Rancang Bangun Sistem Pengiriman Alert Instrusion Detection System Suricata Melalui Telegram

Zaki Akhyar[1], Hendrawaty[2] , Azhar[3]

[1,2,3] *Jurusan Tekniknologi Informasi dan Komputer Politeknik Negeri Lhokseumawe*
Jln. B.Aceh Medan Km.280 Buketrata 24301 INDONESIA

[1]zaki.akhyar@outlook.com

Abstrak— **Instrusion detection system (IDS) Suricata merupakan sebuah aplikasi berbasis open source yang dapat mendeteksi aktivitas yang mencurigakan dalam sebuah jaringan komputer. IDS Suricata memantau lalu lintas (trafic) yang melewatinya dan mengeluarkan peringatan (alert) jika ada paket yang mencurigakan. Alert tersebut dikirimkan kedalam log file. Salah satu log file untuk menampung alert (output) suricata adalah unified2. Informasi pada unified2 akan dikirimkan kedalam database snorby melalui barnyard2. Pada penelitian ini digunakan intrusion detection system suricata untuk mendeteksi serangan pada jaringan. Kemudian hasil deteksi (output) suricata tersebut akan dikirimkan ke handphone administrator melalui pesan instan telegram. Informasi yang dikirimkan terdiri dari, waktu serangan, ip sumber, ip tujuan, dan nama serangan. Dengan demikian Administrator akan mendapatkan informasi secara realtime tentang serangan yang terjadi pada jaringan. Dari hasil penelitian diperoleh bahwa sistem telah berhasil mengirimkan notifikasi/pesan secara realtime ke handphone administrator apabila ada serangan yang terdeteksi oleh suricata.**
Kata kunci— IDS, Pengiriman Alert, Suricata, Suricata Alert, Telegram.

Abstract— **Instrusion detection system (IDS) Suricata is an open source based application that can detect suspicious activity in a computer network. IDS Suricata monitors the traffic (traffic) that passes through it and issues an alert when a suspicious package exists. The alerts are sent to the log file. One log file to accommodate alert (output) suricata is unified2. Information on unified2 will be sent to the snorby database via barnyard2. In this study used intricular detection system suricata to detect attacks on the network. Then the results of detection (output) suricata will be sent to the mobile administrator via instant message telegram. The information transmitted consists of, timing of attack, source ip, destination ip, and name of attack. Thus the administrator will get information in realtime about the attacks that occur on the network. From the results of the study obtained that the system has managed to send notifications/messages in realtime to the mobile administrator if there is an attack detected by suricata.**
Keywords— Alert Delivery, IDS, Suricata, Suricata Alert, Telegram.

I. PENDAHULUAN

Pada saat ini jaringan komputer merupakan bagian utama dari era komunikasi dan internet. Tujuan dari sistem komputer yang saling berhubungan agar lebih efisien dalam pertukaran informasi. Pada sisi yang lain semakin pesat pengguna jaringan komputer juga sebanding dengan bertambahnya serangan pada suatu sistem oleh *intruder*. Sebuah serangan adalah realisasi ancaman untuk menemukan dan meng-ekploitasi kelemahan suatu sistem, untuk itu salah satu pilihan sistem keamanan pada jaringan komputer yaitu suatu sistem pendeteksi serangan (*Instrusion Detection System*).

Intrusion Detection System (IDS) didefinisikan sebagai suatu proses pemantauan kejadian yang terjadi pada sistem keamanan komputer atau jaringan komputer.[1] Tujuan utama dari sistem deteksi intrusi ini adalah sebagai *alarm*. Suricata merupakan contoh *instrusion detection system* yang banyak digunakan dan bersifat *open source*.

Suricata merupakan suatu sistem deteksi intrusi berbasis *open source* yang dikembangkan oleh *Open Information Security Foundation* (OISF). Keluaran (*output*) deteksi instrusi pada suricata secara *default* ditampilkan berbasis command line pada terminal linux. *Output* tersebut tidak mudah untuk dipahami.

Administrator jaringan merupakan orang yang bertanggung jawab terhadap suatu jaringan, administrator dituntut untuk mengetahui kondisi yang terjadi pada jaringan yang dikelolanya, terutama yang berhubungan dengan sistem keamanan. Umumnya suatu jaringan dilengkapi dengan perangkat keamanan contohnya seperti IDS dan firewall. Untuk mengetahui serangan-serangan yang berhasil dideteksi oleh IDS, administrator harus rajin memeriksa log *file* secara berkala (teratur). Hal tersebut menuntut adminstrator harus *standby* di ruangan kerjanya. Tetapi pada kenyataanya administrator tidak dapat *standby* 24 jam.

