View metadata, citation and similar papers at core.ac.uk
brought to you by CORE
provided by Crossref

IJCCS, Vol.8, No.1, January 2014, pp. 13~24
ISSN: 1978-1520

Penerapan Metode *Support Vector Machine* pada Sistem Deteksi Intrusi secara *Real-time*

Agustinus Jacobus*[1], Edi Winarko[2]
[1]Program Studi S2 Ilkom, FMIPA, UGM, Yogyakarta
[2]Jurusan Ilmu Komputer dan Elektronika, FMIPA UGM, Yogyakarta
e-mail: ***[1] a.jacobus@mail.ugm.ac.id**, [2]ewinarko@ugm.ac.id

Abstrak

Sistem deteksi intrusi adalah sebuah sistem yang dapat mendeteksi serangan atau intrusi dalam sebuah jaringan atau sistem komputer, umum pendeteksian intrusi dilakukan dengan membandingkan pola lalu lintas jaringan dengan pola serangan yang diketahui atau mencari pola tidak normal dari lalu lintas jaringan. Pertumbuhan aktivitas internet meningkatkan jumlah paket data yang harus dianalisis untuk membangun pola serangan ataupun normal, situasi ini menyebabkan kemungkinan bahwa sistem tidak dapat mendeteksi serangan dengan teknik yang baru, sehingga dibutuhkan sebuah sistem yang dapat membangun pola atau model secara otomatis.

Penelitian ini memiliki tujuan untuk membangun sistem deteksi intrusi dengan kemampuan membuat sebuah model secara otomatis dan dapat mendeteksi intrusi dalam lingkungan real-time, dengan menggunakan metode support vector machine sebagai salah satu metode data mining untuk mengklasifikasikan audit data lalu lintas jaringan dalam 3 kelas, yaitu: normal, probe, dan DoS. Data audit dibuat dari preprocessing rekaman paket data jaringan yang dihasilkan oleh Tshark.

Berdasar hasil pengujian, sistem dapat membantu sistem administrator untuk membangun model atau pola secara otomatis dengan tingkat akurasi dan deteksi serangan yang tinggi serta tingkat false positive yang rendah. Sistem juga dapat berjalan pada lingkungan real-time.

Kata kunci— *deteksi intrusi, klasifikasi, preprocessing, support vector machine*

Abstract

Intrusion detection system is a system for detecting attacks or intrusions in a network or computer system, generally intrusion detection is done with comparing network traffic pattern with known attack pattern or with finding unnormal pattern of network traffic. The raise of internet activity has increase the number of packet data that must be analyzed for build the attack or normal pattern, this situation led to the possibility that the system can not detect the intrusion with a new technique, so it needs a system that can automaticaly build a pattern or model.

This research have a goal to build an intrusion detection system with ability to create a model automaticaly and can detect the intrusion in real-time environment with using support vector machine method as a one of data mining method for classifying network traffic audit data in 3 classes, namely: normal, probe, and DoS. Audit data was established from preprocessing of network packet capture files that obtained from Tshark.

Based on the test result, the system can help system administrator to build a model or pattern automaticaly with high accuracy, high attack detection rate, and low false positive rate. The system also can run in real-time environment.

Keywords— *intrusion detection, classification, preprocessing, support vector machine*

Received December 1st,2013; Revised January 1st, 2013; Accepted January 15th, 2014

1. PENDAHULUAN

Sistem deteksi intrusi adalah sebuah sistem untuk mendeteksi serangan atau intrusi pada suatu jaringan atau sistem komputer, umumnya pendeteksian intrusi dilakukan dengan mencocokkan pola lalu lintas jaringan dengan pola serangan yang telah diketahui (*misuse*) atau dengan mencari pola lalu lintas jaringan yang tidak normal (*anomaly*)**.** Untuk membangun sebuah sistem deteksi intrusi yang efektif dengan metode *anomaly detection* seorang analis mengandalkan intuisi dan pengalaman untuk memilih ukuran statistik dan untuk pendeteksian intrusi dan untuk *misuse detection* seorang analis pertama kali harus menganalisis dan mengkategorikan skenario serangan, kerentanan sistem, dan membuat aturan dan pola-pola yang sesuai dengan intrusi. Karena dilakukan secara manual dan maka IDS yang telah ada memiliki keterbatasan dalam kemampuan beradaptasi pada jenis serangan baru [1].

