Pengembangan Sistem Keamanan Jaringan Komputer Melalui Perumusan Aturan (*Rule*) *Snort* untuk Mencegah Serangan *Synflood*

Nori Sahrun
Jurusan Teknologi Informasi
STIE Riau
Norisahrun84@gmail.com

Rusdianto Roestam
Jurusan Teknologi Informasi
Universitas Bina Nusantara
rroestam@gmail.com

Sarjon Defit
Jurusan Teknologi Informasi
UPI, YPTK Padang
sarjond@yahoo.co.uk

Abstrak

Rule snort merupakan database yang berisi pola-pola serangan signature jenis serangan yang disusun sesuai dengan perintah-perintah snort. Rule snort ini, harus di update secara rutin supaya ketika ada sesuatu teknik serangan yang baru maka serangan tersebut dapat terdeteksi, dan program dalam penelitian ini yang akan mengupdate rule snort tersebut dalam mencegah serangan SYNflood. Dalam penulisan rule snort terdapat aturan-aturan yang harus di ikuti yaitu pertama rule snort harus ditulis dalam satu baris (single line), dan yang kedua snort terbagi menjadi dua bagian yaitu rule header dan rule option. Rule header berisi tentang rule action, protocol, source dan destination IP address,netmask, source dan destination port. Rule option berisi alert message dan berbagai dan berbagai informasi dimana seharusnya paket tersebut diletakkan. Dalam pengembangan keamanan jaringan sangat penting untuk di rumuskan serangan-serangan yang akan mengakibatkan system down dapat diatasi oleh rule terbaru.

Kata Kunci : Snort Rule, Serangan SYNflood, Rule Option, IP address

1. Pendahuluan

Serangan SYN-Flooding menjadi riset utama dari sisi pendeteksian dan pencegahan. Karena akan semakin kompleksnya administrasi dari jaringan skala luas (WAN) maka diperlukan suatu mekanisme keamanan dan metode untuk dapat mengoptimaan sumber daya jaringan tersebut, semakin besar suatu jaringan maka makin rentan terhadap serangan dan semakin banyak vurnability yang terbuka. Serangan DOS sudah berkembang menjadi serangan yang terdistribusi yang biasa disebut DDOS (Distributed Denial Of Services). Para penyerang (hacker) dapat melakukan serangan DOS lebih banyak lagi dengan zombie host (komputer yang sudah di injeksi dengan script pengontrol jarak jauh / botnet) pada target secara terdistribusi dan bersamaan sehingga efek dari serangan ini adalah dapat melumpuhkan target dengan cepat. Serangan DDoS memiliki tiga karakteristik utama: (1) jumlah sumber serangan adalah raksasa tapi lalu lintas menyerang individu sedikit, (2) trafik penyerang sering menyerupai lalu lintas yang sah dan (3) pola serangan akan dicampur untuk menyalakan serangan yang nyata. Adapun macam-macam serangan DDOS attack yang sering digunakan para hacker untuk menyerang yaitu SYN-Flooding, SMURF Attack, UDP-Flooding, ICMP-Flooding, DNS-Flooding. Berujuk pada survey dari lembaga Arbor'S pada tahun 2008, serangan SYN-Flooding diklasifikasikan sebagai serangan terbesar pada tahun tersebut yang menyerang situs-situs pemerintahan dan 76% diantaranya SYN-Flooding.

Dengan membuat berbagai rules untuk mendeteksi ciri-ciri khas (*signature*) dari berbagai macam serangan, maka Snort dapat mendeteksi dan melakukan *logging* terhadap serangan-serangan tersebut. Pengguna dapat membuat berbagai rules baru untuk mendeteksi tipe-tipe serangan yang baru. Selain itu, berbagai *rules* khusus dapat dibuat untuk segala macam situasi. *Snort* sudah memiliki sebuah database untuk berbagai macam *rules*, dan database ini secara aktif terus dikembangkan oleh komunitas *Snort* sehingga tipe-tipe serangan yang baru dapat dideteksi dan dicatat. Dengan latar belakang diatas dalam penelitian ini akan dirancang sebuah pengembangan sistem keamanan jaringan komputer melalui perumusan aturan (*rule*) *snort* untuk mencegah serangan *synflood*.

2. Tinjauan Pustaka

2.1. Jaringan Komputer

Menurut Ruri Hartika Zain dan Adhista Ricky Yatra (2012) Jaringan komputer adalah komunikasi data antar komputer, yaitu minimal 2 komputer.Jaringan komputer dapat dilakukan melalui media kabel ataupun nirkabel (*wireless*).jaringan komputer dilakukan melalui media kabel antara 2 komputer. *Interface* yang

digunakan adalah DB-25 atau *port paralel*. Data yang dikirimkan antar komputer adalah berupa kode biner 1 dan 0, yang dirancang pada masing-masing program yaitu pada program server dan program client.