Berdasarkan permasalahan diatas, maka penelitian ini dilakukan bertujuan untuk membangun suatu sistem yang dapat mendeteksi serangan yang masuk kedalam jaringan. Serangan yang masuk dideteksi oleh IDS Suricata. Kemudian *alert* yang dihasilkan oleh IDS suricata akan dikirimkan kedalam *database*. Setelah itu sistem akan mengirimkan informasi penting secara *realtime* berupa waktu serangan, ip komputer dari penyerang, ip komputer korban, dan nama serangan ke *handphone* administrator melalui pesan instan telegram. Sehingga administrator dapat mengetahui kondisi jaringannya walaupun berada diluar ruangan.

Darapareddy (2012) melakukan penelitian yang menggunakan *instrusion detection system* (IDS) sebagai dasar menjelaskan deteksi instruksi adalah proses monitoring komputer atau jaringan dari aktivitas atau kegiatan yang tidak sah.[2]

IDS (*Instrusion Detection System*) juga dapat digunakan untuk memonitor lalu lintas jaringan sehingga dapat mendeteksi jika terjadi nya serangan pada jaringan. IDS hanya

berfokus untuk mengidentifikasi serangan yang terjadi dan ketika serangan itu terjadi IDS akan membuat sebuah *report*/laporan.[3]

Beberapa tools IDS/IPS yang terkenal yaitu Snort dan Suricata Pada tahun 2016 Wohyung Park dan Seongjin Ahn melakukan penelitian tentang "*Performance Comparison and Detection Analysis in Snort and Suricata Environment*", didalam penelitiannya mereka membandingkan dan mengalisa perfoma IDS Snort dan Suricata dari hasil penelitian disimpulkan bahwa Snort dan Suricata memiliki karakteristik masing-masing Snort memiliki kinerja yang lebih baik pada pangsa CPU rendah, sedangkan Suricata memiliki keunggulan pada teknik multi-threading, identifikasi dan ekstraksi, HTTP *normalizer & parser* dan juga Suricata dapat ditingkatkan performanya jika GPU di aktifkan.[4]

Suricata mampu mendeteksi sebuah aktifitas jaringan dan mengindentifikasi instrusi dibantu dengan *rules* yang ter-intergrasi. IDS ini memindai setiap paket data yang dikirim pada sesi TCP dan mengubah menjadi informasi kedalam bentuk log. *Rules* pada suricata berperan dalam mengindetifikasi ancaman yang terjadi pada sebuah host.[5] David dan Benjamin telah menganalisis Snort dan suricata dan menyimpulkan bahwa Suricata memiliki tingkat akurasi lebih tinggi daripada Snort.[6] Terdapat beberapa definisi yang terkait dengan instruksi deteksi yaitu :

A. Snorby

Snorby ialah salah satu aplikasi *front end web* berbasis ruby on rails untuk memantau sistem keamanan jaringan komputer dengan *user interface* berbasis GUI (Graphical User Interface). Beberapa fitur kelebihan snorby adalah *Metrics Reports* yaitu menampilkan data-data dan kejadian (event) IDS dengan tampilan grafis.[7]

B. Barnyad2

Barnyard2 adalah tool *open source* sebagai penerjemah *alert* unified. Barnyard2 bekerja dengan membaca *file* log unified2 dan memasukannya kedalam *database*. Jika *database* tidak terkoneksi maka Barnyard2 akan memasukan semua data ketika *database* tersedia kembali sehingga tidak ada *alert* atau log yang hilang. Barnyard2 dapat berjalan pada tiga (3) mode antara lain:

1. *Batch* (*or one-shot*), Barnyard2 akan memproses secara eksplisit *file* yang telah ditentukan.
2. *Continual*, Barnyard2 akan mulai memproses dari lokasi *file* dan *file* tertentu dan terus memproses data baru (new spool file).
3. *Continual with bookmark*, Barnyard2 akan memproses menggunakan *file checkpoint* (waldo file) untuk melacak dimana lokasi berada.[8]

C. Shell Scripting

Shell adalah program (penerjemah perintah) yang menjembatani user dengan sistem operasi dalam hal ini kernel (inti sistem operasi), umumnya shell menyediakan prompt sebagai *user interface*, tempat dimana user mengetikkan perintah-perintah yang diinginkan baik berupa perintah internal shell (*internal command*), ataupun perintah eksekusi suatu *file* progam (*external command*).