Perkembangan aktivitas internet dan serangan terhadap sistem komputer yang semakin meningkat, menyebabkan data yang harus dianalisis menjadi sangat besar dan tentunya ini menjadi masalah bagi seorang analis paket data untuk memilah data dan membentuk skenario dari data yang terkumpul tersebut, sehingga muncul kecurigaan bahwa sistem deteksi intrusi yang ada tidak dapat mendeteksi serangan-serangan yang berbahaya yang dilakukan dengan teknik yang baru, tersembunyi atau keduanya. Permasalahan ini menyebabkan diperlukan sebuah sistem yang dapat membantu analis dalam proses analisis data dan dapat menemukan serangan yang tidak dapat ditemukan oleh analis atau sensor [2].

Penerapan metode-metode *data mining* untuk mendeteksi intrusi dapat menjadi solusi dari permasalahan peningkatan jumlah data yang besar karena memiliki keunggulan dalam memproses *system logs* atau audit data dalam jumlah yang besar dan metode *data mining* dapat membantu mengintegrasikan kedua teknik pendeteksian intrusi (*anomaly* dan *misuse*). Aplikasi *data mining* juga dapat menemukan suatu pola yang rutin dalam *dataset* yang besar dan dapat memberikan suatu solusi untuk masalah reduksi data pada *dataset* yang besar sehingga mempermudah bagi analis untuk mengidentifikasi data dan memberikan analisis yang lebih efisien [3].

Support vector machine (SVM) sebagai salah satu metode dari *data mining* terbukti memiliki tingkat akurasi yang tinggi dalam mengklasifikasikan pola-pola paket data jaringan, ini dibuktikan oleh beberapa penelitian yang mempergunakan *dataset* DARPA KDD'99 yang dibuat oleh Lincoln Lab [4][5]. Berdasar dari hasil penelitian-penelitian tersebut maka SVM dinilai tepat untuk diimplementasikan dalam penelitian ini.

2. METODE PENELITIAN

2.1 Analisa

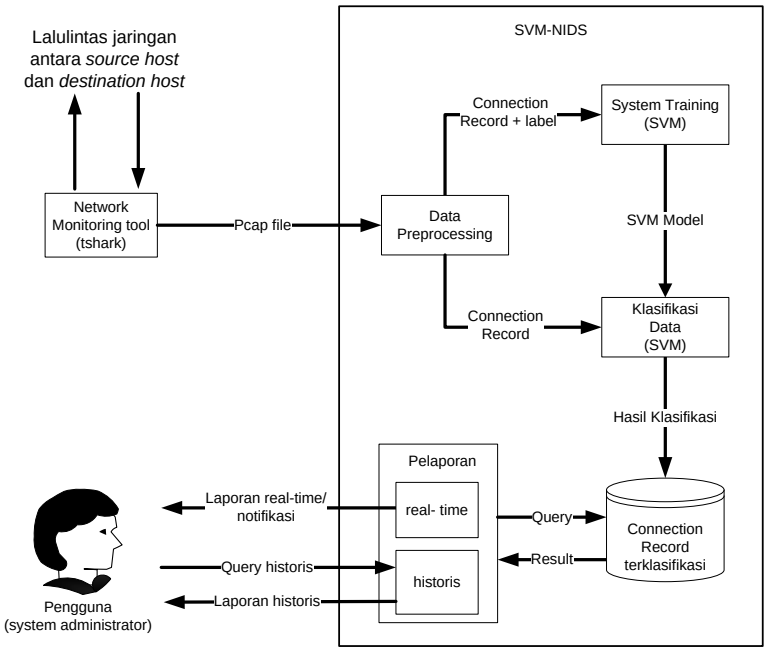
Data yang digunakan untuk mendeteksi intrusi adalah data rekaman lalu lintas paket data dalam jaringan komputer yang direkam menggunakan aplikasi *network monitoring* (*tshark*) dalam bentuk *pcap files*. Sistem yang dibangun dalam penelitian ini melakukan pengolahan rekaman paket data jaringan (*pcap file*) kedalam bentuk data audit (*connection record*). Kemampuan pendeteksian intrusi dari sistem yang dibangun masih terbatas pada jenis intrusi yang dapat diketahui lewat analisis informasi *header* paket data. *Connection record* yang terbentuk diklasifikasikan kedalam 3 kelas yaitu: normal untuk aktivitas pengaksesan layanan sistem komputer secara normal dan sifatnya tidak mengganggu, *probe* untuk aksi intrusi yang bertujuan mengumpulkan informasi sumber daya jaringan atau sistem komputer, dan *DoS* (*Denial of Service*) untuk aksi intrusi yang dapat mengganggu *availability* dari layanan sebuah sistem komputer.