2.2 Topologi Jaringan

Menurut (Bayoe, 2008) Topo*log*i adalah suatu cara menghubungkan komputer yang satu dengan komputer lainnya sehingga membentuk jaringan. Sebenarnya ada banyak topo*log*i jaringan komputer, namun yang sering didengar pada umumnya berkisar pada 3 bentuk (*Topology*) jaringan komputer yaitu : *Topology* Bus, *Token-Ring*, dan Star *Network*. Masingmasing topo*log*i ini mempunyai ciri khas, dengan kelebihan dan kekurangannya sendiri.

2.3 Media Tranmisi Komputer

Menurut Ruri Hartika Zain, S. Kom, M. Kom (2012:86) Media transmisi adalah media yang menghubungkan antara pengirim dan penerima informasi (data), karena jarak yang jauh, maka data terlebih dahulu diubah menjadi kode/isyarat, dan isyarat inilah yang akan dimanipulasi dengan berbagai macam cara untuk diubah kembali menjadi data. Media transmisi digunakan pada beberapa peralatan elektronika untuk menghubungkan antara pengirim dan penerima supaya dapat melakukan pertukaran data.

2.4 Keamanan Komputer

Menurut Tri Wahyu W dan Aidil Sanjaya (2008) *Computer Security* adalah bagian dari ilmu komputer yang bertugas untuk mengontrol resiko yang berhubungan dengan penggunaan komputer. *Computer Security* yang dimaksud adalah keamanan sebuah komputer yang terhubung ke dalam sebuah jaringan (Internet), dari akses yang tidak memiliki hak untuk mencoba masuk untuk memperoleh informasi dan service tertentu yang ada di dalam sistem. Usaha untuk mengakses paksa ini terdapat banyak macamnya, baik itu *Intrusion* (serangan dari luar organisasi) atau *misuse* (serangan dari dalam organisasi), dengan level *Hacker* (hanya mencoba masuk ke dalam sistem komputer) atau bahkan *cracker* (mencoba masuk dan merusak untuk keuntungan pribadi).

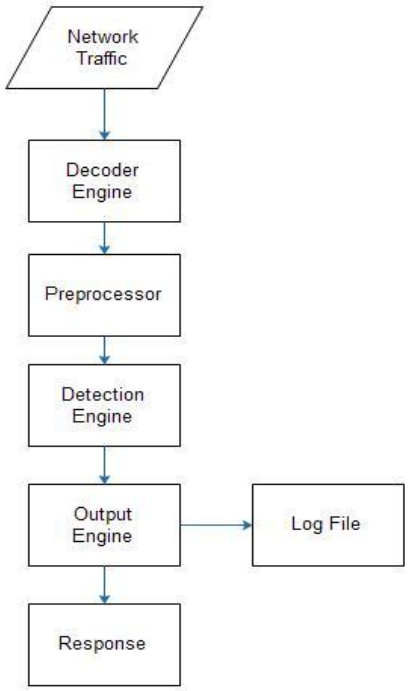
2.5 *IDS* (*Intrusion Detection System*)

Menurut Onno Purbo 2010, *IDS* adalah usaha mengidentifikasi adanya penyusup yang memasuki sistem tanpa otorisasi (misal *cracker*) atau seorang *user* yang sah tetapi menyalahgunakan (*abuse*) sumberdaya sistem.

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 (Wardhani, 2011).

2.6 Snort

Menurut Author Jacqueline (2014) *Snort* menggunakan himpunan yang berisi aturan-aturan yang diturunkan dari serangan-serangan yang ada atau hal lain yang dicurigai. Konten yang relevan pada untai diekstraksi dari header serangan yang diketahui. Distribusi *Snort* menggunakan sebuah himpunan aturan-aturan yang mampu meng-cover serangan-serangan yang telah dikenal seperti *buffer overflow* atau sebaran eksploitasi. Aturan-aturan ditambahkan ke *Snort* jika kelemahan tersebut ditemukan. Masing-masing aturan menyimpan konten, berasosiasi dengan aturan pada lokasinya, dan tipe dari paket yang ia kontrol. Kemudian, ada Penderajatan dari Basisdata Pendeteksian *Intrusi*. Himpunan aturan yang telah dipaparkan didukung oleh pendeteksian *Intrusi* dan disimpan dalam basisdata. Distribusi standar dari *Snort* melebihi 1500 aturan.