Shell pada sistem operasi keluarga Unix misalnya Linux sampai saat ini memiliki beberapa sistem shell, diantara lain:

a. Bourne shell(sh),
b. C shell(csh),
c. Korn shell(ksh),
d. Bourne again shell(bash), dll.

Bash (bourne again shell) adalah shell Unix dan bahasa perintah yang ditulis oleh Brian Fox untuk Proyek GNU sebagai pengganti perangkat lunak gratis untuk Bourne shell. Pertama kali dirilis pada tahun 1989, telah didistribusikan secara luas karena merupakan shell *default* pada distribusi Linux dan OS X. Bash adalah *command processor* yang biasanya berjalan di jendela teks, di mana pengguna mengetik perintah yang menyebabkan tindakan. Bash juga dapat membaca perintah dari *file*, yang disebut skrip.

Skrip shell adalah program komputer yang dirancang untuk dijalankan oleh shell unix, seorang penerjemah baris perintah. Berbagai dialek skrip shell dianggap bahasa scripting. Operasi umum yang dilakukan oleh skrip shell termasuk manipulasi *file*, eksekusi program, dan teks pencetakan.[9]

Pemrograman shell yaitu menyusun atau mengelompokkan beberapa perintah shell (internal atupun eksternal *command*) menjadi kumpulan perintah yang melakukan tugas tertentu sesuai tujuan penyusunnya. Kelebihan shell di linux dibanding sistem operasi lain adalah bahwa shell di linux memungkinkan kita untuk menyusun serangkaian perintah seperti halnya bahasa pemrograman (*interpreter language*), melakukan proses I/O, menyeleksi kondisi, looping, membuat fungsi, dsb. adalah proses - proses yang umumnya dilakukan oleh suatu bahasa pemrograman, jadi dengan shell di linux kita dapat membuat program seperti halnya bahasa pemrograman, untuk pemrograman shell pemakai unix atau linux menyebutnya sebagai shell scripting.[10]

D. Telegram

Telegram Messenger adalah aplikasi pesan singkat multiplatform yang memungkinkan pengguna mengirimkan pesan singkat rahasia yang dienkripsi *end-to-end* sebagai keamanan tambahan. Nicolas Bernaerts (2016) pernah mengimplementasikan pengiriman notifikasi monitoring sistem dengan uji coba sederhana dengan menggunakan bahasa bash.[11] Selain Nicolas, pada awal 2017 Deekshith hadil melakukan percobaan mengirimkan notifikasi untuk memberitahukan kepada administrator jika penggunaan *resource* server tinggi.[12]

II. METODOLOGI PENELITIAN

A. Analisa Kebutuhan

Instrusion detection system (IDS) server membutuhkan beberapa aplikasi dan *tools* untuk penggunaannya.

Gambar 1. Analisis kebutuhan sistem IDS Suricata

Suricata merupakan *software* untuk mendeteksi serangan yang terjadi dan menghasilkan output yang tersimpan dalam bentuk *file* log. Untuk menyimpan log yang dihasilkan suricata ke dalam database dibutuhkannya aplikasi barnyard2, target database penyimpanannya yaitu database yang telah dibuat oleh snorby. Kemudian *database* yang telah berisikan event / serangan ditampilkan ke antar muka web snorby. API Telegram merupakan izin penggunaan layanan pesan telegram, agar server IDS dapat mengirimkan notifikasi, pengambilan informasi event / serangan dari *database* menggunakan *coding* PHP lalu dikirimkan ke telegram administrator. Bash *Scripting* adalah skrip / program yang digunakan untuk membuat semua proses diatas berjalan pada saat server dijalankan.

