ased and Anomaly ased and Anomaly B B B Based ased ased ased Detection Detection Detection," IJCSN Int. J. Comput. Sci. Netw., vol. 3, no. 3, pp. 274–2277, 2014, [Online]. Available:

- [11] D. R. Arrasy and A. Noertjahyana, “Analisis Perbandingan Keakuratan Deteksi Serangan Dan Efisiensi Pemakaian CPU Resources Dari Tools Pendeteksi Serangan Snort Dan Suricata Yang Dipasang Di Web Server,” *J. Infra*, vol. 10, no. 1, 2022.
- [12] “Pengertian Jaringan Komputer: Manfaat, Jenis, Macamnya [LENGKAP].” <https://www.nesabamedia.com/pengertian-jaringan-komputer/> (accessed Nov. 11, 2022).
- [13] S. Rosnawati, “Penilaian Resiko Jaringan Komputer Menggunakan Framework Nist Sp 800-30 Revisi 1 Pada Smk Muhammadiyah 2 Pekanbaru,” vol. 8, no. 2, pp. 189–195, 2021, [Online]. Available: <http://repository.uin-suska.ac.id/42746/>
- [14] P. Holbrook, “Network Working Group,” 1991.
- [15] W. Stallings, *Network & Internetwork Security*. Prentice Hall, 1995.
- [16] Etza nofarita, “Implementasi Aplikasi Software Natural Network Mendeteksi Tingkatan Serangan Ddos Pada Jaringan Komputer,” *Elkom J. Elektron. dan Komput.*, vol. 14, no. 2, pp. 268–277, 2021, doi: 10.51903/elkom.v14i2.501.
- [17] E. Butu et al., “Pencegahan Penyerangan Spoofing Dengan Filtering Routers Dan Aplikasi Colasoft Mac”.
- [18] A. Syaimi, P. Utami, L. Lidyawati, and Z. Ramadhan, “Perancangan dan Analisis Kinerja Sistem Pencegahan Penyusupan Jaringan Menggunakan Snort IDS dan Honeyd,” *J. Reka Elkomika*  
©TeknikElektro | Itenas J. Online Inst. Teknol. Nas. J. Reka Elkomika, vol. 1, no. 4, pp. 2337–439, 2013.
- [19] Y. Yulianingsih, “Menangkal Serangan SQL Injection Dengan Parameterized Query,” *J. Edukasi dan Penelit. Inform.*, vol. 2, no. 1, pp. 46–49, 2016, doi: 10.26418/jp.v2i1.15507.
- [20] M. S. Hasibuan and L. M. Gultom, “Analisis Serangan Deface Menggunakan Backdoor Shell Pada Website,” *Techno.Com*, vol. 17, no. 4, pp. 415–423, 2018, doi: 10.33633/tc.v17i4.1887.

- [21] R. A. R. Ashfaq, X. Z. Wang, J. Z. Huang, H. Abbas, and Y. L. He, "Fuzziness based semi-supervised learning approach for intrusion detection system," *Inf. Sci. (Ny).*, vol. 378, pp. 484–497, Feb. 2017, doi: 10.1016/J.INS.2016.04.019.
- [22] Anitha, "Network Security Using Linux Intrusion Detection System," *Int. J. Res. Comput. Sci.*, vol. 2, no. 1, 2011.
- [23] S. J, "Diverse methods for signature based intrusion detection schemes adopted," *Int. J. Recent Technol. Eng.*, vol. 9, no. 2, 2020.
- [24] G. B. Gunawan, P. Sukarno, and A. G. Putrada, "Pendeteksian Serangan Denial of Service (DoS) pada Perangkat Smartlock Berbasis Wifi Menggunakan SNORT IDS".
- [25] dan R. B Widyanto, R. Munadi and Mayasari, "Implementasi dan Analisis Perbandingan Performansi VoIP Server pada VPS Berbasis OpenVZ dan Cloud Computing," *e-Proceeding Eng.*, vol. 2, no. 2, pp. 3195–3202, 2015.
- [26] A. Alhomoud, R. Munir, J. P. Disso, I. Awan, and A. Al-Dhelaan, "Performance evaluation study of Intrusion Detection Systems," *Procedia Comput. Sci.*, vol. 5, pp. 173–180, 2011, doi: 10.1016/J.PROCS.2011.07.024.
- [27] B. S. Anggoro and W. Sulisty, "Implementasi Intrusion Prevention System Suricata dengan Anomaly- Based untuk Keamanan Jaringan PT. Grahamedia Informasi," *Semin. Nas. APTIKOM*, p. 2019, 2019.
- [28] J. Gondohanindijo, *Sistem Untuk Mendeteksi Adanya Penyusup ( IDS : Intrusion Detection System )*. Semarang, 2011.
- [29] M. Fluorida Fibrianda and A. Bhawiyuga, "Analisis Perbandingan Akurasi Deteksi Serangan Pada Jaringan Komputer Dengan Metode Naïve Bayes Dan Support Vector Machine (SVM)," *J. Pengemb. Teknol. Inf. dan Ilmu Komput.*, vol. 2, no. 9, pp. 3112–3123, 2018, [Online]. Available: <http://j-ptiik.ub.ac.id>
- [30] D. A. Nugroho, A. F. Rochim, and E. D. Widiyanto, "Perancangan dan Implementasi Intrusion Detection System di Jaringan Universitas Diponegoro," *J. Teknol. dan Sist.*