Gambar 1. *Flowchart* snort

2.7 Snort Rule

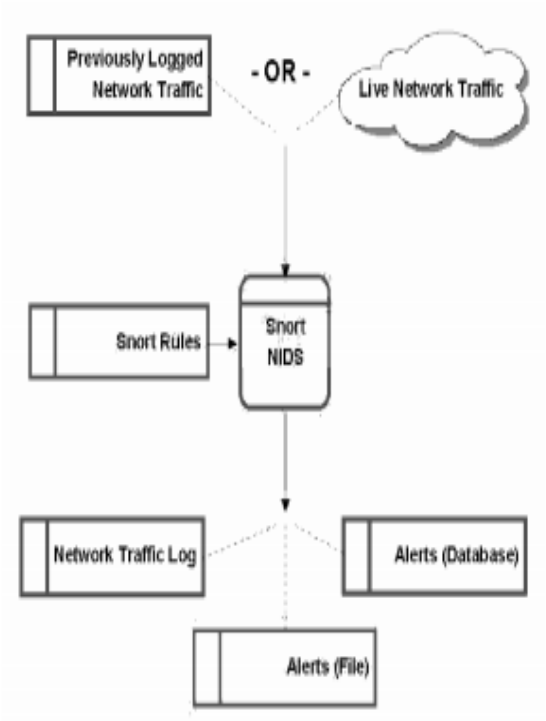
Rule Snort merupakan *database* yang berisi pola-pola serangan *signature* jenis serangan. *Rule Snort* ini, harus di *update* secara rutin supaya ketika ada sesuatu teknik serangan yang baru maka serangan tersebut

dapat terdeteksi, dalam tesis ini yang akan mengupdate *rule Snort* tersebut dalam mencegah serangan *SYNflood*. Dalam penulisan *rule Snort* terdapat aturan-aturan yang harus di ikuti yaitu pertama *rule Snort* harus ditulis dalam satu baris (*single line*), dan yang kedua *Snort* terbagi menjadi dua bagian yaitu *rule header* dan *rule option*. *Rule header* berisi tentang *rule action*, *protocol*, *source* dan *destination IP address* dan *netmask*, dan *source* dan *destination port*. *Rule option* berisi *alert message* dan berbagai dan berbagai informasi dimana seharusnya paket tersebut diletakkan.

Contoh *rule Snort*

Alert tcp any any ->192.168.1.0/24 111
(*content*:"ada serangan baru";*msg:"/bin/sh";)*

Karena pemahaman yang jelas tentang aturan *Snort* adalah penting untuk penelitian kami, rinci penjelasan berikut.



Gambar 2. Data flow diagram fleksibilitas snort

Snort tanda set, yang digunakan untuk mengidentifikasi pelanggaran keamanan, disebut Aturan *Snort*. Grup aturan *Snort* disebut sebagai file *.rules*, yang masing-masing dapat selektif termasuk ke dalam file konfigurasi *Snort*.conf. Sebuah file *.rule*s adalah polos file teks di mana setiap baris memiliki aturan tersendiri. Berikut catatan pada aturan *Snort* itu Format disatukan menggunakan Pengguna *Snort Manual*, untuk detail lengkap lihat *Snort* itu situs. Sebuah aturan secara formal didefinisikan sebagai ditunjukkan di bawah ini. Teks dalam <kurung siku> akan diganti dengan variabel wajib yang tepat, tanpa kurung siku hadir. Teks di [kurung] adalah opsional dan baik mewakili apa-apa atau mewakili teks itu sendiri, dalam kedua kasus tanpa tanda kurung persegi. <*Action* aturan> <protokol> [!] <Sumber ip> [!] <*Port* sumber> <Arah> < ip tujuan> <port tujuan> < aturan pilihan >

Contoh*: alert tcp any any -> any 25*
[Membuat peringatan untuk lalu lintas masuk kirim ke port 25]

Sebuah bentuk umum dari aturan snort ditunjukkan dibawah ini

<ruleaction> <protocol> <source ip> <source port> <direction> <dest.ip> <dest.port> <rule option>

Tabel 1 menunjukkan beberapa Aturan kami telah mengembangkan di Snort berdasarkan kondisi tertentu.

Table 1. Aturan *Snort Rule*

Kondisi	Solusi
Cara mengirim Siaga bila ada ping dengan TTL 254 ?	*Alert tcp any -> any any (msg:‖ping with ttl 254‖; any ttl:254;)*
Bagaimana menghasilkan Siaga ketika sedikit DF diatur dalam *icmp* paket ?	*Alert icmp any any any any(msg:‖alert don't fragment bit set‖;fragbits: D;)*
Bagaimana menghasilkan Siaga ketika paket ftp dari setiap IP dikirim ke 192.168.1.1 ?	*Alert IP any -> 192.168.1.1 any(msg: ―packet is 21 detected‖;)*
Bagaimana menghasilkan Siaga ketika sirip syn terdeteksi dalam paket ?	*Alert IP any any -> any any(msg:‖syn-fin detected‖;flags:SF;)*
Cara Log semua lalu lintas yang milik tertentu aplikasi?	*Alert tcp any any -> any any(msg:‖log created‖; session: all;)*
Bagaimana menghasilkan Siaga bila ada akses kesitus yang tidak sah ?	*Alert tcp any any < > 192.168.1.0/24 any (msg:‖unauthorized access‖;content-list:‖social‖; react:block;)* ***CONTENT-LIST*** *# social sites www.twitter.com #....*
Bagaimana menghasilkan Siaga ketika sebuah paket dari server telnet mengandung kata - *WHATSAPP*‖	*Alert tcp 192.168.1.0/24 23 -> any any (content:◊ "whatsapp"; msg: "Detected whatsapp";)*

Sesuai dengan aturan dalam membuat *rules* yang digunakan untuk mencegah serangan *SYNflood* sehingga dapat diterapkan didalam snort. Pemahaman

yang dapat diberikan *Snort Rule* yang disediakan bersifat default dan tidak di publikasikan bagaimana mengantisipasi serangan.

