DETEKSI PENYUSUPAN PADA JARINGAN KOMPUTER MENGGUNAKAN IDS SNORT

Walid Fathoni[1], Fitriyani[2], Galih Nugraha Nurkahfi[3]

[1,2,3]Prodi Ilmu Komputasi, Fakultas Informatika, UniversitasTelkom, Bandung

[1]walidfathoni@live.com, [2]Fitriyani@telkomuniversity.ac.id, [3]galihnugrahanurkahfi@gmail.com

Abstrak

Masalah keamanan jaringan semakin menjadi perhatian dikarenakan semakin banyaknya alat yang bermunculan dan tehnik yang digunakan oleh seseorang secara ilegal untuk masuk kedalam sistem atau membuat lumpuh sistem yang ada secara ilegal. Selain itu adanya celah dan tidak ada sistem keamanan yang melindungi sistem menjadikan sistem rentan terhadap serangan.

Tugas akhir ini disusun untuk melakukan penelitian terhadap beberapa jenis serangan yang ada dan sering terjadi, sehingga dapat membantu menangkal serangan yang dilakukan hacker terhadap sistem. Penulis menggunakan software SNORT dalam penelitiannya dan berdasarkan rule berbasis signature base. Pada penelitian ini penulis berhasil mendeteksi semua serangan yang diujicobakan dan menghasilkan nilai 1 yang berarti pendeteksian berjalan dengan baik.

Kata kunci : Keamanan jaringan, kerentanan, sistem deteksi penyusupan, SNORT, IDS

Abstract

Network security issues is increasingly becoming a concern due to the increasing number ofemerging tools and techniques used by someone illegally to entered into the system or paralyze the existing system illegally. Besides gaps and no security system that protects the system makes the system vulnerable to attack.

The final task is structured to conduct research on some kind of attack that is often the case, so it can help ward offhacker attacks carried out against the system. Snort software author uses in his research and is based on signature-based rule. In this study, the authors successfully detect all attacks that tested and the value is 1, means the detection goes well.

1. Pendahuluan [10 pts/Bold]

Seiring dengan kemajuan teknologi informasi (IT) yang telah banyak diterapan oleh hampir semua kalangan munculah permasalahan tentang keamanan dari sistem IT, agar data atau informasi yang ada didalam sistem tidak bisa diakses oelh orang yang tidak berkepentingan, dan bagai mana agar sistem tersebut terhindar dari tindakan pengerusakan.

Untuk mengamankan teknologi IT ada banyak caranya, salah satunya dengan mengamankan sisi jaringanya. Salah satu cara yang dapat digunakan adalah dengan menerapkan sistem pendeteksian penyusupan jaringan atau *intrusion detection system* (IDS) [1].

IDS adalah sebuah operasi untuk mendeteksi adanya ancaman yang terjadi pada jaringan komputer seperti pencurian data, informasi dan pengerusakan sistem oleh hacker. Pada konsep kerjanya IDS akan mencoba mendeteksi adanya tindakan ancam yang ada dan segera mengirimkan informasi pada administrator jaringan[1].

IDS akan mendeteksi ancaman dengan meilihat dan mengevaluasi koneksi TCP/IP. Ketika IDS mendeteksi adanya ancaman maka IDS akan segera meberikan peringan dan membuat catatan. Catatan tersebut mengandng informasi tentang ancaman yang terdeteksi seperti alamat asal dan tujuan target dan memberitahukan jenis serangan yang terdeteksi[1].

Salah satu aplikasi berbasis IDS yang dapat digunakan dalam pengamanan jaringan adalah SNORT [1]. SNORT adalah software open-source yang bebas digunakan dan dimodifikasi sesuai dengan kebutuhan.

Sesuai dengan penjelasan yang telah dijabarkan sebelumnya , penelitian ini bertujuan untuk membuat sebuah sistem pendeteksian ancaman yang terjadi pada jaringan komputer menggunakan salah satu *software* IDS yaitu SNORT.

2. Landasan Teori

2.1. Intrusion Detection System (IDS)

IDS adalah tehnik yang digunakan unutk mendeteksi adanya aktifitas mencurigakan pada jaringan komputer pada leve *network* dan *host.* IDS akan mengumpulan data – data dari sensor yang telah ada sebelumnya, apabila ada aktifitas

mencurigakn IDS akan memberikan peringatan dan melakukan pencatatan.

IDS mendeteksi adanya aktifitas mencurigakan berdasarkan signature base dan anomali base. Signature base dapat diumpamakan seperti virus yang memiliki ciri khas, sedangan anomali base berdasakan kejanggalan yang terjadi pada pola lalulintas jaringan yang diawasi.

Dalam pengaplikasiannya IDS dapat menjadi *Network Intrusion detection System* (NIDS) yang akan melakukan pengawasan pada jaringan komputer yang diawasi. Pengaplikasian keduanya *Host intrusion detection system* (HIDS), IDS akan melakukan pengawasan terhadap sistem tertentu saja, ini merupakan desain pertama pada IDS.