Komput., vol. 3, no. 2, p. 171, 2015, doi: 10.14710/jtsiskom.3.2.2015.171-178.

[31] I. Mukhopadhyay, M. Chakraborty, and S. Chakrabarti, “A Comparative Study of Related Technologies of Intrusion Detection & Prevention Systems,” J. Inf. Secur., vol. 02, no. 01, pp. 28–38, 2011, doi: 10.4236/jis.2011.21003.

## LAMPIRAN

### A. Kode Program dari Tampilan Monitor

```
<?php
    $host = 'localhost';
    $username = 'antonel';
    $password = '1234underworld1234';
    $database = 'ids';

    $conn = new mysqli($host, $username, $password, $database);

    $data1 = file_get_contents('alert');
    $lines1 = explode("\n", $data1);

    foreach ($lines1 as $line)
    {
        if (!empty($line))
        {
            $columns = explode(" ", $line);

            $timestamp = $columns[0];
            $id_rules = trim($columns[3], '[');
            $message = $columns[4];
            $classification = trim($columns[7], '[');
            $severity = trim($columns[9], '[');
            $protocol = trim($columns[10], '{');
            $src = explode(':', $columns[11]);
            $ip_src = $src[0];
            $port_src = $src[1];
            $dest = explode(':', $columns[13]);
            $ip_dest = $dest[0];
            $port_dest = $dest[1];

            $checkQuery = "SELECT * FROM snort WHERE
timestamp='$timestamp'";
            $checkResult = $conn->query($checkQuery);

            if ($checkResult->num_rows == 0)
            {
```



```

$query = "INSERT INTO snort (timestamp, id_rules,
message, tool, classification, severity, protocol, ip_src, port_src, ip_dest, port_dest,
status, status2)

```

```

VALUES
('$timestamp', '$id_rules', '$message', 'Snort', '$classification', '$severity',
'$protocol', '$ip_src', '$port_src', '$ip_dest', '$port_dest', 'Terdeteksi', '-')";

```

```

        if($conn->query($query) === TRUE)
        {
            $snortQuery = "SELECT * FROM suricata
WHERE timestamp='$timestamp'";
            $snortResult = $conn->query($snortQuery);
        }
        else
        {
        }
    }
    else
    {
    }
}
}

```

```

$data2 = file_get_contents('fast.log');
$lines2 = explode("\n", $data2);

```

```

foreach ($lines2 as $line)
{
    if (!empty($line))
    {
        $columns = explode(" ", $line);

        $timestamp = $columns[0];
        $id_rules = trim($columns[3], '[');
        $message = $columns[4];
        $classification = trim($columns[7], '[');
        $severity = trim($columns[9], '[');
        $protocol = trim($columns[10], '{ }');
    }
}

```

```

$src = explode(':', $columns[11]);
$ip_src = $src[0];
$port_src = $src[1];
$ddest = explode(':', $columns[13]);
$ip_dest = $ddest[0];
$port_dest = $ddest[1];

$checkQuery = "SELECT * FROM suricata WHERE
timestamp='$timestamp'";
$checkResult = $conn->query($checkQuery);

if($checkResult->num_rows == 0)
{
    $query = "INSERT INTO suricata (timestamp,
id_rules, message, tool, classification, severity, protocol, ip_src, port_src, ip_dest,
port_dest, status, status2)
VALUES
('$timestamp', '$id_rules', '$message', 'Suricata', '$classification', '$severity',
'$protocol', '$ip_src', '$port_src', '$ip_dest', '$port_dest', '-', 'Terdeteksi')";

    if($conn->query($query) === TRUE)
    {
        $snortQuery = "SELECT * FROM snort
WHERE timestamp='$timestamp'";
        $snortResult = $conn->query($snortQuery);

        if($snortResult->num_rows > 0)
        {
            $status2 = "Terdeteksi";

            $updateQuery = "UPDATE snort SET
status2='$status2' WHERE timestamp='$timestamp'";
            $conn->query($updateQuery);
        }
        else
        {
            $status2 = "-";
        }
        $updateQuery = "UPDATE suricata SET
status='$status2' WHERE timestamp='$timestamp'";