B. Alat dan Bahan

Ada beberapa alat dan bahan yang dibutuhkan dalam melakukan penelitian (*hardware* dan *software*). Spesifikasi *hardware* yang digunakan untuk keperluan penelitian ini, sebagai berikut:

TABEL I
SPESIFIKASI HARDWARE YANG DIGUNAKAN

No.	Nama Peralatan	Spesifikasi	Jumlah	Fungsi
1	PC Server	Intel Core i5, 4 GB RAM,500GB HDD,100MB/Ethernet	1	Server IDS
2	Notebook	Intel Core i5, 6GB RAM, 1TB HDD,100MB/Ethernet	1	Laptop *attacker*
3	PC WebServer	Intel Core i5, 4 GB RAM, 1 TB HDD, 1 Gbps/Ethernet	1	Server target penyerangan / korban
4	Switch	Cisco Switch	1	Media penghubung jaringan
5	Smartphone	Lenovo A7000 , 2 GB RAM	1	*Handphone* Administrator

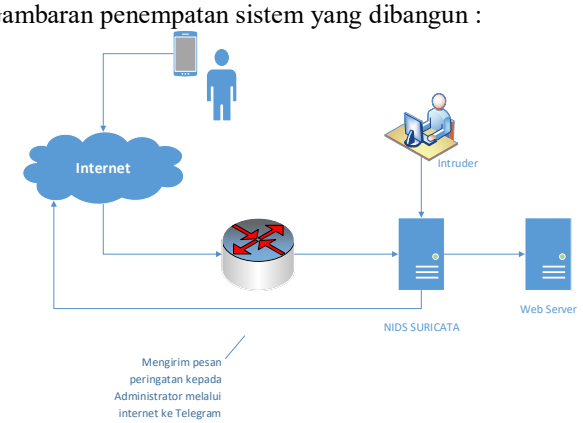
Spesifikasi *software* yang digunakan untuk keperluan penelitian ini, sebagai berikut:

TABEL II
SPESIFIKASI PERANGKAT LUNAK

No.	Spesifikasi	Jumlah	Fungsi
1	Ubuntu 16.04	1	Sistem operasi pada sisi server.
2	Suricata	1	Software IDS
3	Snorby	1	Tool pembuat log grafis.
4	Barnyard2	1	Membaca alert file dan dituliskan ke database
5	Kali Linux	1	Sistem operasi pada sisi penyerang
6	Telegram API	1	Mengirimkan alert suricata ke pesan telegram administrator
7	PHP 7	1	Bahasa pemograman pembangunan sistem pengiriman alert
8	MySQL Server	1	Database server
9	Hping3	1	Software ping flood (DoS)
10	Nmap	1	Tool port scanning
11	Htop	1	Software monitoring resource

C. Perancangan Ilustrasi Arsitektur Jaringan

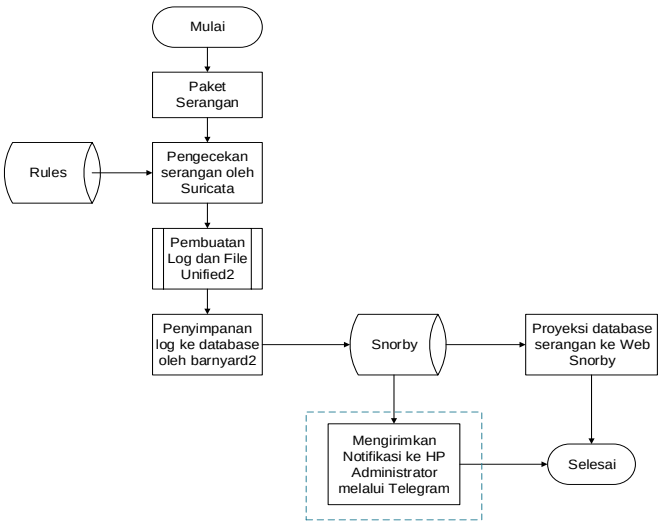
IDS Suricata akan menjembatani jalur antara jaringan luar (internet) dengan lokal, pada saat intruder (penyerang) menyerang jaringan lokal maka akan di *capture* paket yang masuk ke jaringan, lalu dilakukan pengecekan oleh Suricata jika paket yang dikirimkan berupa serangan maka NIDS Suricata akan membuat log *file* dan mengirimkan pesan peringatan ke telegram Adminstrator melalui internet. Gambaran penempatan sistem yang dibangun :



Gambar 2. Ilustrasi arsitektur jaringan untuk pengujian IDS Suricata

D. Diagram Proses Pengiriman Alert ke HP Admin

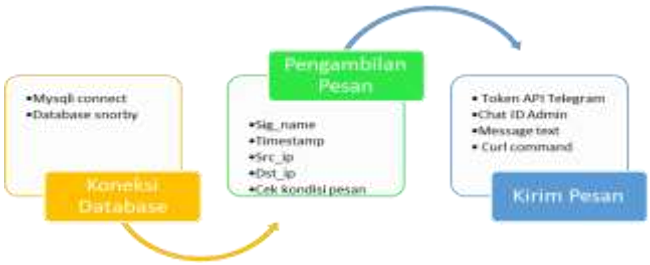
Tahapan ini membahas tentang proses dari awal deteksi serangan hingga terkirim *alert* kepada administrator. Berikut diagram proses pengiriman *alert* ke HP Admin :