3. Hasil Dan Pembahasan

Dengan perumusan *rule snort* yang telah di lakukan dengan format :

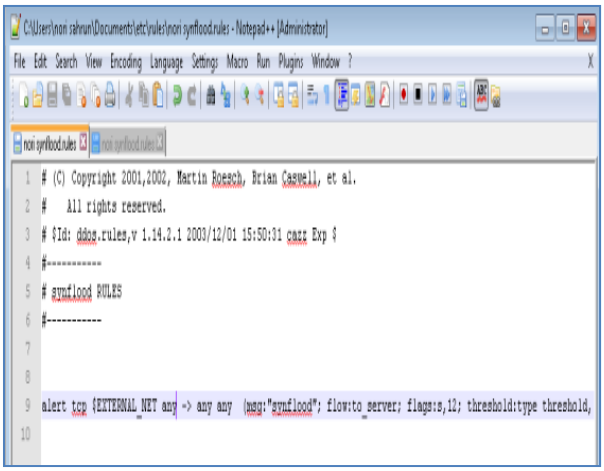
<Action aturan> <*protokol*> <Sumber ip> <*Port* sumber> <Arah panah> < ip tujuan> <*port* tujuan> <aturan pilihan >
<rule header> <rule option flow> <rule option flags> <rule option sid> <rule option rev> <rule option rule_filter> <rule gen_Id> <rule sig_id> <rule option count > <rule option seconds > <rule option new_action alert > <rule option timeout ><rule option gen_id 1> <rule option sig_id 0 > <rule option count > <rule option seconds >

Sesuai dengan ketentuan dalam perumusan *Snort Rule* maka diperoleh *Snort Rule* seperti dibawah ini:
alert tcp $EXTERNAL_NET any -> any any (msg:"DDOS synflood"; flow:to_server; flags:s,12; sid:1000015; rev: 1) rate_filter, gen_id 1, sig_id 1000015, count 3000, seconds 1, new_action alert, |drop|pass|log|sdrop|reject ,timeout 0 event_filter, gen_id 1, sig_id 0, count 1, seconds 20

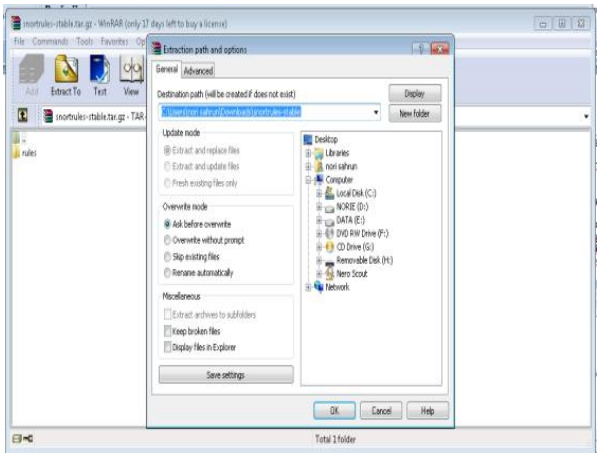
Dari *rule* diatas dengan penjelasan sebagai berikut:
Jenis_perintah *protocol* sumber *port*_sumber -> *network*_tujuan port_tujuan *<rule* menghasilkan peringatan dengan menggunakan metode peringatan yang dipilih , dan kemudian log paket header> *<rule* Hal ini memungkinkan aturan untuk hanya berlaku untuk *client* atau server *flow* > *<rule* memeriksa spesifik *bit flag TCP flags> <rule option* unik mengidentifikasi aturan *snort*. Informasi ini membantu keluaran *plug-in* untuk mengidentifikasi aturan mudah *sid> <rule* mengidentifikasi revisi aturan snort > *<rule option rate_filter* menyediakan pencegahan serangan tingkat berdasarkan dengan memungkinkan pengguna untuk mengkonfigurasi tindakan baru untuk mengambil untuk waktu tertentu ketika tingkat tertentu terlampaui *rule_filter> <rule gen_id* > Tentukan generator ID dari aturan terkait . gen_id 0, sig_id 0 dapat digunakan untuk menentukan ambang batas " global" yang berlaku untuk semua aturan *<rule sig_id > <rule option* jumlah aturan yang cocok dalam s detik yang akan menyebabkan batas *event_filter* untuk dilampaui. *count* harus menjadi nilai nol *Count > <rule option* periode waktu di mana perhitungan yang masih harus dibayar. s harus menjadi nilai nol *seconds > <rule option new_action* menggantikan tindakan aturan untuk t detik . turun , menolak , dan sdrop dapat digunakan hanya ketika mendengus digunakan dalam mode inline. sdrop dan menolak secara kondisional dikompilasi dengan Gids *new_action alert > <rule gen_id >*

