Perbandingan Nilai Akurasi Snort dan Suricata dalam Mendeteksi Intrusi Lalu Lintas di Jaringan

Adam Dwi Ralianto[1)], Setiyo Cahyono[2)]
(1) Keamanan Siber, Politeknik Siber dan Sandi Negara/adam.dwi@bssn.go.id
(2) Keamanan Siber, Politeknik Siber dan Sandi Negara/setiyo.cahyono@poltekssn.ac.id

Abstrak

Seiring bertambahnya pengguna internet, semakin canggih juga serangan siber yang terjadi. Berdasarkan laporan tahunan dari Honeynet Project BSSN, tahun 2018 telah terjadi 12.895.554 serangan yang masuk ke Indonesia dan 513.863 berupa aplikasi berbahaya. Serangan-serangan ini apabila tidak terdeteksi dan dicegah, maka dapat menurunkan kredibilitas layanan, seperti kerahasiaan data, integritas, dan ketersediaan data. Sehingga dibutuhkan aplikasi yang mampu mendeteksi banyaknya serangan tersebut, yaitu Instrusion Detection System (IDS). Terdapat beberapa aplikasi IDS yang ada, seperti Snort dan Suricata. Dari banyak aplikasi yang ada, perlu dilakukan analisis terhadap kemampuannya dalam mendeteksi intrusi di jaringan. Salah satu kemampuan yang harus dianalisis yaitu akurasi. Akurasi adalah sebuah metrik yang mengukur seberapa benar IDS bekerja dengan mengukur persentase deteksi dan kegagalan serta jumlah peringatan palsu yang dihasilkan suatu sistem. Akurasi dalam mendeteksi serangan-serangan ini menjadi tantangan untuk aplikasi IDS. Dalam melakukan analisis diawali dengan melakukan pengujian dengan menggunakan Pytbull terhadap aplikasi Snort dan Suricata. Pytbull dikonfigurasi dengan 70 serangan yang dikelompokkan dalam 11 modul serangan. Pengujian dilakukan dalam 3 skenario, yaitu menggunakan rules asli, rules dari Emerging Threat, dan rules yang dibuat sendiri. Penelitian ini, memberikan penjelasan terkait bagaimana melakukan pengujian menggunakan Pytbull terhadap Snort dan Suricata menggunakan 3 skenario yang telah ditentukan yang kemudian dilanjutkan analisis dengan menghitung nilai akurasinya untuk dibandingkan mana yang lebih baik. Dari penelitian ini didapatkan hasil bahwa Suricata versi 5.0.2 dengan pengujian menggunakan Pytbull dalam 3 skenario, memiliki akurasi lebih tinggi daripada Snort versi 2.9.15.1 karena memiliki rules yang lebih banyak. Walaupun rules lebih banyak, namun penggunaan memory Suricata lebih stabil karena menggunakan fitur multi-threading yang dimilikinya.

Kata kunci: Perbandingan nilai akurasi, Pytbull, Snort, Suricata

1. PENDAHULUAN

Internet merupakan jaringan komunikasi elektronik yang menghubungkan jaringan komputer dan fasilitas komputer yang terorganisasi di seluruh dunia melalui telepon atau satelit [1]. Menurut Berners Lee, internet adalah suatu jaringan yang terdiri dari beberapa jaringan, dimana suatu jaringan lokal juga bisa terhubung ke dalam suatu jaringan lainnya [2]. Pengguna internet dari tahun ke tahun mengalami peningkatan. Internet World Stats pada 30 Juni 2019 memaparkan bahwa terdapat 4.536.248.808 pengguna internet di dunia [3]. Semakin meningkatnya penggunaan internet, maka akan semakin banyak kerawanan yang ada di dunia internet. Honeynet Project BSSN menyebutkan dalam laporan tahunannya di 2018, terdapat 12.895.554 serangan yang masuk ke Indonesia, dan 513.863 berupa aplikasi berbahaya [4].

Semakin canggihnya serangan siber di dunia menjadi tantangan agar dapat mendeteksi serangan-serangan tersebut secara lebih akurat [5]. Hal ini perlu dilakukan karena apabila serangan ini tidak dapat dicegah dapat menurunkan kredibilitas layanan, seperti kerahasiaan data, integritas, dan ketersediaan data [5]. Oleh karena itu, dibutuhkan sistem yang dapat mendeteksi banyaknya serangan tersebut, yaitu *Intrusion Detection System* (IDS). IDS merupakan mekanisme dinamis untuk menganalisis jaringan [6]. IDS memantau proses melalui sistem komputer atau jaringan dan menganalisanya untuk medeteksi intrusi lalu lintas yang melanggar kebijakan keamanan komputer [6]. IDS memberikan peran penting dalam mengamankan jaringan [7]. Saat ini, banyak sekali aplikasi IDS seperti disebutkan oleh Comparitech.net, seperti Snort dan Suricata [8].