```

```

                                $conn->query($updateQuery);
                                }
                                else
                                {

                                }
                                }
                                else
                                {

                                }
                                }
                                }

$snortQuery = "SELECT * FROM snort";
$snortResult = $conn->query($snortQuery);

if ($snortResult->num_rows > 0)
{
    while ($row = $snortResult->fetch_assoc())
    {
        $timestamp = $row['timestamp'];
        $id_rules = $row['id_rules'];
        $message = $row['message'];
        $tool = $row['tool'];
        $classification = $row['classification'];
        $severity = $row['severity'];
        $protocol = $row['protocol'];
        $ip_src = $row['ip_src'];
        $port_src = $row['port_src'];
        $ip_dest = $row['ip_dest'];
        $port_dest = $row['port_dest'];
        $status = $row['status'];
        $status2 = $row['status2'];

        $checkQuery = "SELECT * FROM monitor WHERE
timestamp='$timestamp' AND tool='$tool'";
        $checkResult = $conn->query($checkQuery);

        if($checkResult->num_rows == 0)

```

```

        {
            $insertQuery = "INSERT INTO monitor (timestamp,
id_rules, message, tool, classification, severity, protocol, ip_src, port_src, ip_dest,
port_dest, status, status2)
                                VALUES
('$timestamp', '$id_rules', '$message', '$tool', '$classification', '$severity',
'$protocol', '$ip_src', '$port_src', '$ip_dest', '$port_dest', '$status', '$status2')";
            $conn->query($insertQuery);
        }
        else
        {
        }
    }
}

$suricataQuery = "SELECT * FROM suricata";
$suricataResult = $conn->query($suricataQuery);

if ($suricataResult->num_rows > 0)
{
    while ($row = $suricataResult->fetch_assoc())
    {
        $timestamp = $row['timestamp'];
        $id_rules = $row['id_rules'];
        $message = $row['message'];
        $tool = $row['tool'];
        $classification = $row['classification'];
        $severity = $row['severity'];
        $protocol = $row['protocol'];
        $ip_src = $row['ip_src'];
        $port_src = $row['port_src'];
        $ip_dest = $row['ip_dest'];
        $port_dest = $row['port_dest'];
        $status = $row['status'];
        $status2 = $row['status2'];

        $checkQuery = "SELECT * FROM monitor WHERE
timestamp='$timestamp' AND tool='$tool'";
        $checkResult = $conn->query($checkQuery);
    }
}

```

```

        if($checkResult->num_rows == 0)
        {
            $insertQuery = "INSERT INTO monitor (timestamp,
id_rules, message, tool, classification, severity, protocol, ip_src, port_src, ip_dest,
port_dest, status, status2)
                                VALUES
('$timestamp', '$id_rules', '$message', '$tool', '$classification', '$severity',
'$protocol', '$ip_src', '$port_src', '$ip_dest', '$port_dest', '$status', '$status2')";
            $conn->query($insertQuery);
        }
        else
        {
        }
    }
}
?>

```

```

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link rel="preconnect" href="https://fonts.googleapis.com">
    <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>
    <link
href="https://fonts.googleapis.com/css2?family=Open+Sans&display=swap"
rel="stylesheet">
    <link rel="preconnect" href="https://fonts.googleapis.com">
    <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>
    <link
href="https://fonts.googleapis.com/css2?family=Open+Sans:wght@700&display=s
wap" rel="stylesheet">
    <link rel="stylesheet" type="text/css" href="style.css">
    <script type="text/javascript" src="script.js"></script>
<title></title>
<style>
    *{

```

```

        margin: 0px;
        padding: 0px;
        font-family: 'Open Sans', sans-serif;
        text-decoration: none;
    }

    header{
        text-align: center;
        font-size: 30px;
        font-weight: bold;
        margin: 30px
    }

    .navbar{
        display: flex;
        justify-content: center;
        background-color: #36304A;
    }

    .navbar > a{
        color: white;
        padding: 25px;
        font-size: 15px;
        font-weight: bold;
        transition: 1s;
    }

    .navbar > a:hover{
        background-color: white;
        color: black;
    }

    .monitor{
        display: flex;
        justify-content: center;
        margin: 50px;
    }

    table{
        overflow: auto;