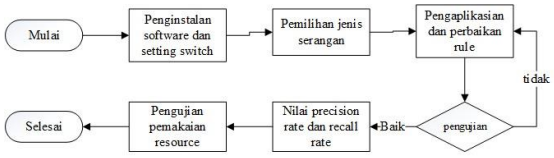
2.2. SNORT

Snort salah satu software keamanan jaringan berbasis IDS. Snort bersifat open-source sehingga boleh dimodifikasi sesuai dengan kebutuhan. Dalam pengoprasiannya Snort dapat berjalan pada mode *Packet Sniffer*, *packet logger* dan *Network Intrusion Detection*. Pada mode sniffer Snort hanya akan melihat jaringan yang sedang diawasi tanpa melakukan apapun, pada mode *packet logger* Snort akan mencatat semua paket yang lewat dan pada *Network Intrusion Detection* Snort hanya akan mendeteksi adanya tindakan ancaman sesuari dengan setting dan rule yang diimplementasikan, pencatatan hanya dilakukan hanya ketika Snort mendeteksi adanya tindakan ancaman.

3. Perancangan Sistem

Rancangan umum pada sistem ini adalah sebagai berikut :

1. Penginstallan software dan configurasi switch.
2. Pemilihan jenis serangan yang akan diujicobakan.
3. Pengaplikasian dan perbaikan pada rule
4. Melakukan pengujian rule terhadap serangan yang dilakukan.
5. Perhitungan precision rate dan recall rate.
6. Perhitungan penggunaan resource mesin IDS



Gambar 3.1 alur pengerjaan sistem

4. Analisi pengujian sistem

Pada bagian ini yang pertama dilakukan adalah melakukan sejumlah serangan yang telah dipilih selanjutnya, kemudian melihat reaksi IDS dalam menangkap adanya tindakan penyusupan. Apabila kemampuan IDS dalam menangkap adanya ancaman pada jaringan komputer dirasa belum memuaskan, maka dilakukan lagi pengujian dengan sebelumnya melakukan tuning *rule* yang telah diimplementasikan. Apabila Snort menghasilkan terlalu bnyak false posive walaupaun dapat menangkap semua serangan yang dilakukan, maka kembali kita akan melakukan *tuning* pada *rule*.

5. Hasil pengujain sistem.

5.1. Perhitungan recall rate dan precision rate

Pada tahap ini adalah pemaparan hasil pengujian yang telah dilakukan sebelumnya. Hasil tiap pengujian yang telah dilakukan serta hasil perbandingan *rules* yang telah diimplementasikan dengan rule yang ditemukan sebelumnya.

Peringatanyang dihasilkan oleh rules yang diimplementasikan akan dihitung berdasarkan precision rate (menentukan adanya false positive) dan recall rate (menentukan adanya false negative). Berikut ini adalah rumus precision rate dan recall rate,

$$= \frac{\text{[illegible]}}{\text{[illegible]} + \text{[illegible]}}$$

$$= \frac{\text{[illegible]}}{\text{[illegible]} + \text{[illegible]}}$$

Dan berikut ini adalah peringantan yang dihasilkan dari pengimplementasian rule yang digunakan,

1. SQL Injection

   True Positive = 585 SQL *injection* terdeteksi
   False Positive = 0
   False Negative = 0
2. Cross Site Scriping (XSS)

   True Positive = 6 *Cross Site Scripting* terdeteksi
   False Positive = 0
   False Negative = 0
3. Denial of Service (DoS)

   True Positive = 1575 *Denial of Service* terdeteksi
   False Positive = 0
   False Negative = 0

4. SSH Brute Force

True Positive = 30 SSH *Brute Forece* terdeteksi
False Positive = 0
False negative = 0

Berdasarkan rumus tersebut , semua rules yang diimplementasikan menghasilkan nilai precision rate 1 dan nilai recall rate 1. Ini berarti bahwa rule yang diimplementasikan tidak menghasilkan false positive dan false negative.

5.2. Perbandingan rule yang diimplementasikan dengan rule yang telah ditemukan sebelumnya.

Berikut ini adalah perbandingan antara rule yang diimplementasikan dengan rule yang telah penulis temukan sebelumnya, dengan pengertian bahwa rule nomer 1 adalah rule yang diimplementasikan.