Setelah hasil perumusan di dapatkan maka kita akan memasukkan kedalam server dengan langkah-langkah sebagai berikut :

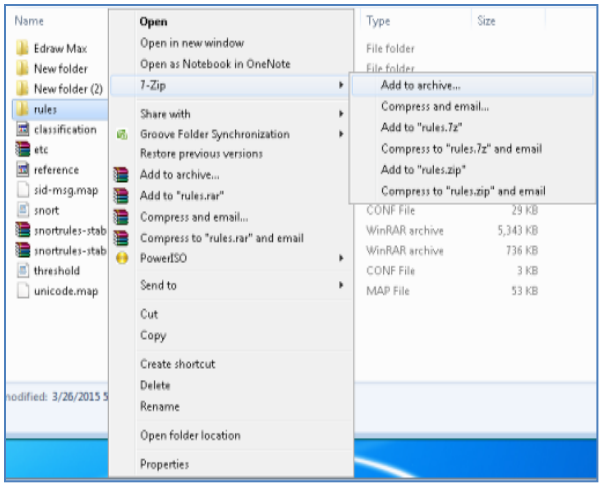
1. Melakukan download *snort rule* di www.*snort*.org dengan nama *snortrule* s-stable.tar.gz
2. *Snortrule s-stable.tar.gz* dilakukan *extract* keseluruhan data
3. Selanjutnya membuka dengan *Notepad++* untuk membuat *rule* baru dengan perumusan yang sudah di buat.
4. Setelah dibuat dengan format ".*rule* s" maka dapat di satukan dengan format *rule* yang lain.
5. Langkah selanjutnya melakukan instalasi 7-Zip agar dapat menggabungkan semua *rule* menjadi format .tar.gz
6. *Rule snort* yang telah dilakukan perumusan di lakukan format agar dapat berjalan di *system* sesuai dengan gambar 3 berikut :



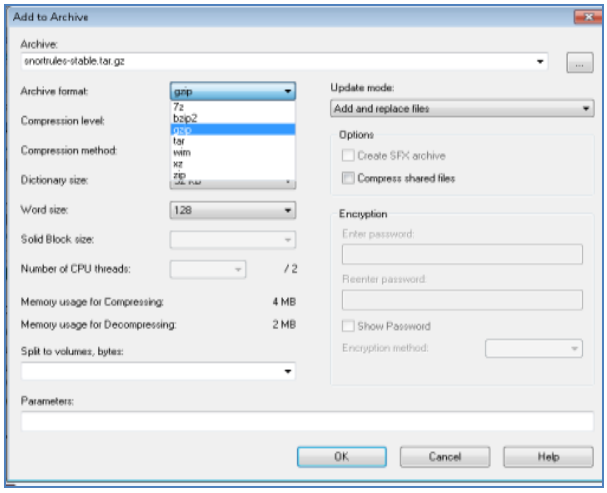
Gambar 3. *Rule snort* di masukkan ke dalam aplikasi *notepad++*



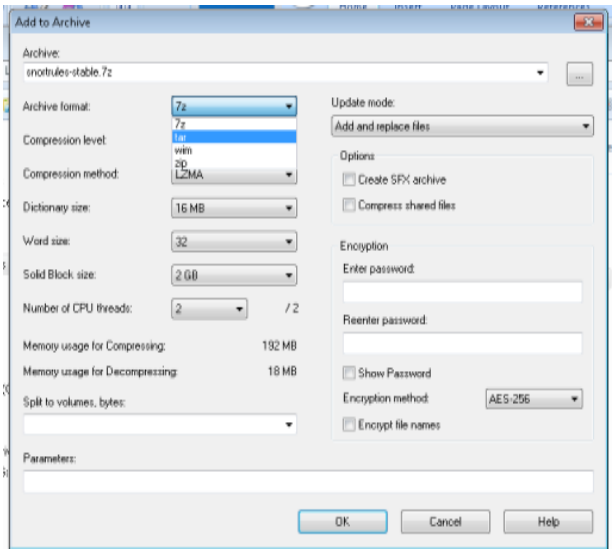
Gambar 4. Extract *snortrule* s-stable.tar.gz



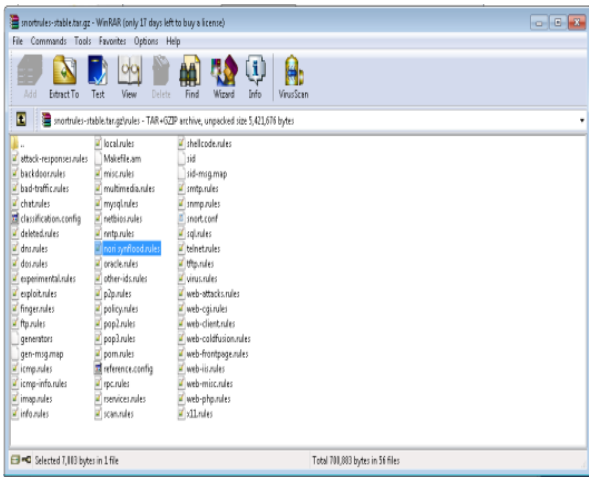
Gambar 5. Add to archive membuat format .tar