```

```

        max-height: 370px;
    }

    table, th, td{
        border-collapse: collapse;
        font-size: 13px;
        height: 60px;
    }

    th, td{
        width: 180px;
        text-align: center;
    }

    th{
        background-color: #36304A;
        color: white;
        padding: 5px;
        letter-spacing: 1px;
    }

    tr:last-child td:last-child{
        border-bottom-right-radius: 15px;
    }

    tr:last-child td:first-child{
        border-bottom-left-radius: 15px;
    }

    th:first-child{
        border-top-left-radius: 15px;
    }

    th:last-child{
        border-top-right-radius: 15px;
    }

    tr:nth-child(odd){
        background-color: #F5F5F5;
    }

```

```

.par-add{
    display: flex;
    justify-content: space-between;
    margin-left: 65px;
    margin-right: 65px;
    margin-bottom: 50px;
}

.additional, .additional2{
    margin-top: 50px;
    background-color: #9a97a4;
    padding: 15px;
    width: 210px;
    color: white;
    font-weight: 600;
    font-size: 13px;
    letter-spacing: 1px;
    border-radius: 4px;
    word-spacing: 3px;
    border-left: 3px solid #36304A;
}

.additional2{
    display: flex;
    align-items: center;
}

.anima{
    display: flex;
    justify-content: space-between;
    align-items: center;
    overflow: hidden;
    width: 93px;
    transition: 1s;
}

.anima > a, .anima > p{
    color: white;
    padding: 25px;
}

```



```

        font-size: 15px;
        font-weight: bold;
        transition: 1s;
    }

    .anima:hover{
        width: 293px;
        transition: 1s;
    }

    .anima > p:hover, .anima > a:hover{
        background-color: white;
        color: black;
        transition: 1s;
    }

    .titleTable{
        text-align: center;
        font-size: 30px;
        font-weight: bold;
        margin: 50px;
    }
</style>
</head>
<body>
    <header>
        <p>Monitoring Snort & Suricata</p>
    </header>

    <main>
        <div class="navbar">
            <a href="http://localhost/ids/home.php">Home</a>
            <a href="http://localhost/ids/snort_alert.php">Monitor</a>
            <div class="anima">
                <p>Rules</p>
                <a
href="http://localhost/ids/rules_snort.php">Snort</a>
                <a
href="http://localhost/snort/rules_suricata.php">Suricata</a>
            </div>

```

```

</div>

<div class="par-add">
    <div class="additional">
        <?php
            date_default_timezone_set('Asia/Jakarta');

            $zona_waktu = date_default_timezone_get();
            $tanggal = date('d');
            $bulan = date('m');
            $tahun = date('Y');

            echo "<p>" . $zona_waktu . " </p>";
            echo "<p>" . $tahun . " - " . $bulan . " - " .
$tanggal . " </p>";
        ?>
        <p id="jam"></p>
    </div>

    <div class="additional2">
        <?php
            $query = "SELECT COUNT(*) AS
total_rows FROM snort";

            $result = mysqli_query($conn, $query);

            if($result)
            {
                $rows = mysqli_fetch_assoc($result);
                $total_rows = $rows['total_rows'];

                echo "<p>". "Jumlah Alerts Snort : " .
$total_rows . " </p>";
            }
        ?>
    </div>

    <div class="additional2">
        <?php
            $query = "SELECT COUNT(*) AS
total_rows FROM suricata";

```

```

        $result = mysqli_query($conn, $query);

        if($result)
        {
            $rows = mysqli_fetch_assoc($result);
            $total_rows = $rows['total_rows'];

            echo "<p>". "Jumlah Alerts Suricata : "
" . $total_rows . "</p>";
        }
    ?>
</div>

<div class="additional2">
    <?php
        $query = "SELECT COUNT(*) AS
total_rows FROM monitor";
        $result = mysqli_query($conn, $query);

        if($result)
        {
            $rows = mysqli_fetch_assoc($result);
            $total_rows = $rows['total_rows'];

            echo "<p>". "Jumlah Semua Alerts : "
. $total_rows . "</p>";
        }
    ?>
</div>

</div>

<div class="monitor" style="overflow: auto; max-height: 1000px;">

    <?php
        $query = "SELECT * FROM monitor";

        $result = $conn->query($query);

        if ($result->num_rows > 0)