Performa IDS dalam mendeteksi intrusi lalu lintas di jaringan membuat pengujian semakin menantang dan dibutuhkan [9]. Salah satu performanya yaitu akurasi. Akurasi adalah sebuah metrik yang digunakan untuk mengukur seberapa benar IDS bekerja dengan mengukur persentase deteksi dan kegagalan serta jumlah peringatan palsu yang dihasilkan suatu sistem [10]. Pytbull adalah *framework* yang menguji performa dari aplikasi IDS untuk dapat dibandingan [11]. Terdapat kurang lebih 300 uji dalam *framework* Pytbull yang dikelompokkan dalam 11 modul pengujian.

Akurasi deteksi aplikasi IDS *signature-based* bergantung pada *rules* yang digunakan [12]. Dalam banyak kesempatan, banyak orang menggunakan *rules* yang ada [13]. Nama lain dari *rules* adalah *signature*. *Rules*/*Signature* ini memainkan peran penting dalam penggunaan aplikasi IDS [14]. Di dalam dokumentasinya, *rules* yang digunakan aplikasi IDS dapat berasal dari *rules* asli dari aplikasi IDS, open source *rules* (https://rules.emergingthreat.net), dan *rules* yang ditentukan sendiri dengan mengikuti

format yang ada [13][14]. *Rules* asli dari Snort didapatkan dari *rules* komunitas pada https://www.snort.org, sedangkan Suricata didapatkan dengan menjalankan perintah `suricata-update`.

Pada penelitian ini dilakukan pengimplementasian 2 aplikasi IDS, yaitu Snort dan Suricata untuk dilakukan pengujian dengan menggunakan *framework* Pytbull. Dalam pengujiannya, Snort dan Suricata menggunakan tiga skenario implementasi, yaitu menggunakan *rules* asli, *open source rules*, dan *rules* yang ditentukan sendiri. Hasil pengujian yang didapatkan selanjutnya dianalisis dengan menghitung nilai akurasinya antara satu aplikasi dengan yang lain. Pengujian dilakukan dengan menggunakan 70 uji dalam *framework* Pytbull yang dikelompokkan dalam 11 modul pengujian. Sebelas modul pengujian ini nantinya menjadi parameter analisis perbandingan nilai akurasi antara aplikasi IDS satu dengan yang lainnya. 70 uji diambil dari 300 uji dari *framework* Pytbull karena beberapa uji lainnya merupakan uji ipReputation yang menggunakan ratusan *Internet Protocol* (IP) yang berbahaya, sehingga hanya diambil beberapa.

2. LANDASAN TEORI

Bab ini akan menjelaskan mengenai tinjauan pustaka dalam penelitian yang dilakukan. Pada subbab 2.1 dijelaskan terkait apa itu IDS yang menjadi objek utama dalam penelitian. Subbab 2.2 dijelaskan terkait *framework* Pytbull yang digunakan untuk melakukan serangan sebagai pengujian. Subbab 2.3 dijelaskan terkait akurasi, sedangkan pada Subbab 2.4 dijelaskan mengenai studi literatur terkait penelitian yang sebelumnya pernah dilakukan.

2.1. *Intrusion Detection System* (IDS)

Intrusion Detection adalah proses pemantauan *events* yang terjadi dalam sistem komputer atau jaringan dan menganalisanya untuk melihat tanda kemungkinan terjadinya insiden, yang merupakan pelanggaran kebijakan keamanan komputer, kebijakan penggunaan, atau praktik keamanan standar. Insiden memiliki banyak penyebab, seperti *malware*, penyalahgunaan *privileges* atau percobaan untuk mendapatkan *privileges*. Perangkat lunak untuk melakukan otomatisasi proses *Instrusion Detection* disebut dengan *Intrusin Detection System* (IDS) [15]. Contoh Aplikasi IDS untuk jaringan adalah Snort dan Suricata.