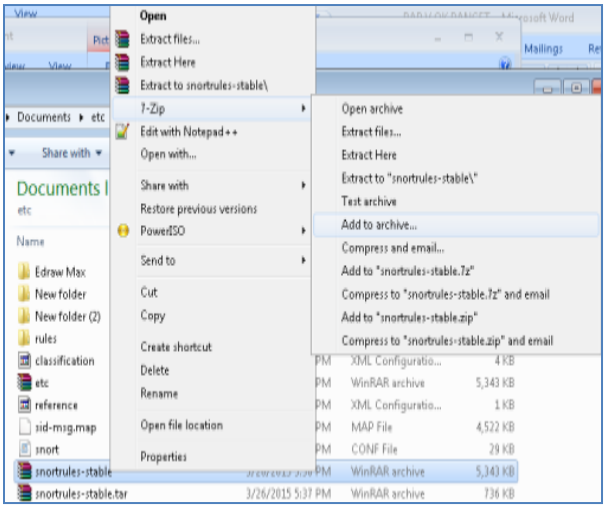
Gambar 8. Memilih format gzip



Gambar 6. Memilih format .tar di 7-Zip

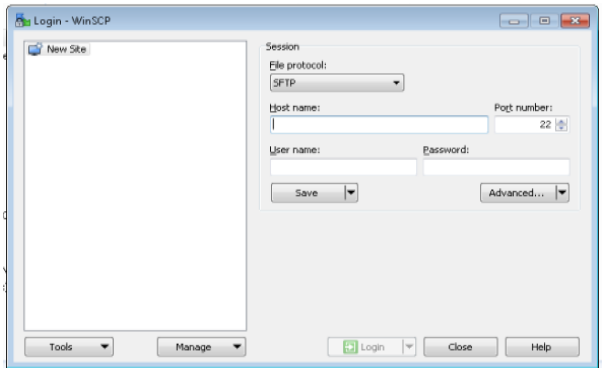


Gambar 9. R*ule* yang sudah dirumuskan telah di format .tar.gz



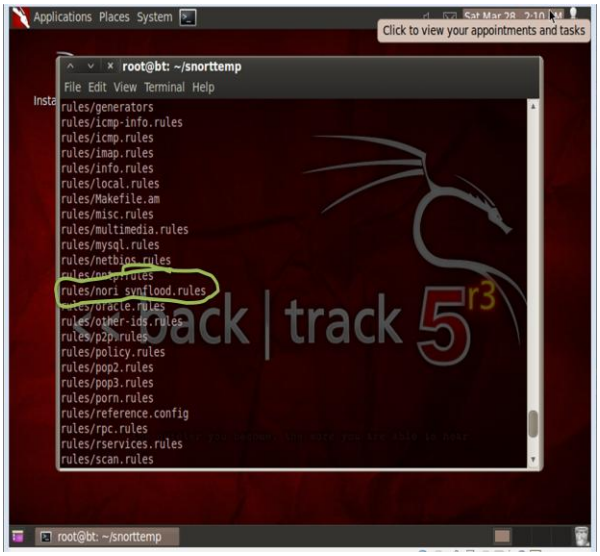
Gambar 7. Add to Archive ke format .tar.gz

7. Selanjutnya dilakukan format yang sudah ada di masukkan kedalam server yang penulis pilih yaitu Sistem Operasi Backtrack
8. Penulis menggunakan *Winscp* sebagai aplikasi memindahkan data *rule* yang sudah dirumuskan kedalam server



Gambar 10. Winscp untuk mengirim data *rule snort*

9. Selanjutnya dilakukan instalasi kedalam server dengan perintah " *tar* xvzf *snortrule s-stable.tar.gz* –C /usr/local/*snort* hasilnya *rule* sudah di masukkan kedalam server seperti gambar berikut



Gambar 11. R*ule snort* telah dimasukkan kedalam sistem

10. S*ystem* dengan *rule* yang sudah dirumuskan dapat mencegah serangan *SYNflood* sehingga server dapat bekerja dengan baik

3.1 Pengujian *system* dengan Serangan *SYNflood* Serangan SYN *flooding attack*

Menurut Sucipta, Wirawan, & Muliantara (2012), *Serangan Denial of Services (DoS)* adalah salah satu contoh jenis serangan yang dapat mengganggu infrastruktur dari jaringan komputer, serangan jenis ini memiliki suatu pola khas, dimana dalam setiap serangannya akan mengirimkan sejumlah paket data secara terus-menerus kepada target serangannya. Dengan menggunakan metode deteksi anomali, serangan *DoS* dapat dideteksi dengan mengidentifikasi pola-pola *anomali* yang ditimbulkan.