```

```

{
    echo "<table id='iniTable'>
        <tr>
            <th>Timestamp</th>
            <th>Message</th>
            <th>Tool</th>
            <th>Classification</th>
            <th>Severity</th>
            <th>Source IP</th>
            <th>Source Port</th>
            <th>Destination
IP</th>
            <th>Destination
Port</th>
            <th>Snort</th>
            <th>Suricata</th>
        </tr>";

    while($row = $result->fetch_assoc())
    {
        echo "<tr>
            <td>" .
$row['timestamp'] . "</td>
            <td>" . $row['message']
. "</td>
            <td>" . $row['tool'] .
            <td>" .
$row['classification'] . "</td>
            <td>" . $row['severity']
. "</td>
            <td>" . $row['ip_src'] .
            <td>" . $row['port_src']
. "</td>
            <td>" . $row['ip_dest'] .
            <td>" .
$row['port_dest'] . "</td>

```

```

        <td>" . $row['status'] .
    "</td>
        <td>" . $row['status2'] .
    "</td>
        </tr>";
    }
    echo "</table>";
}
?>
</div>
</main>

<script>
let table1 = document.getElementById("iniTable");

applyColorToTable(table1);

function applyColorToTable(table)
{
    let rows = table.getElementsByTagName("tr");

    for (let i = 0; i < rows.length; i++)
    {
        let cols = rows[i].getElementsByTagName("td");

        for (let j = 0; j < cols.length; j++)
        {
            if (cols[j].innerText === '0')
            {
                cols[j].style.backgroundColor =
"#caf0f8";
            }
            else if (cols[j].innerText === '1')
            {
                cols[j].style.backgroundColor =
"#C7E8CA";
            }
            else if (cols[j].innerText === '2')
            {

```

```

                                cols[j].style.backgroundColor =
"#F3E99F";
                                }
                                else if (cols[j].innerText === '3')
                                {
                                    cols[j].style.backgroundColor =
"#ffbaba";
                                }
                            }
                        }
                    }
                }
            }
        }
    }
</script>

<script>
window.onload = function() { jam(); }

function jam()
{
    var e = document.getElementById('jam'),
    d = new Date(), h, m, s;
    h = d.getHours();
    m = set(d.getMinutes());
    s = set(d.getSeconds());

    e.innerHTML = h + ' : ' + m + ' : ' + s;

    setTimeout('jam()', 1000);
}

function set(e)
{
    e = e < 10 ? '0' + e : e;
    return e;
}
</script>
</body>
</html>

```

## B. Kode Program dari Tampilan Rules Snort

```
<?php
    $host = 'localhost';
    $username = 'antone1';
    $password = '1234underworld1234';
    $database = 'ids';

    $conn = new mysqli($host, $username, $password, $database);

    $data = file_get_contents('rulesnort.txt');

    $lines = explode("\n", $data);

    foreach($lines as $line)
    {
        if(!empty($line))
        {
            if($line[0] !== '#')
            {
                $columns = explode(" ", $line);

                $action = $columns[0];
                $protocol = $columns[1];
                $ip_src = $columns[2];
                $port_src = $columns[3];
                $direction = $columns[4];
                $ip_dest = $columns[5];
                $port_dest = $columns[6];

                $msg = explode("'", $columns[7]);
                $message = trim($msg[1], ';');

                $ctp = explode(':', $columns[8]);
                $classtype = trim($ctp[1], ';');

                $sids = explode(':', $columns[9]);
                $sids2 = explode(';', $sids[1]);
                $sid_rules = $sids2[0];
            }
        }
    }
}
```

```

                                $checkQuery = "SELECT idrules FROM snortrules
WHERE idrules = '$id_rules'";
                                $result = $conn->query($checkQuery);

                                if($result->num_rows == 0)
                                {
                                    $query = "INSERT INTO snortrules (action,
protocol, ip_src, port_src, direction, ip_dest, port_dest, message, classtype, idrules)
                                VALUES ('$action', '$protocol', '$ip_src',
                                '$port_src', '$direction', '$ip_dest', '$port_dest', '$message', '$classtype',
                                '$id_rules')";

                                    if($conn->query($query) === TRUE)
                                    {
                                        }
                                    else
                                    {
                                        }
                                    }
                                else
                                {
                                    }
                                }
                            }
                        }
                    }
                }
            }
        }
    }
?>

```

```

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link rel="preconnect" href="https://fonts.googleapis.com">
    <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>