Snort adalah *Network Intrusion Detection and Detection System* (NIDPS) bersifat *free* dan *open source* yang dibuat oleh Martin Roesch (Sourcefire) pada tahun 1998 [16]. Saat ini, Snort dikembangkan oleh Cisco dengan *Single-threaded*. Snort memiliki kemampuan untuk melakukan analisis lalu lintas *real-time* dan *packet logging* di jaringan IP [16]. Snort melakukan analisis protokol, pencarian dan pencocokan konten. Konfigurasi Snort terdiri dari 3 mode, yaitu sebagai *Sniffer*, *Packet Logger*, dan *Network Intrusion Detection*. Dalam mode *Network Instruion Detetction*, Snort akan memantau lalu lintas jaringan dan menganalisisnya terhadap *rules* yang diterapkan [14].

Suricata adalah mesin pendeteksi ancaman jaringan yang *free* dan *open source*, teruji, cepat, dan handal yang dikembangkan tahun 2009 oleh Open Information Security Foundation (OISF) dengan fitur *Multi-threaded* [17]. Suricata mampu melakukan deteksi intrusi secara *real-time*, pencegahan intrusi *inline*, pemantauan keamanan jaringan, dan pemrosesan pcap *offline*. Suricata memeriksa lalu lintas jaringan menggunakan *rules* dan mendukung skrip Lua yang kuat untuk medeteksi ancaman yang kompleks [13][17].

Pada penelitian ini, digunakan 2 buah aplikasi IDS yang bersifat *open source*, berjalan pada *platform* Linux, dan menggunakan *Signature-based* (berbasis *rules*), yaitu Snort versi 2.9.15.1 dan Suricata versi 5.0.2. Dalam instalasinya, Snort dan Suricata mengikuti langkah-langkah yang didokumentasikan oleh *blog* rapid7 [18][19].

2.2. *Framework* Pytbull

Pytbull adalah *framework* untuk melakukan pengujian *Intrusion Detection/Prevention System* (IDS/IPS) untuk Snort, Suricata, dan lainnya yang menghasilkan *alert*. *Framework* ini juga dapat digunakan untuk membandingkan kemampuan IDS/IPS dalam mendeteksi dan memblokir, serta membandingkan modifikasi konfigurasi, juga memerika/memvalidasi konfigurasi. *Framework* ini memiliki kurang lebih 300 tes yang dikelompokkan dalam 11 modul pengujian, yaitu [11]:

a. badTraffic : Paket yang tidak sesuai RFC dikirim ke server untuk menguji bagaimana paket diproses.
b. bruteForce : menguji kemampuan server untuk melacak serangan *brute force* (contoh: FTP).
c. clientSideAttacks : menggunakan *reverse shell* untuk memberikan intruksi mengunduh *remote malicious file* kepada server.
d. denialOfService : menguji kemampuan untuk melindungi terhadap upara *Denial of Service* (DoS).
e. evasionTechniques : berbagai teknik *evasion* yang dijalankan.
f. fragmentedPackets : berbagai *payload* yang terfragmentasi dikirim ke server untuk menguji kemampuan menyusun ulang paket dan mendeteksi serangannya.
g. ipReputation : menguji kemampuan server untuk mendeteksi lalu lintas dari/ke server dengan reputasi rendah.
h. normalUsage : *payload* yang sesuai dengan penggunaan normal.
i. pcapReplay : memungkinkan untuk melakukan *replay* file pcap.

j. shellCodes : mengirim berbagai *shellcodes* ke server melalui port 21/tcp untuk menguji kemampuan server mendeteksi/menolak *shellcodes* tersebut.
k. test*Rules* : melakukan uji aturan dasar (*basic rules*) dengan melakukan serangan-serangan yang seharusnya terdeteksi oleh IDS/IPS.

2.3. Akurasi

Akurasi adalah sebuah metrik yang mengukur seberapa benar IDS bekerja dengan mengukur persentase deteksi dan kegagalan serta jumlah peringatan palsu yang dihasilkan suatu sistem. Penilaian dilakukan dengan menghitung berapa jumlah *True Positive* (TP), *False Positive* (FP), dan *False Negative* (FN). TP merupakan kondisi di mana ketika aplikasi IDS mendeteksi adanya serangan, dan memunculkan alert dari serangan tersebut, FN adalah kondisi di mana aplikasi IDS mendeteksi suatu serangan namun tidak memunculkan alert, sedangkan FP adalah kondisi di mana aplikasi IDS tidak mendeteksi adanya serangan namun menghasilkan alert. TN dalam hal ini dinilai 0, karena semua pengujian merupakan serangan. Hasil perhitungan TP, FN, FP tersebut kemudian menjadi nilai input untuk menghitung nilai akurasi dengan rumus TP+TN/(TP+FN+FP+TN). Gambar 1 menunjukkan visualisasi dari TP, FN, FP terhadap *attack* dan *alert*. *Alert* adalah pesan peringatan indikasi serangan yang dihasilkan oleh IDS. *Attack* adalah kejadian (*event*) yang mencurigakan [10].