Pembentukan data audit atau *connection record* memiliki 2 tujuan yaitu untuk keperluan data *training* dan pendeteksian intrusi. Pembentukan *connection record* untuk data *training* pengklasifikasiannya oleh sistem administrator dengan memberikan label atau kelas

pada tiap *record* yang terbentuk berdasar kategori serangan atau intrusi yang dilakukan, data *training* digunakan dalam proses *training* untuk memperoleh model yang digunakan dalam proses pengklasifikasian. Pada pembentukan *connection record* untuk pendeteksian intrusi tidak dilakukan penetapan label atau kelas dilakukan oleh proses pengklasifikasian berdasar model yang terbentuk dari proses *training*.

2.2 Arsitektur dan Rancangan Sistem

Desain arsitektur sistem yang dibangun diperlihatkan pada Gambar 1. Sistem (SVM-NIDS) memperoleh masukan dari aplikasi *network monitoring* (*tshark*) berupa *pcap file* dan diterima oleh bagian *preprocessing* untuk dilakukan ekstraksi data dan membentuk membentuk fitur-fitur *connection record*. Ada perbedaan untuk keluaran dari bagian *preprocessing* yaitu: *connection record* yang dilabel berdasar kelas dan tidak dilabel. *Connection record* yang dilabel digunakan untuk keperluan *training* sistem dan terbentuk pada saat pengumpulan data *training*, sedangkan *connection record* yang tidak dilabel digunakan untuk mendeteksi intrusi dalam jaringan.