```



```

<link
href="https://fonts.googleapis.com/css2?family=Open+Sans&display=swap"
rel="stylesheet">
<link rel="preconnect" href="https://fonts.googleapis.com">
<link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>
<link
href="https://fonts.googleapis.com/css2?family=Open+Sans:wght@700&display=s
wap" rel="stylesheet">
<link rel="stylesheet" type="text/css" href="style.css">
<script type="text/javascript" src="script.js"></script>
<title>Rules | Snort</title>
<style>
    *{
        margin: 0px;
        padding: 0px;
        font-family: 'Open Sans', sans-serif;
        text-decoration: none;
    }

    header{
        text-align: center;
        font-size: 30px;
        font-weight: bold;
        margin: 30px
    }

    .navbar{
        display: flex;
        justify-content: center;
        background-color: #36304A;
    }

    .navbar > a{
        color: white;
        padding: 25px;
        font-size: 15px;
        font-weight: bold;
        transition: 1s;
    }

```

```

.navbar > a:hover{
    background-color: white;
    color: black;
}

.monitor{
    display: flex;
    justify-content: center;
    margin: 50px;
}

table, th, td{
    border-collapse: collapse;
    font-size: 13px;
    height: 60px;
}

th, td{
    width: 180px;
    text-align: center;
}

th{
    background-color: #36304A;
    color: white;
    padding: 5px;
    letter-spacing: 1px;
}

tr:last-child td:last-child{
    border-bottom-right-radius: 15px;
}

tr:last-child td:first-child{
    border-bottom-left-radius: 15px;
}

th:first-child{
    border-top-left-radius: 15px;
}

```

```

th:last-child{
    border-top-right-radius: 15px;
}

tr:nth-child(odd){
    background-color: #F5F5F5;
}

.par-add{
    display: flex;
    justify-content: space-evenly;
    margin-left: 65px;
    margin-right: 65px;
}

.additional, .additional2, .additional3{
    margin-top: 50px;
    background-color: #9a97a4;
    padding: 15px;
    width: 200px;
    color: white;
    font-weight: 600;
    font-size: 13px;
    letter-spacing: 1px;
    border-radius: 4px;
    word-spacing: 3px;
    border-left: 3px solid #36304A;
}

.additional2{
    display: flex;
    align-items: center;
}

.anima{
    display: flex;
    justify-content: space-between;
    align-items: center;
    overflow: hidden;

```

```

        width: 93px;
        transition: 1s;
    }

    .anima > a, .anima > p{
        color: white;
        padding: 25px;
        font-size: 15px;
        font-weight: bold;
        transition: 1s;
    }

    .anima:hover{
        width: 293px;
        transition: 1s;
    }

    .anima > p:hover, .anima > a:hover{
        background-color: white;
        color: black;
        transition: 1s;
    }

    .abc{
        background-color: blanchedalmond;
        color: aqua;
    }
</style>
</head>
<body>
    <header>
        <p>Snort Rules</p>
    </header>

    <main>
        <div class="navbar">
            <a href="http://localhost/ids/home.php">Home</a>
            <a href="http://localhost/ids/snort_alert.php">Snort</a>
            <a href="http://localhost/ids/suricata_alert.php">Suricata</a>
            <div class="anima">

```

```

        <p>Rules</p>
        <a
href="http://localhost/ids/rules_snort.php">Snort</a>
        <a
href="http://localhost/ids/rules_suricata.php">Suricata</a>
    </div>
</div>

<div class="par-add">
    <div class="additional">
        <?php
            date_default_timezone_set('Asia/Jakarta');

            $zona_waktu = date_default_timezone_get();
            $tanggal = date('d');
            $bulan = date('m');
            $tahun = date('Y');

            echo "<p>" . $zona_waktu . " </p>";
            echo "<p>" . $tahun . " - " . $bulan . " - " .
$tanggal . " </p>";
        ?>
        <p id="jam"></p>
    </div>

    <div class="additional2">
        <?php
            $query = "SELECT COUNT(*) AS
total_rows FROM snortrules";

            $result = mysqli_query($conn, $query);

            if($result)
            {
                $rows = mysqli_fetch_assoc($result);
                $total_rows = $rows['total_rows'];

                echo "<p>". "Jumlah Rules : " .
$total_rows . " </p>";
            }
        ?>
    </div>