Gambar 1. Gambaran Topologi dalam VirtualBox
(Sumber : Diolah sendiri)

2.4. Tinjauan Literatur

Penelitian ini merujuk pada 3 literatur sebelumnya yang terkait aplikasi Snort dan Suricata. Tiga rujukan literatur tersebut melakukan penelitian terkait perbandingan performa dari aplikasi Snort dan Suricata [7][9][20]. Namun, 3 rujukan literatur tersebut tidak melakukan penilaian akurasi, sehingga pada penelitian ini dilakukan perbandingan Snort dan Suricata dengan melakukan penilaian akurasi dari deteksinya.

3. METODE PENELITIAN

Metodologi penelitian yang digunakan dalam penelitian ini adalah *Benchmarking Methodology for Network Security Device Performance draft-ietf-bmwg-ngfw-performance-01*. Metodologi ini merupakan metodologi untuk membandingkan dan melakukan pengujian penggunaan aplikasi *layer* 7 [21]. Aplikasi *layer* 7 yang akan diuji dan dibandingkan pada penelitian ini adalah aplikasi IDS yang sudah ditentukan. Terdapat 4 tahapan dalam metodologi ini, yaitu *Objective*, *Test Setup*, *Test Parameters*, dan *Test Procedures and Expected Result.*

Pada tahap *Objective*, peneliti akan membuat skenario pengujian aplikasi IDS (Snort dan Suricata) yang akan dibandingkan menggunakan *framework* Pytbull berdasarkan literatur yang sudah dipelajari. Skenario yang dibuat yaitu aplikasi IDS yang diinstal akan memantau seluruh lalu lintas jaringan yang dilindungi. Kemudian uji performa setiap aplikasi IDS dilakukan dengan menjalankan Pytbull berdasarkan 3 skenario yang telah ditentukan. Setelah uji selesai, Pytbull akan memberikan laporan hasil.

Pada tahap *Test Setup*, peneliti melakukan instalasi dan konfigurasi lingkungan uji (sever, aplikasi IDS) yang digunakan untuk pengujian aplikasi IDS. Instalasi dan konfigurasi harus dilakukan dengan skenario yang sama antara satu aplikasi IDS dengan yang lainnya.

Pada tahap *Test Parameters*, peneliti melakukan instalasi dan konfigurasi alat uji (*framework* Pytbull) dengan menentukan beberapa parameter yang digunakan untuk melakukan pengujian terhadap aplikasi IDS berdasarkan 70 serangan dalam 11 modul yang dimiliki oleh *framework* Pytbull. 70 serangan diambil dari 300 serangan lainnya dikarenakan beberapa serangan lainnya menggunakan modul yang sama (ipReputation) dengan variasi IP berbahaya, sehingga hanya diambil beberapa.

Pada tahap *Test Procedures and Expected Results*, peneliti mengawali pengujian dengan melakukan uji secara normal terhadap lingkungan uji. Hal ini dilakukan untuk memastikan bahwa server web sudah dapat berjalan dan diakses, aplikasi IDS sudah dapat melakukan deteksi terhadap intrusi yang terjadi dalam lalu lintas jaringan. Apabila hal tersebut sudah berjalan dengan baik, dilanjutkan dengan pengujian sesuai skenario yang sudah dibuat.

Pengujian sesuai skenario yang dibuat dilakukan dengan menjalankan Pytbull. Hal tersebut dilakukan untuk setiap aplikasi IDS yang diimplementasikan. Ketika semua aplikasi IDS sudah dilaksanakan uji menggunakan Pytbull, maka seluruh hasil yang didapatkan dari pengujian setiap aplikasi IDS akan dianalisis. Analisis dilakukan dengan menghitung nilai akurasinya yang kemudian dilanjutkan membandingkan nilai akurasi antara satu aplikasi IDS dengan yang lainnya. Perbandingan ini dilakukan untuk mendapatkan aplikasi IDS yang memiliki nilai akurasi lebih tinggi dalam mendeteksi intrusi lalu lintas di jaringan.