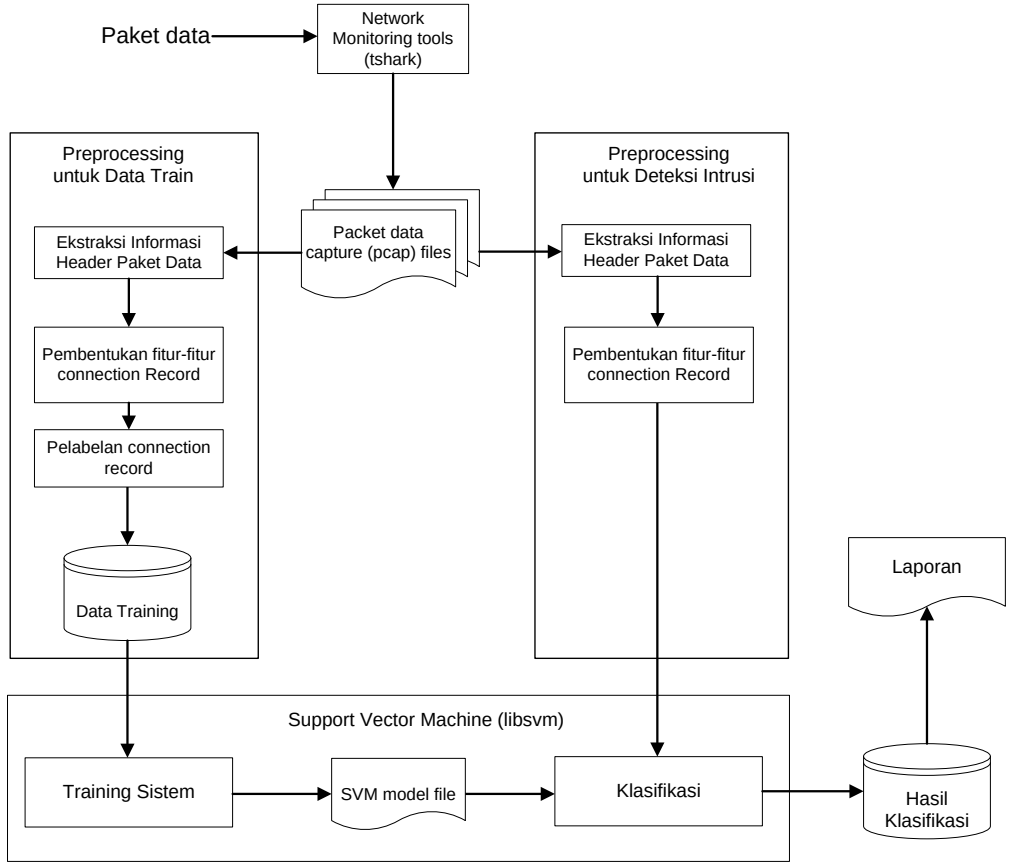
Gambar 1 Rancangan arsitektur sistem

Untuk *connection record* yang terlabel digunakan oleh bagian sistem *training* untuk melakukan *training* sistem atau analisis dari data *training* yang dihasilkan dari proses *preprocessing* dengan metode SVM. Adapun tujuan dari proses *training* itu sendiri adalah mencari model atau fungsi yang menggambarkan dan membedakan kelas-kelas data atau konsep untuk memprediksikan atau mengklasifikasikan kelas dari suatu objek yang kelas labelnya belum diketahui [6]. Model yang diperoleh dari bagian sistem *training* digunakan oleh bagian klasifikasi data untuk mengklasifikasikan data *connection record* yang belum terlabel atau untuk mendeteksi aksi-aksi intrusi. Metode yang digunakan untuk proses *training* dan klasifikasi adalah *support vector machine* (SVM), dan untuk penerapannya digunakan fungsi-fungsi yang ada dalam *library libsvm*. Hasil klasifikasi disimpan dalam basis data *connection record* terklasifikasi untuk digunakan oleh bagian pelaporan.

Bagian pelaporan bertugas untuk melakukan *query* ke dalam basis data dan melaporkannya kepada pengguna yaitu sistem administrator dengan format HTML. Pelaporan

dilakukan dalam 2 cara yaitu secara *real-time* dan historis. Pelaporan *real-time* melaporkan informasi paket data yang melewati jaringan secara berkala dan memberikan peringatan kepada sistem administrator ketika terjadi intrusi, sedangkan pelaporan historis memberikan laporan berdasar *query* dari pengguna dan digunakan untuk melihat kejadian-kejadian di masa yang lalu.

Untuk gambaran lebih jelas dari proses *preprocessing*, *training* sistem dan klasifikasi yang terjadi dalam sistem dapat dilihat pada Gambar 2.