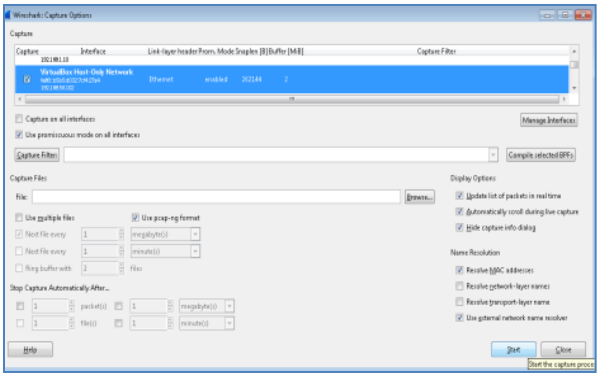
Gambar 12. *Low Orbit Ion Cannon*

Penulis menggunakan *Low Orbit Ion Cannon* adalah Sebuah aplikasi yang digunakan untuk menyerang server dengan *SYNflood* dengan memilih *port* yang akan diserang dan IP Address mana yang akan dilakukan serangan sehingga dapat membuat *system down*.

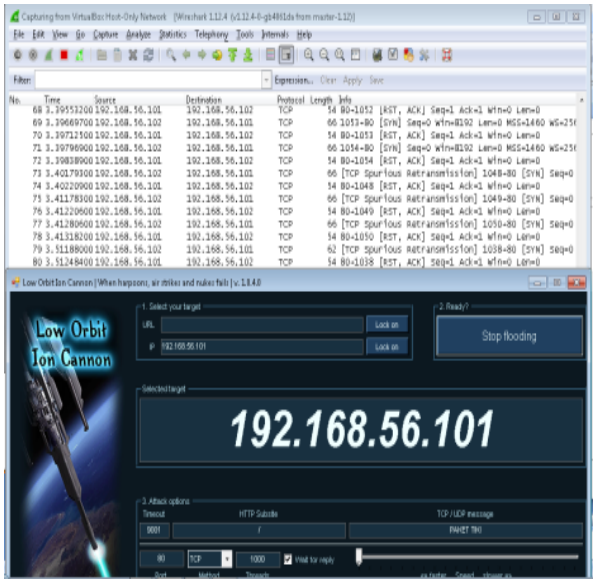
Setelah perumusan *snort rule* agar dapat hasil bahwa *rule* yang sudah di rumuskan dapat mencegah serangan *SYNflood* maka dilakukan proses pengujian, *client* yang bertindak sebagai penyerang mengirimkan paket-paket SYN ke dalam *port-port* yang sedang berada dalam keadaan terbuka yang berada dalam server target.

3.2 Monitoring jaringan

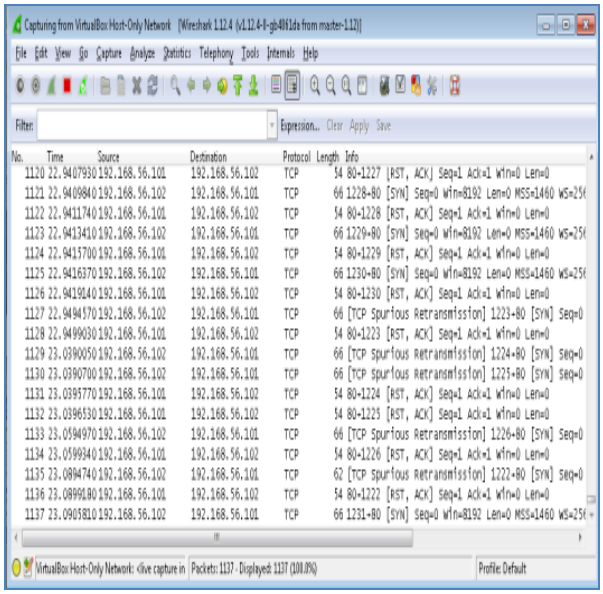
Setelah dilakukan pengujian dengan melakukan serangan *SYNflood* dapat di lihat traffic jaringan yang berhasil di blok oleh *system*. Tentunya *system* tidak dapat di tembus dan dapat di cegah dengan *rule* yang sudah dibuat.