4. HASIL DAN PEMBAHASAN

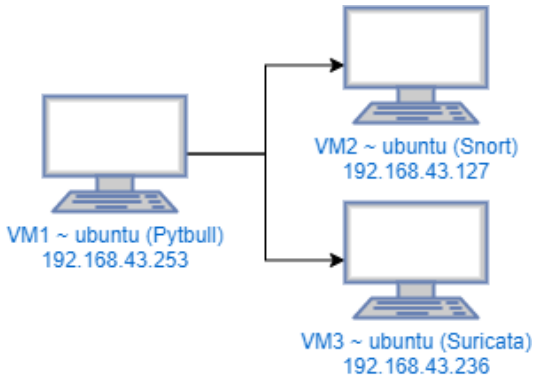
4.1. *Objective*

Objective adalah tahapan awal dari penelitian ini. Pada tahap ini, peneliti akan menentukan skenario yang akan digunakan dan mendefinisikan segala

kebutuhan yang digunakan dalam penelitian ini. Kebutuhan dalam penelitian ini meliputi spesifikasi dari lingkungan pengujian, aplikasi yang diuji, maupun alat uji. Tabel 1 menunjukkan spesifikasi lingkungan pengujian yang digunakan dalam penelitian ini. Gambar 2 menunjukkan topologi 3 mesin virtual yang ada di dalam virtualisasi.

Tabel 1. Spesifikasi Lingkungan Pengujian

Laptop
Sistem Operasi : Windows 10 Pro 64-bit (10.0, Build 17134)
Manufaktur Sistem : ASUSTeK COMPUTER INC.
Model Sistem : P453UA
BIOS : P453UA.302
Processoor : Intel(R)Core(TM) i3-6006U @2.00GHz(4 CPUs)
Memory : 12288 MB RAM
Aplikasi Virtualisasi
VirtualBox Graphical User Interface
Version 5.2.8 r121009 (Qt5.6.2)



Gambar 2. Gambaran Topologi dalam VirtualBox

Tabel 2 menunjukkan spesifikasi aplikasi IDS yang digunakan, sedangkan Tabel 3 menunjukkan spesifikasi dari Pytbull. Tiga skenario yang digunakan dalam penelitian ini dijelaskan pada Tabel 4.

Tabel 2. Spesifikasi Aplikasi IDS

Aplikasi Snort
Snort Version 2.9.15.1 GRE (Build 15125)
By Martin Roesch & The Snort Team
Using libpcap version 1.7.4
Using PCRE version: 8.43 2019-02-23
Using ZLIB version: 1.2.8
Mesin Virtual Instalasi Snort
Device name : ubuntu
Memory : 1024 MB
Processor : Intel(R) Core(TM) i3-6006U CPU @2.00 GHz
OS type : ubuntu 16.04 LTS 64-bit
Disk : 20 GB
Aplikasi Suricata
Suricata Version 5.0.2 RELEASE
Mesin Virtual Instalasi Suricata
Device name : ubuntu
Memory : 1024 MB
Processor : Intel(R) Core(TM) i3-6006U CPU @2.00 GHz
OS type : ubuntu 16.04 LTS 64-bit
Disk : 20 GB

Tabel 3. Spesifikasi Aplikasi Pytbull

Aplikasi Pytbull
Pytbull
Sebastien Damaye, aldeid.com
Version 2.0M
Mesin Virtual Aplikasi Pytbull
Device name : adam-VirtualBox
Memory : 6000 MB
Processor : Intel(R) Core(TM) i3-6006U CPU @2.00 GHz x 2
OS type : ubuntu 16.04 LTS 64-bit
Disk : 25 GB

Tabel 4. Skenario Pengujian

Skenario Pengujian		
Skenario 1	Pengujian dilakukan menggunakan Pytbull terhadap Snort menggunakan 3917 *rules* komunitas dari https://www.snort.org dan Suricata dengan 20620 *rules* dari `suricata-update`.	Bertujuan mengetahui akurasi deteksi Snort dan Suricata apabila digunakan *rules* asli yang disediakan.
Skenario 2	Pengujian dilakukan menggunakan Pytbull terhadap Snort menggunakan 29488 *rules* dan Suricata dengan menggunakan 30117 *rules* yang berasal dari https://rules.emergingthreats.net yang di perbarui pada tanggal 1 Juli 2020.	Bertujuan mengetahui akurasi deteksi Snort dan Suricata apabila digunakan *open source rules*.
Skenario 3	Pengujian dilakukan menggunakan Pytbull terhadap Snort dan Suricata menggunakan *rules* yang disamakan dan ditentukan oleh peneliti sebanyak 38 *rules* yang ditentukan berdasarkan serangan yang ada dalam 11 modul serangan Pytbull.	Bertujuan mengetahui akurasi deteksi Snort dan Suricata apabila digunakan *rules* yang sama dengan jumlah yang sama.