Gambar 3. Alur Proses Pengiriman Alert

Proses awal ketika adanya serangan kepada sistem, Suricata akan mendeteksi seluruh paket yang melewatinya, pengecekan terjadi berdasarkan rules yang telah dibuat oleh administrator, jika paket terdeteksi oleh Suricata , proses selanjutnya Suricata akan membuat log *file* (fast.log) dan Log Unified2. Selanjutnya barnyard2 membaca Log Unified2 yang dihasilkan oleh Suricata dan akan disimpan pada *database* Snorby. Setelah masuk kedalam *database* dapat ditampilkan, yang pertama menampilkan/memproyeksikan hasil serangan yang dideteksi pada *interface* web menggunakan Snorby dan yang kedua penilitian penulis mengambil informasi serangan terbaru yang tersimpan dalam *database* dan mengirimkan ke pesan instan telegram administrator.

E. Proses Skrip Pengiriman Pesan



Gambar 4. Proses Skrip Pengiriman

Hal yang pertama kali dilakukan yaitu mengakses *database* snorby . *Database* ini secara *default* telah dibuat pada saat snorby di-installkan. Skema yang dimiliki snorby sudah mengikuti aturan dasar skema dari barnyard2. Selanjutnya yaitu fungsi pengambilan pesan, nama *field* yang diambil yaitu: id (id serangan), sig_name (keterangan event), src_ip (IP asal), dst_ip (IP tujuan) , timestamp (waktu event) dan notes_count. Secara *default* notes_count berisikan 0 , pada penilitian ini digunakan *field* notes_count untuk membuat kondisi pengambilan pesan, pesan yang diambil berdasarkan id dimana notes_count bernilai 0 , setelah pesan diambil notes_count akan di *update* dengan nilai 1 yang berarti pesan tidak akan diambil lagi seterusnya . Format penulian pesan yang terkirim kepada administrator yaitu seperti dibawah ini :
$Ket_serangan dari $src_ip ke $dst_ip pada $timestamp
Pesan yang telah diambil diteruskan kedalam fungsi telegram. Didalamnya terdapat akses token dan chat id penerima pesan dalam hal ini yaitu chat id administrator. Lalu isi pesan yang telah diambil tersebut dikirimkan kepada administrator.

F. Pengujian Sistem

Pengujian akan dilakukan oleh peneliti yang bertindak sebagai administrator dan *attacker*, target serangan yaitu sebuah server yang telah terinstall webserver. Berikut tahapan pengujian secara garis besar, terdapat beberapa pengujian yang dilakukan yaitu, pengujian koneksi sesuai dengan ilustrasi jaringan, pengujian skrip pengoperasian suricata, pengujian suricata apakah berfungsi dengan baik, pengujian barnyard2, pengujian serangan ping of death, port scanning serta pengujian pengiriman *alert* ke handphone administrator.

III. Hasil dan Pembahasan

Hasil penelitian yang telah dilakukan yang terdiri dari pembahasan hasil uji koneksi dari seluruh perangkat yang terhubung pada jaringan, hasil uji *script* pengoperasian suricata, hasil pengujian barnyard2, dan snorby, hasil dari pengujian serangan Ping Of Death dan Port scanning, serta hasil pengujian pengiriman notifikasi ke *handphone* administrator melalui telegram.

A. Uji Koneksi pada Jaringan

Uji koneksi dilakukan pada ketiga perangkat, yaitu PC IDS Suricata, Webserver dan Attacker, proses yang dilakukan untuk mengetahui koneksi ketiga perangkat terhubung yaitu dengan cara menggunakan ping kepada setiap IP perangkat. Berikut salah satu uji koneksi antara IDS Suricata dengan Webserver.

Gambar 5. Uji Koneksi IDS Suricata ke webserver

B. Hasil Uji Skrip Pengoperasian Suricata

Hasil pengujian skrip pengoperasian dimaksudkan untuk melihat apakah rancangan skrip pengoperasian suricata, snorby dan kirimpesan berjalan dengan baik. Hasil pengujian skrip pengoperasian IDS suricata ini dilakukan untuk memastikan apakah fungsi *d_start(), d_stop*, dan *d_status* pada IDS suricata yang terdapat pada /opt/bin/suricata.sh

sudah dapat bekerja dengan baik atau belum. Gambar 6 memperlihatkan *script* pengoperasian (*script d_start(), d_stop*, dan *d_status)* IDS suricata.