```

```

        </div>
    </div>

    <div class="monitor">
        <?php
            $query = "SELECT * FROM snortrules";

            $result = $conn->query($query);

            if ($result->num_rows > 0)
            {
                echo "<table id='iniTable'>
                    <tr>
                        <th>Action</th>
                        <th>Protocol</th>
                        <th>Source IP</th>
                        <th>Source Port</th>
                        <th>Direction</th>
                        <th>Destination
IP</th>
                        <th>Destination
Port</th>
                        <th>Message</th>
                        <th>Classtype</th>
                        <th>ID Rules</th>
                    </tr>";

                while($row = $result->fetch_assoc())
                {
                    echo "<tr>
                        <td>" . $row['action'] .
                        <td>" . $row['protocol']
                        <td>" . $row['ip_src'] .
                        <td>" . $row['port_src']
                        <td>" .
                        $row['direction'] . "</td>

```

```

" </td>
$row['port_dest'] . " </td>
. " </td>
$row['classtype'] . " </td>
" </td>

</tr>";
    }
    echo "</table>";
}
$conn->close();
?>

</div>
</main>

<script>
window.onload = function() { jam(); }

function jam()
{
    var e = document.getElementById('jam'),
        d = new Date(), h, m, s;
    h = d.getHours();
    m = set(d.getMinutes());
    s = set(d.getSeconds());

    e.innerHTML = h + ' : ' + m + ' : ' + s;

    setTimeout('jam()', 1000);
}

function set(e)
{
    e = e < 10 ? '0' + e : e;
    return e;
}

```

```
</script>
</body>
</html>
```

### C. Kode Program dari Tampilan Rules Suricata

```
<?php
    $host = 'localhost';
    $username = '#namausername';
    $password = '1234underworld1234';
    $database = 'ids';

    $conn = new mysqli($host, $username, $password, $database);

    $data = file_get_contents('rulesuricata.txt');

    $lines = explode("\n", $data);

    foreach($lines as $line)
    {
        if(!empty($line))
        {
            if($line[0] !== '#')
            {
                $columns = explode(" ", $line);

                $action = $columns[0];
                $protocol = $columns[1];
                $ip_src = $columns[2];
                $port_src = $columns[3];
                $direction = $columns[4];
                $ip_dest = $columns[5];
                $port_dest = $columns[6];

                $msg = explode("'", $columns[7]);
                $message = trim($msg[1], ';');

                $ctp = explode(':', $columns[8]);
                $classtype = trim($ctp[1], ';');
```



```

        $ids = explode(':', $columns[9]);
        $ids2 = explode(';', $ids[1]);
        $id_rules = $ids2[0];

        $checkQuery = "SELECT idrules FROM snortrules
WHERE idrules = '$id_rules'";
        $result = $conn->query($checkQuery);

        if($result->num_rows == 0)
        {
            $query = "INSERT INTO snortrules (action,
protocol, ip_src, port_src, direction, ip_dest, port_dest, message, classtype, idrules)
VALUES ('$action', '$protocol', '$ip_src',
'$port_src', '$direction', '$ip_dest', '$port_dest', '$message', '$classtype',
'$id_rules')";

            if($conn->query($query) === TRUE)
            {
            }
            else
            {
            }
        }
        else
        {
        }
    }
}
}
?>

```

```

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">

```

```

<link rel="preconnect" href="https://fonts.googleapis.com">
<link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>
<link
href="https://fonts.googleapis.com/css2?family=Open+Sans&display=swap"
rel="stylesheet">
<link rel="preconnect" href="https://fonts.googleapis.com">
<link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>
<link
href="https://fonts.googleapis.com/css2?family=Open+Sans:wght@700&display=s
wap" rel="stylesheet">
<link rel="stylesheet" type="text/css" href="style.css">
<script type="text/javascript" src="script.js"></script>
<title>Rules | Suricata</title>
<style>
*{
    margin: 0px;
    padding: 0px;
    font-family: 'Open Sans', sans-serif;
    text-decoration: none;
}

header{
    text-align: center;
    font-size: 30px;
    font-weight: bold;
    margin: 30px
}

.navbar{
    display: flex;
    justify-content: center;
    background-color: #36304A;
}

.navbar > a{
    color: white;
    padding: 25px;
    font-size: 15px;
    font-weight: bold;
    transition: 1s;

```

```

}

.navbar > a:hover{
    background-color: white;
    color: black;
}

.monitor{
    display: flex;
    justify-content: center;
    margin: 50px;
}

table, th, td{
    border-collapse: collapse;
    font-size: 13px;
    height: 60px;
}

th, td{
    width: 180px;
    text-align: center;
}

th{
    background-color: #36304A;
    color: white;
    padding: 5px;
    letter-spacing: 1px;
}

tr:last-child td:last-child{
    border-bottom-right-radius: 15px;
}

tr:last-child td:first-child{
    border-bottom-left-radius: 15px;
}

th:first-child{