4.2. *Test Setup*

Test Setup merupakan tahap kedua dari peneltian ini. Pada tahap ini dilakukan instalasi dan konfigurasi terhadap lingkungan yang akan digunakan untuk pengujian. Instalasi Aplikasi IDS (Snort dan Suricata) dilakukan pada tahap ini.

a. Snort merupakan aplikasi *Network Intrusion Detection System* (NIDS) yang paling banyak digunakan untuk mendeteksi intrusi dengan mencari protokol, analisis konten, dan bebagai pra-prosesor [18]. Terdapat beberapa dependensi yang perlu diinstal sebelum melakukan instalasi Snort. Dependensi adalah aplikasi lain yang dibutuhkan oleh aplikasi utama untuk memenuhi persyaratan instalasinya. Contoh beberapa dependensi aplikasi Snort adalah ethtool, build-essential, dan lib-pcap-dev.
b. Suricata adalah sistem deteksi intrusi jaringan bersifat *open source* yang digunakan untuk memeriksa lalu lintas jaringan menggunakan *signatures* dan *rules* [22]. Suricata membutuhkan beberapa dependensi sebelum dilakukan

instalasinya seperti libpcap, libpcre, dan libmagic.

4.3. *Test Parameters*

Test Parameters merupakan tahap ketiga dari penelitian ini. Pada tahap ini dilakukan instalasi dan konfigurasi *framework* yang digunakan untuk melakukan pengujian, yaitu aplikasi Pytbull. Selain itu, *rules* yang digunakan oleh kedua aplikasi juga didefinisikan pada bab ini. Pada Pytbull harus dipastikan bahwa parameter yang digunakan untuk pengujian sudah aktif, yaitu dengan memberi nilai 1 pada setiap modul pengujian seperti pada Tabel 5 dan *rules* yang sudah ditentukan telah diimplementasikan sesuai skenario masing-masing.

Tabel 5. Konfigurasi Modul Pengujian

Config.cfg	
[TESTS]	
clientSideAttacks	= 1
test*Rules*	= 1
badTraffic	= 1
fragmentedPackets	= 1
bruteForce	= 1
evasionTechniques	= 1
shellCodes	= 1
denialOfService	= 1
pcapReplay	= 1
normalUsage	= 1
ipReputation	= 1

4.4. *Test Procedures and Expected Results*

Pengujian diawali dengan uji coba normal, yaitu uji coba untuk memastikan bahwa Snort, Suricata, dan Pytbull sudah mau berjalan sesuai dengan yang diharapkan. Snort dan Suricata dilakukan uji coba normal dengan melakukan koneksi sederhana, sedangkan Pytbull dilakukan dengan melakukan konfigurasi modul tidak aktif. Setelah uji coba normal berhasil dilakukan, dilanjutkan dengan melakukan uji sesuai skenario yang telah ditentukan.

Tabel 6 dan Tabel 7 merupakan hasil uji dari 3 skenario yang dijalankan terhadap Snort versi 2.9.15.1 dan Suricata versi 5.0.2. Secara keseluruhan Suricata memiliki nilai akurasi yang lebih tinggi yaitu 0.611451088 (61%), sedangkan Snort hanya memiliki nilai akurasi 0,31267153 (31%).

Tabel 8 menunjukkan hasil bahwa dari 3 skenario yang dijalankan, penggunaan *memory* Suricata yang menggunakan fitur *multi-threading* lebih stabil daripada Snort yang hanya mendukung fitur *single-threading*. Terkait lama waku yang dibutuhkan dalam pengujian, Suricata dalam 2 skenario awal memiliki waktu yang sedikit lebih lama daripada Snort. Hal ini disebabkan oleh kondisi *rules* Suricata yang jauh lebih banyak pada skenario 1 dan 2 daripad *rules* yang dimiliki Snort. Namun, pada skenario 3 ketika jumlah *rules* yang dimiliki sama, maka Suricata memiliki waktu deteksi lebih cepat daripada Snort.