Gambar 6. Script suricata.sh

Pada Gambar 6 dapat dilihat bahwa pada *script d_start()* terdapat beberapa *command*, yaitu:

1. *IP link set* enp0s8 promisc on
   Perintah tersebut dimaksudkan untuk mengaktifkan mode promiscous pada *network adapter* enp0s8. Pada mode promiscous berarti mengijinkan perangkat/komponen jaringan untuk membaca setiap paket jaringan yang melewatinya.
2. Suricata -c /etc/suricata/suricata.yaml -i enp0s8 --init-errors-fatal --pidfile /usr/local/var/run/suricata.pid >> /var/log/suricata/suricata-init-script.log & Perintah tersebut dimaksudkan untuk menjalankan suricata dengan *file* konfigurasi yang terletak pada /etc/suricata/suricata.yaml, *network adapter* yang mengawasi lalu lintas jaringan yaitu enp0s8, --init-errors-fatal adalah opsi yang memberitahukan jika terjadi error pada saat suricata dijalankan, --pidfile merupakan opsi untuk menyimpan nomor proses id kedalam *file* yaitu /usr/local/var/run/suricata.pid dan menyimpan seluruh output dari perintah suricata tersebut kedalam *file* suricata-init-script.log.
3. echo "Proses ID --> $(cat /usr/local/var/run/suricata.pid)"
   Perintah tersebut berfungsi untuk menampilkan proses ID suricata.

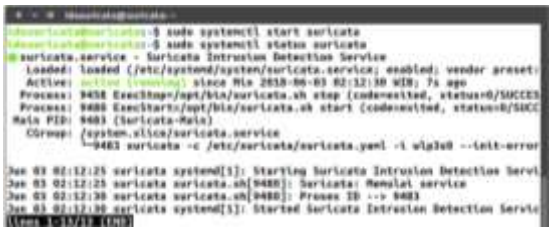
Agar *script* pengoperasian Suricata (suricata.sh) dapat berjalan pada saat server dihidupkan dibutuhkan satu *file* init systemd seperti pada gambar berikut :

Gambar 7. File *Service* Suricata.Service

Gambar diatas merupakan *file* /etc/systemd/system/ suricata.service . pada *file* tersebut harus berisi fungsi start stop dan restart yang scriptnya berada pada *file* /opt/bin/suricata.sh, *file* ini berfungsi supaya fungsi start,stop,dan restart tersebut dapat dieksekusi atau dijalankan.

Pengujian *script d_start()* suricata (untuk memulai *service*) dilakukan dengan cara membuka terminal dan mengetikkan perintah `$ sudo systemctl start suricata`, seperti yang terlihat pada Gambar 8. Hasil dari proses pengujian *script d_start()* menunjukkan bahwa *script d_start()* sudah dapat berjalan sesuai dengan fungsinya. Pengujian *script d_status* suricata dimaksudkan untuk melihat status IDS suricata apakah dalam keadaan sudah berjalan (*running*) atau dalam keadaan berhenti (*stop*). Pengujian dilakukan dengan cara membuka terminal dan mengetikkan perintah:

`$ sudo systemctl status suricata`, seperti yang terlihat pada Gambar 8. hasil dari status yang diperoleh adalah Active: active (running). Hal ini menunjukkan bahwa *script d_start* yang dibuat sudah dapat berjalan dengan baik sebagaimana fungsinya.



Gambar 8. Pengoperasian Suricata (*Start* dan *Status*)

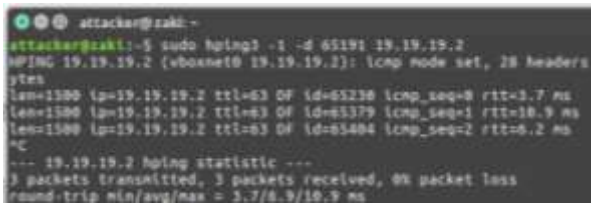
Pada fungsi *d_stop* terdapat perintah kill yang berfungsi untuk menghentikan IDS suricata yang sedang berjalan. Pengujian dilakukan dengan cara menjalankan IDS suricata terlebih dahulu, kemudian mengetikkan perintah `$ sudo systemctl stop suricata` pada terminal. hasil (berhasil atau tidak) dari *script d_stop* yang dijalankan dapat dilihat dengan melihat status dari IDS suricata.