Gambar 13. *Setting virtual LAN card*



Gambar 14. Serangan *SYNflood* dan monitoring dengan *wireshark*



Gambar 15. Hasil seluruh blok serangan *synflood* oleh *system*

6. Kesimpulan dan Saran

6.1 Kesimpulan

Berdasarkan pada hasil pengujian aplikasi yang memanfaatkan *snort rule* dari pengembangan system keamanan jaringan melalui perumusan *snort rule* mencegah serangan *SYNflood* berhasil di uji dan menghasilkan kondisi *snort rule* yang telah dibuat penulis mampu mengatasi serangan *SYNflood* lebih baik sehingga dengan fakta ini bisa disimpulkan :

1. Bahwa perumusan *Snort Rule* yang telah dikembangkan telah berfungsi dengan baik dalam mencegah serangan *SYNflood*
2. Memahami dan Mampu melakukan Perumusan *Snort rule.*
3. Bahwa perumusan *Snort Rule* yang telah dikembangkan telah berfungsi dengan baik dalam mencegah serangan S*YNflood.*

6.2 Saran

Snort rule yang dirumuskan masih belum terintegrasi dengan Aplikasinya, Sehingga perlu di kembangkan aplikasi yang terintegrasi dengan aplikasi perumusan *snort rule.* Saran Perbaikan adalah

1. *Snort rule* yang dirumuskan masih belum terintegrasi dengan aplikasinya, sehingga perlu di kembangkan aplikasi yang terintegrasi dengan aplikasi perumusan *snort rule*.
2. Aplikasi ini hanya mampu untuk mendeteksi serangan SYNflood, sehingga perlu dikembangkan *rule-rule* yang bisa mendeteksi serangan terhadap sistem komputer lainnya.

Referensi

[1] U Aickelin, J Twycross and T Hesketh Roberts (2010), *Rule Generalisation using Snort* Int. J. , Vol. 1, No. 1, 2010

[2] Sumiti Bansal, Mandeep Singh and Saurabh Mittal (2014), *Developing rules or Signatures to Detect Novel Attacks using open Source IDS: Snort* Volume 4, Issue 3, March 2014 ISSN: 2277 128X

[3] Alamsyah, (2011), *Implementasi Keamanan Instrusion Detection System (Ids) Dan Instrusion Prevention System (Ips) Menggunakan Clearos* Jurnal SMARTek, Vol. 9 No. 3. Agustus 2011

[4] Choirul Muallifah, Lies Yulianto**,** (*2013*), *Pembuatan Jaringan Local Area Network Pada Laboratorium MA Pembangunan Kikil Arjosari* IJNS – Indonesian Journal on Networking and Security - ISSN: 2302-5700

[5] Bosman Tambunan, Willy Sudiarto Raharjo, Joko Purwadi, (2013), *Desain dan Implementasi Honeypot dengan Fwsnort dan PSAD sebagai Intrusion Prevention System* ULTIMA Computing, Vol. V, No. 1 | September 2013

[6] FajarP. Pongsapan,Yaulie D.Y. Rindengan, XaveriusB.N. Najoan, (2014), *Desain Arsitektur Jaringan Teknologi Informasi dan Komunikasi untuk Manado Smart city; Studi Kasus Pemerintah Kota Manado* e-journal Teknik Elektro dan Komputer (2014), ISSN: 2301-8402

[7] Jutono Gondohanindijo, (2012), *IPS (Intrusion Prevention System) Untuk Mencegah Tindak Penyusupan / Intrusi* Majalah Ilmiah INFORMATIKA Vol. 3 No. 3, Sept. 2012

[8] Octavianus Wijaya, Jusak, Anjik Sukma aji, (2014), *Pemodelan Karakterisitik Denial Of Service Attack Melalui Analisis Data Trafik* Journal of Control and Network Systems JCONES Vol. 3, No. 1 (2014) 105-111

[9] Ruri Hartika Zain, S. Kom, M. Kom Adhista Ricky Yatra, (2012), *Aplikasi Pagar Elektrik Pada Keamanan Fasilitas Lembaga Permasyarakatan Dilengkapi Alarm Deteksi Pemutusan Arus Listrik Dan Sensor Menggunakan Jaringan Komputer* Vol.13 No.2. Agustus 2012 Jurnal Momentum ISSN : 1693-752X

[10] Warsito, Bekti Ratna Timur Astuti, (2013), *Perancangan Dan Instalasi Jaringan Local Area Network Sekolah Menengah Kejuruan Muhammadiyah Enam Gemolong Sragen* Indonesian Jurnal on Networking and Security (IJNS) - ijns.org

[11] Zaid Amin, (2012), *analisis vulnerabilitas host pada keamananJaringan komputer di pt. Sumeks tivi palembang (paltv) menggunakan router berbasis unix* Jurnal teknologi dan informatika (teknomatika) vol. 2 no. 3 sept 2012

[12] Yudha kristanto , Muhammad Salman, ST, MIT, (2011), *Implementasi Dan Analisa Unjuk Kerja Keamanan Jaringan Pada Infrastuktur Berbasis Idps (Intrusion Detection And Prevention Sistem***)** Fakultas Teknik Program Studi Teknik Komputer Universitas Indonesia

[13] Alva S. M. Tumigolung, Arie S. M. Lumenta, Arthur M. Rumagit, (2015), *Perancangan Sistem Pencegahan Flooding Data Pada Jaringan Komputer* E-Journal Teknik Elektro dan Komputer (2015), ISSN : 2301-8402