Tabel 6. Hasil Keseluruhan Akurasi Snort

Hasil Keseluruhan Akurasi Snort				
Modul Serangan	**Skenario 1**	**Skenario 2**	**Skenario 3**	**Rata-Rata**
clientSideAttacks	0	0,822019417	0,923397052	0,581805
testRules	0,44444444	0,90361126	0,60409731	0,650717
badTraffic	0	0,03196347	0,02140673	0,017790
fragmentedPackets	0,25	0,23280423	0,07392364	0,185575
bruteForce	0	0	0,57071282	0,190237
evasionTechniques	0,98216939	0,84578566	0,82672176	0,884892
shellCodes	0,14285714	0,00492610	0,06596167	0,071248
denialOfService	0	0,89523932	0,92274819	0,605995
pcapReplay	0	0	0	0
normalUsage	0,66666667	0,0548054	0,03189352	0,251121
ipReputation	0	0	0	0
Rata-Rata Akurasi Snort	0,22601323%	0,34465044 (35%)	0,36735116 (37%)	0,312671 53 (31%)

Tabel 7. Hasil Keseluruhan Akurasi Suricata

Hasil Keseluruhan Akurasi Suricata				
Modul Serangan	**Skenario 1**	**Skenario 2**	**Skenario 3**	**Rata-Rata**
clientSideAttacks	0,09090909	0,4	0,16666667	0.2191919 2
testRules	0,82142857	0,87037037	0,80555556	0.8324515
badTraffic	0,73333333	0,55555556	0,57142857	0.6201058 2
fragmentedPackets	0	0	0	0
bruteForce	1	0,975	0,97560976	0.9835365 87
evasionTechniques	0,99705263	0,99429006	0,98875562	0.9933661 03
shellCodes	0,72	0,72916667	0,66666667	0.7052777 8
denialOfService	0,99972364	0,91666667	0,99954819	0.9719795
pcapReplay	0	0	0	0
normalUsage	1	0,91666667	0,9	0.9388888 9
ipReputation	0,4	0,33333333	0,42105263	0.3847953 2
Rata-Rata Akurasi Suricata	0,61476862%	0,62910493 3	0,59048033 (59%)	0.6114510 88 (61%)

Tabel 8. Perbandingan *Rules*, *Alerts*, Waktu, dan *Memory*

Kategori	**IDS**	**Skenario 1**	**Skenario 2**	**Skenario 3**
Jumlah *Rules*	**Snort**	3917	29488	38
	Suricata	20620	30117	38
Jumlah *Alert*	**Snort**	676	102821	22864
	Suricata	9737	5589	8153
Lama Uji	**Snort**	8 menit 13,06 detik	10 menit 14,96 detik	10 menit 55,50 detik
	Suricata	8 menit 25,75 detik	11 menit 23,54 detik	10 menit 2,72 detik
Memory (*bytes*)	**Snort**	51566528	53358908	4362480
	Suricata	7489720	7503496	7492016

5. KESIMPULAN

Berdasarkan perbandingan nilai akurasi dari hasil uji terhadap Snort versi 2.9.15.1 dan Suricata versi 5.0.2 dengan menggunakan Pytbull dalam 3 skenario, maka dapat disimpulkan bahwa akurasi Suricata lebih tinggi (61%) daripada Snort (31%) karena memiliki *rules* yang lebih banyak. Walaupun *rules* lebih banyak, namun penggunaan *memory* Suricata lebih stabil daripada Snort karena menggunakan fitur *multi-threading* yang dimilikinya. Berikut detail simpulan akurasi dari setiap skenario yang dilakukan:

a. Pada Skenario 1, Suricata versi 5.0.2 menggunakan *rules* dari `suricata-update` memiliki nilai akurasi yang lebih tinggi dengan nilai 0,614768 (62%) daripada Snort versi 2.9.15.1 yang menggunakan *rules* komunitas dari https://www.snort.org dengan nilai akurasi 0,226013 (27%).
b. Pada Skenario 2, Suricata versi 5.0.2 menggunakan *rules* dari https://*rules*.emergingthreats.net memiliki nilai akurasi yang lebih tinggi dengan nilai 0,629104933 (63%) daripada Snort versi 2.9.15.1 yang menggunakan *rules* dari https://*rules*.emergingthreats.net dengan nilai 0,34465044 (35%).
c. Pada Skenario 3, Suricata versi 5.0.2 menggunakan *rules* yang ditentukan peneliti dan disamakan untuk *rules* Snort mempunyai nilai akurasi 0,59048033 (59%), yaitu lebih tinggi daripada Snort versi 2.9.15.1 dengan nilai 0,36735116 (37%).

6. SARAN

Dari penelitian yang dilakukan masih dapat dikembangkan dengan cara melakukan pengujian lebih lanjut menggunakan aplikasi IDS versi lebih baru dan spesifikasi mesin virtual instalasi yang lebih bervariasi.

REFERENSI

[1] E. Setiawan, "Arti kata internet - Kamus Besar Bahasa Indonesia (KBBI) Online," *Badan Pengembangan dan Pembinaan Bahasa, Kemdikbud*, 2019. [Online]. Available: https://kbbi.web.id/internet. [Accessed: 07-Nov-2019].