C. Pengujian Sistem Secara Keseluruhan

Hasil pengujian sistem secara keseluruhan dimaksudkan untuk untuk mengetahui apakah sistem yang telah dibangun berfungsi dengan baik atau tidak. Pengujian dilakukan dengan cara melakukan penyerangan pada sistem dengan seranngan ping of death dan port scanning. Kemudian melihat apakah *alert* suricata sudah berhasil terkirim kedalam *database*

snorby. Setelah itu memeriksa apakah pesan pemberitahuan dapat terkirim ke *handphone* administrator melalui pesan instan telegram secara *realtime*.

Penyerangan ping of death dilakukan dari komputer *attacker* dengan mengetikkan perintah hping3 -1 -d 65191 19.19.19.2 pada terminal seperti yang tampak pada Gambar 9.



Gambar 9. Pengujian serangan ping of death

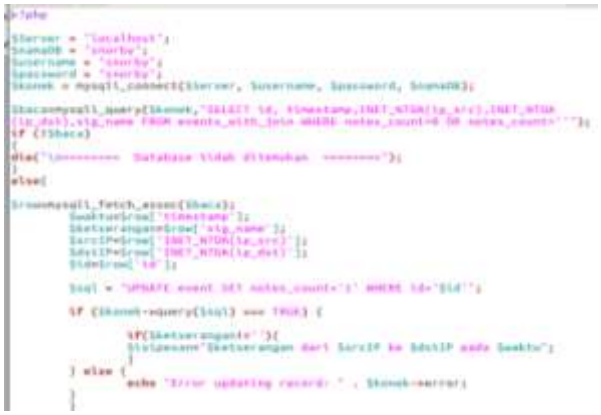
Perintah tersebut akan menyebabkan terkirimnya paket icmp yang berukuran 65191 bytes dengan interval waktu 1 detik ke komputer target (webserver). Setelah proses penyerangan, pada *database* snorby terkirim informasi seperti yang terlihat pada Gambar 10.

type	number_of_events	timestamp	id	ip_src	ip_dst	sig_priority	sig_name
1	0	2018-06-07 20:19:30	7310	303174146	320017154	3	Ping Of Death Attacks

Gambar 10. Hasil alert ping of death didalam database

Berdasarkan Gambar 10 dapat dilihat bahwa *alert* suricata berhasil dikirimkan ke dalam *database* snorby. Pembuktian bahwa pesan / informasi yang masuk kedalam *database* sesuai dengan *alert* yang dihasilkan suricata dapat kita buktikan dengan mengecek isi *alert* yang terdapat pada *file* eve.json, fast.log.

Supaya sistem dapat mengirimkan pesan pemberitahuan ke handphone administrator maka dijalankan *service* kirimpesan.sh. Gambar 11 memperlihatkan script untuk pengambilan informasi keterangan serangan, ip penyerang, ip korban, waktu serangan dari *database* snorby yang akan dikirimkan ke *handphone* administrator.



Gambar 11. Script pengambilan informasi dari database snorby (kirimpesan.php)

Gambar dibawah ini memperlihatkan *script* untuk mengirimkan informasi yang diambil dari *database* untuk dikirimkan ke *handphone* administrator melalui telegram.

Gambar 12. Script pengiriman pesan ke *handphone* melalui telegram (kirimpesan.php)

Script pengambilan informasi dari *database* dan skrip untuk mengirimkan pemberitahuan ke *handphone* melalui telegram terdapat pada *file* kirimpesan.php. Untuk mengeksekusi skrip kirimpesan.php dibutuhkan sebuah skrip shell yang mengulang eksekusi skrip tersebut, sebagai berikut :

Gambar 13. Script eksekusi kirimpesan.sh

Skrip shell kirimpesan.sh ini berfungsi untuk mengulang *command* php kirimpesan.php interval waktu perulangan

command selama 10 detik. Hasil yang diperoleh dari eksekusi *file* kirimpesan.php dapat dilihat pada Gambar 14.

Gambar 14. Tampilan pesan pada *handphone* saat terjadi ping of death

Dari gambar dapat dilihat bahwa pesan telah berhasil terkirim ke *handphone* melalui pesan instan telegram dengan selang waktu 10 detik setelah *alert* yang dihasilkan suricata berhasil masuk kedalam *database* snorby.