```

```

        border-top-left-radius: 15px;
    }

    th:last-child{
        border-top-right-radius: 15px;
    }

    tr:nth-child(odd){
        background-color: #F5F5F5;
    }

    .par-add{
        display: flex;
        justify-content: space-evenly;
        margin-left: 65px;
        margin-right: 65px;
    }

    .additional, .additional2, .additional3{
        margin-top: 50px;
        background-color: #9a97a4;
        padding: 15px;
        width: 200px;
        color: white;
        font-weight: 600;
        font-size: 13px;
        letter-spacing: 1px;
        border-radius: 4px;
        word-spacing: 3px;
        border-left: 3px solid #36304A;
    }

    .additional2{
        display: flex;
        align-items: center;
    }

    .anima{
        display: flex;
        justify-content: space-between;
    }

```

```

        align-items: center;
        overflow: hidden;
        width: 93px;
        transition: 1s;
    }

    .anima > a, .anima > p{
        color: white;
        padding: 25px;
        font-size: 15px;
        font-weight: bold;
        transition: 1s;
    }

    .anima:hover{
        width: 293px;
        transition: 1s;
    }

    .anima > p:hover, .anima > a:hover{
        background-color: white;
        color: black;
        transition: 1s;
    }

    .abc{
        background-color: blanchedalmond;
        color: aqua;
    }
</style>
</head>
<body>
    <header>
        <p>Suricata Rules</p>
    </header>

    <main>
        <div class="navbar">
            <a href="http://localhost/ids/home.php">Home</a>
            <a href="http://localhost/ids/snort_alert.php">Snort</a>

```

```

        <a href="http://localhost/ids/suricata_alert.php">Suricata</a>
        <div class="anima">
            <p>Rules</p>
            <a
href="http://localhost/ids/rules_snort.php">Snort</a>
            <a
href="http://localhost/ids/rules_suricata.php">Suricata</a>
        </div>
    </div>

    <div class="par-add">
        <div class="additional">
            <?php
                date_default_timezone_set('Asia/Jakarta');

                $zona_waktu = date_default_timezone_get();
                $tanggal = date('d');
                $bulan = date('m');
                $tahun = date('Y');

                echo "<p>" . $zona_waktu . " </p>";
                echo "<p>" . $tahun . " - " . $bulan . " - " .
$tanggal . " </p>";
            ?>
            <p id="jam"></p>
        </div>

        <div class="additional2">
            <?php
                $query = "SELECT COUNT(*) AS
total_rows FROM snortrules";
                $result = mysqli_query($conn, $query);

                if($result)
                {
                    $rows = mysqli_fetch_assoc($result);
                    $total_rows = $rows['total_rows'];

                    echo "<p>". "Jumlah Rules : " .
$total_rows . " </p>";
                }
            }
        </div>
    </div>

```

```

    }
    ?>
</div>
</div>

<div class="monitor">
    <?php
        $query = "SELECT * FROM snortrules";

        $result = $conn->query($query);

        if ($result->num_rows > 0)
        {
            echo "<table id='iniTable'>
                <tr>
                    <th>Action</th>
                    <th>Protocol</th>
                    <th>Source IP</th>
                    <th>Source Port</th>
                    <th>Direction</th>
                    <th>Destination
IP</th>
                    <th>Destination
Port</th>
                    <th>Message</th>
                    <th>Classtype</th>
                    <th>ID Rules</th>
                </tr>";

            while($row = $result->fetch_assoc())
            {
                echo "<tr>
                    <td>" . $row['action'] .
                    <td>" . $row['protocol']
                    <td>" . $row['ip_src'] .
                    <td>" . $row['port_src']
                    <td>" . $row['message'] .
                    <td>" . $row['classtype'] .
                    <td>" . $row['id_rules'] .
                </td>";
            }
        }
    }
}
?>
</div>
</div>

```

```

$row['direction'] . "</td>
" </td>
$row['port_dest'] . "</td>
. "</td>
$row['classtype'] . "</td>
" </td>

</tr>";
}
echo "</table>";
}
$conn->close();
?>

</div>
</main>

<script>
window.onload = function() { jam(); }

function jam()
{
    var e = document.getElementById('jam'),
    d = new Date(), h, m, s;
    h = d.getHours();
    m = set(d.getMinutes());
    s = set(d.getSeconds());

    e.innerHTML = h + ' : ' + m + ' : ' + s;

    setTimeout('jam()', 1000);
}

function set(e)
{
    e = e < 10 ? '0' + e : e;

```



```
        return e;
    }
</script>
</body>
</html>
```