1. SQL Injection

Tabel 5.1 Perbandingan rule SQL Injection

rules	Error/double	Blind	String	Union
1	Iya	Iya	Iya	Iya
2	Tidak	Tidak	Tidak	Iya
3	Tidak	Tidak	Tidak	Iya
4	Tidak	Iya	Tidak	Iya
5	Tidak	Tidak	Tidak	Tidak
6	Tidak	Tidak	Tidak	Iya
7	Tidak	Tidak	Tidak	Iya
8	Tidak	Tidak	Tidak	Iya
9	tidak	Tidak	Tidak	Tidak

2. Cross Site Scripting

Tabel 5.2 Perbandingan rule Cross Site Scripting

rules	Cross Site Scripting
1	Iya
2	Tidak
3	Tidak

3. Denial of Service (DoS)

Tabel 5.3 Perbandingan rule Dos

Rule	Loic	Hping 3	
		Rand Source	*Same Source*
1	Ya	Ya	Ya
2	Tidak	Tidak	Ya

4. SSH Brute Force Attack

Tabel 5.4 perbandingan rule SSH Brute Force

Rule	Hydra	Medusa	Keterangan
1	Iya	Iya	Tidak ada false
2	Iya	Iya	Terdapat false

Dapat dilihat dari tabel perbandingan tersebut rule yang diimplementasikan memiliki kinerja lebih baik dari pada rule yang telah penulis temukan sebelumnya terlihat dengan dapat mendeteksi semua serangan yang diujicobakan.

5.3. Perbandingan penggunaan resource.

Pada bagian ini pengujain dilakukan dengan membandingakan penggunaan CPU dan RAM pada mesin IDS ketika mesin menyal tanpa mendeteksi adanya ancaman dan ketika Snort mendeteksi adanya ancaman yang terjadi pada jaringan komputer. Berikut ini adalah pebandingan yang dapat dilihat pada tabel 5.5,

Tabel 5.5 perbandingan penggunaan resource mesin IDS

No	Kondisi	CPU	RAM
1	Awal tanpa ada ancaman	3.4 %	1418936 KiB
2	Snort mendeteksi ancaman	52 %	3181080 KiB

Pada tabel diatas kenaikan terjadi sebesar 48,6 % pada CPU dan kenaikan sebesar 1763144 KiB.

6. Analisis secara keseluruhan

Kinerja Snort sangat baik pada pengujian yang dilakuan , dapat dilihat dari hasil precision rete dan recall rate yang bernilai 1. Tapi peningkatan penggunaan resource pada mesin IDS menjadi permasalahan ketika pengujian akan dilakukan pada jaringan yang lebih besar lagi. Hal ini terkat pada kemampuan mesin mengola data – data yang masuk dalam jumlah yang banyak dan cepat.

7. Kesimpulan

Berdasarkan pengujian dan analisis yang telah dilakukan dapat diambil kesimpulan sebagai berikut :

1. Pengujian dan implementasi *rule* yang digunakan dapat mendeteksi semua jenis serangan yang diujikan.
2. Kemampuan tiap *rule* dalam mendeteksi serangan berdasarkan *precision rate* dan *recal rate* menghasilkan nilai 1 pada setiap jenis serangan yang diujikan. Ini menunjukan *rule* dapat mendeteksi 100% ancaman yang diujikan dan tidak menghasilkan *false positive*.
3. Peringatan yang dihasilkan oleh Snort ditunjukkan dalam bentuk web yang dapat dilihat detail dari tiap serangan yang dihasilkan.
4. Perbandingan penggunaan resource pada saat awal dan seteleh pengujian menghasilkan selisih kenaikan 48,6 % CPU dan 1763144 KB RAM.

8. Daftar Pustaka

[1] Anonim. (2014). *http://manual.snort.org/node2.html*. (Cisco, Editor, & Cisco) Dipetik 02 07, 2014, dari Snort User Manual 2.9.6: http://manual.snort.org/

[2] Deuble, A. (2012). *Detecting and Preventing Web Application Attacks with Security Onion*.

[3] Dietrich, N. (2015). Snort 2.9.7.x on Ubuntu 12 and 14. *with Barnyard2, PulledPork, and BASE*.

[4] Hussein AlNabulsi, Izzat Alsmadi, and Mohammad Al-Jarrah. (2014). I.J. Computer Network and Information Security. *Textual Manipulation for SQL Injection Attacks*, 26-33.

[5] Mohammad Dabbour, Izzat Alsmadi and Emad Alsukhni. (2013). Efficient Assessment and Evaluation for Websites Vulnerabilities Using Snort. *International Journal of Security and Its Applications*.

[6] Northcutt, S. (2004). *Intrusion Detection, Second Editon.* United States of America: Syngress Publising, Inc.

[7] Rafeeq, R. U. (2003). *Inrusion Detection with SNORT: Advanced IDS Techniques Using SNORT, Apache, MySql, PHP, and ACID.* New Jersey: Pearson Education, Inc.

[8] Rafudin, R. (2010). *Mengganyang Hacker degan SNORT*. Yogyakarta: C.V Andi Offset.

[9] Aninom. (2015, juni 11). *http://www.computersecuritystudent.com/cgi-bin/CSS/process_request_v3.pl?HID=688b0913be93a4d95daed400990c4745&TYPE=SUB*. Diambil kembali dari Computer Security Student (CSS): http://www.computersecuritystudent.com

[10] Deuble, A. (2012). *Detecting and Preventing Web Application Attacks with Security Onion*.

[11] ZentrixPlus. (2015, juni 27). Diambil kembali dari ZentrixPlus: http://zerofreak.blogspot.co.id/