IV. KESIMPULAN

Skrip pengoperasian d_start pada IDS suricata sudah berhasil berjalan dengan baik sebagaimana yang diharapkan. Suricata dapat mendeteksi serangan-serangan yang masuk pada sistem dan mengirimkan *alert* nya kedalam *database* snorby. Skrip pengoperasian d_start pada snorby sudah berhasil berjalan dengan baik sebagaimana yang diharapkan, karena sudah dapat menampilkan web *interface* snorby ketika diketikkan alamat ip IDS suricata dengan port 3000 pada web browser.

Service barnyard2 telah bekerja dengan baik sebagaimana yang diharapkan karena barnyard2 sudah dapat membaca informasi yang ada pada *file* log unified2 dan mengirimkannya ke *database* snorby.

Informasi yang terkirim kedalam *database* snorby relevan dengan informasi serupa yang terdapat pada *file* fast.log dan *file* eve.json yang juga merupakan *file* untuk menampung *alert* yang dihasilkan suricata setelah berhasil mendeteksi serangan yang masuk.

Pesan pemberitahuan kepada administrator yang terdiri dari keterangan serangan, ip penyerang, ip target, dan waktu penyerangan telah berhasil terkirim ke handphone administrator melalui pesan instan telegram setelah 10 detik setelah *alert* yang dihasilkan suricata berhasil masuk kedalam *database* snorby. Dengan adanya sistem ini (yang terdiri dari IDS Suricata, Snorby, Barnyard2 dan Telegram) Administrator dapat memonitor keamanan pada jaringan yang dikelolanya melalui web *interface* snorby dan pesan pemberitahuan yang dikirimkan melalui telegram.

REFERENSI

[1] Shaik Akbar, Dr.K.Nageswara Rao, Dr.J.A.Chandula "Intrusion Detection System Methodologies Based on Data Analysis", international Journal of Computer Application(0975-8887) Volume 5-no.2, August 2010.

[2] Balaji Darapareddy and Vijayadeep Gummadi, "An Advanced Honeypot System for Efficient Capture and Analysis of Network Attack Traffic", International Journal of Engineering Trends and Technology- vol. 3, no. 5, pp.616-621, 2012.

[3] Sofyan Hadi, Periyadi,ST., M.T., Anang Sularsa, S.T., M.T. "Implementasi Network Intrusion Detection System pada Sistem Smart Identification", e-Proceeding of Applied Science – vol.2, No.3 December 2016.

[4] Park Wohyung, Ahn Seongjin., "Performance Comparison and Detection Analysis in Snort and Suricata Environment", International Journal Wireless Pers Common DOI 10.1007/s11277-016-3209-9, Springer Science, New York 2016

[5] OISF, "Suricata User Guide Release 4.0.0-dev", Suricata, July 18,2018

[6] Day, D.J. and B.M. Burns. A performance analysis of snort and suricata network intrusion detection and prevention engines. in The Fifth International Conference on Digital Society. 2011.

[7] Wibowo, R.A., "Analisis dan Implementasi IDS Menggunakan Snort pada Cloud Server di Jogja Digital Valley", Naskah Publikasi, Jurusan Teknik Informatika SMIK AMIKOM Yogyakarta, Yogyakarta, 2014.

[8] Forensic Wiki. "Barnyard2", Forensicwiki.org, 2013. https://www.forensicswiki.org/wiki/Barnyard2 (Di akses terakhir : 6 Juni 2018,12.45).

[9] Shivangi Shandilya, "Shell Scripting And Shell Programming In Unix", International Journal Of Innovative Research In Technology (IJIRT 101640), 2014.

[10] Moch Fajar, "Pengantar Pemrograman Bash Shell di Linux". Linux.or.id 2002. http://pemula.linux.or.id/programming/bash-shell.html (Di akses terakhir : 29 Agustus 2018, 21.53).

[11] Bernaeth, Nicolas., "Debian - Send your Server Notifications thru Telegram", Dyndns.org. http://bernaerts.dyndns.org/linux/75-debian/351-debian-send-telegram-notification (Di akses terakhir : 26 Okt 2017, 6:51).

[12] Hadil Deekshith., "Get Server Notification on Telegram App", Assistanz.com. https://www.assistanz.com/get-server-notification-telegram-app/ (Di akses terakhir : 27 Agustus 2018, 22:30).