[2] Maxmanroe, "Pengertian INTERNET adalah: Definisi, Fungsi, Manfaat, Dampak Internet," 2017. [Online]. Available: https://www.maxmanroe.com/vid/teknologi/internet/pengertian-internet.html.

[3] Miniwatts Marekting Group, "World Internet Users Statistics and 2019 World Population Stats," 2019. [Online]. Available: https://www.internetworldstats.com/stats.htm. [Accessed: 06-Nov-2019].

[4] C. (BSSN) Lim and A. (BSSN) Yusuf, "Laporan Tahunan Honeynet Project BSSN IHP 2018," 2018.

[5] A. Khraisat, I. Gondal, P. Vamplew, and J. Kamruzzaman, "Survey of intrusion detection systems: techniques, datasets and challenges," *Cybersecurity*, vol. 2, no. 1, 2019.

[6] S. K. Biswas, "Intrusion Detection Using Machine Learning: A Comparison Study," *Int. J. Recent Technol. Eng.*, vol. 8, no. 2 Special Issue 6, pp. 832–837, 2018.

[7] F. M. Isa, S. Saad, A. Firdaus, A. Fadzil, and R. M. Saidi, "Comprehensive Performance Assessment on Open Source Intrusion Detection System," in *Proceedings of the Third International Conference on Computing, Mathematics and Statistics (iCMS2017)*, 2019, pp. 579–584.

[8] S. Cooper, "2019 Best Intrusion Detection Systems (10+ IDS Tools Reviewed)," *Comparitech.Com*, 2019. [Online]. Available: https://www.comparitech.com/net-admin/network-intrusion-detection-tools/.

[9] E. Albin and N. C. Rowe, "A realistic experimental comparison of the Suricata and Snort intrusion-detection systems," in *Proceedings - 26th IEEE International Conference on Advanced Information Networking and Applications Workshops, WAINA 2012*, 2012, pp. 122–127.

[10] M. H. Bhuyan, D. K. Bhattacharyya, and J. K. Kalita, *Network Traffic Anomaly Detection Techniques and Systems*. 2017.

[11] S. Damaye, "pytbull - IDS_IPS Testing Framework - documentation," 2016. [Online]. Available: http://pytbull.sourceforge.net/index.php?page=documentation#description. [Accessed: 02-Dec-2019].

[12] W. Fathoni, "DETEKSI PENYUSUPAN PADA JARINGAN KOMPUTER MENGGUNAKAN IDS SNORT," Bandung, 2015.

[13] OISF, "Suricata User Guide Release 5.0.2." OISF, p. 91, 2020.

[14] Snort Project Team, "SNORT Users Manual 2.9.16." p. 698, 2020.

[15] K. Scarfone and P. Mell, "Guide to Intrusion Detection and Prevention Systems (IDPS)." National Institute of Standards and Technology, United States of America, 2007.

[16] Wikimedia, "Snort," 2020. [Online]. Available: https://en.wikipedia.org/wiki/Snort_(software). [Accessed: 03-Jul-2020].

[17] Suricata Open Information Security Foundation (OISF), "Suricata Open Source IDS / IPS / NSM engine," *Https://Suricata-Ids.Org*, 2017. [Online]. Available:

https://suricata-ids.org/. [Accessed: 03-Jul-2020].

[18] Rapid7, "How to Install Snort NIDS on Ubuntu Linux," 2017. [Online]. Available: https://blog.rapid7.com/2017/01/11/how-to-install-snort-nids-on-ubuntu-linux/. [Accessed: 11-Mar-2020].

[19] Rapid7, "How to Install Suricata NIDS on Ubuntu Linux," 2017. [Online]. Available: https://blog.rapid7.com/2017/02/14/how-to-install-suricata-nids-on-ubuntu-linux/. [Accessed: 11-Mar-2020].

[20] W. Park and S. Ahn, "Performance Comparison and Detection Analysis in Snort and Suricata Environment," *Wirel. Pers. Commun.*, vol. 94, no. 2, pp. 241–252, 2016.

[21] B. Balarajah, C. Rossenhoevel, and B. Monkman, "Benchmarking Methodology for Network Security Device Performance draft-ietf-bmwg-ngfw-performance-01," *Internet Engineering Task Force (IETF)*. IETF Trust, pp. 1–57, 2019.

[22] OISF, "What is Suricata," 2019. [Online]. Available: https://suricata.readthedocs.io/en/suricata-5.0.2/what-is-suricata.html. [Accessed: 11-Mar-2020].