Gambar 2 Gambaran umum sistem

Preprocessing dilakukan pada dua keadaan, keadaan pertama bertujuan membentuk data *training* dan yang kedua bertujuan untuk mendeteksi intrusi. *Preprocessing* untuk data *training* dibentuk selama masa pengumpulan data *training* atau selama masa dilakukan simulasi intrusi dan komunikasi data normal yang dilakukan oleh sistem *administrator*, selama simulasi berlangsung data yang terbentuk setelah proses ekstraksi dan proses pembentukan *connection record* diberi label sesuai kategori simulasi yang berlangsung. Data *training* yang terbentuk disimpan dalam basis data pada tabel data *training* untuk digunakan dalam proses *training* sistem. *Preprocessing* untuk keperluan pendeteksian intrusi tidak mengalami proses pelabelan, *conection record* yang terbentuk menjadi masukan dalam proses klasifikasi.

Pada proses *preprocessing, pcap file* sebelum menjadi *connection record* melewati beberapa tahap yaitu ekstraksi informasi *header* paket data, pembentukan fitur-fitur *connection record* dan pelabelan *connection record* pada *preprocessing* untuk data *training*. Tahap ekstraksi informasi *header* paket data membaca *pcap file* yang dihasilkan oleh *tshark* untuk mendapatkan informasi *header* paket data dengan protokol TCP, UDP, dan ICMP. Informasi-informasi yang diambil ditampilkan pada Tabel 1.

Tabel 1 Informasi *header* paket data TCP, UDP, ICMP

No.	Informasi Header paket data	Penjelasan
1.	*Time*	Waktu kedatangan paket data.
2.	*IP source*	Alamat IP *host* pengirim
3.	*IP destination*	Alamat IP *host* penerima
4	*IP Protocol*	Mengindikasikan tipe protokol yang digunakan (UDP, TCP, ICMP)
5.	*TCP source port*	Nomor *port* dari *source host* yang mengirimkan segmen TCP bersangkutan
6.	*TCP destination port*	Nomor *port* dari *destination host* yang menerima segmen TCP bersangkutan
7.	*TCP length*	Mengindikasikan panjang segmen TCP yang dikirim
8.	*TCP stream*	Mengindikasikan nomor *stream/connection* dari TCP
9.	*UDP source port*	Nomor *port* dari *source host* yang mengirimkan segmen UDP bersangkutan.
10.	*UDP destination port*	Nomor *port* dari *destination host* yang menerima segmen UDP bersangkutan
11.	*UDP length*	Mengindikasikan panjang segmen UDP yang dikirim.
12.	*Frame length*	Mengindikasikan panjang *frame* yang dikirim.
13.	*ICMP sequence*	Nomor *sequence* dari paket ICMP yang dikirim
14.	*ICMP type*	Mengindikasikan tipe ICMP yang dikirimkan

Berdasar informasi *header* paket data hasil ekstraksi dibentuk data audit (*connection record*) dengan fitur-fitur yang ditampilkan pada Tabel 2.

Tabel 2 Fitur-fitur *connection record*

No.	Fitur	Penjelasan
1.	*Timestamp*	Waktu kedatangan paket
2.	*Ip_src*	Alamat IP *source host*
3.	*Ip_dest*	Alamat IP *destination host*
4.	*Protocol*	Tipe protokol (ICMP, TCP, UDP).
5.	*Service_port*	Nomor port dari *destination host* dalam koneksi
6.	*Src_bytes*	Jumlah rata-rata *byte*, termasuk informasi *header* yang diterima oleh *destination host*
7.	*Dst_bytes*	Jumlah rata-rata *byte* termasuk informasi *header* yang diterima oleh *source host*
8.	*Count*	Fitur yang mengindikasikan jumlah koneksi pada *host* yang sama dalam 2 detik terakhir
9.	*Srv_count*	Jumlah koneksi pada *service* sama untuk koneksi yang sama dalam 2 detik terakhir
10.	*Dst_host_srv_count*	Jumlah koneksi ke tujuan yang sama dan *service* yang sama dalam rentang 100 *connection record* terakhir
11.	*Dst_host_same_src_port_rate*	Persentase jumlah koneksi ke *destination host* dengan nomor *source port* yang sama dalam rentang 100 *connection record* terakhir

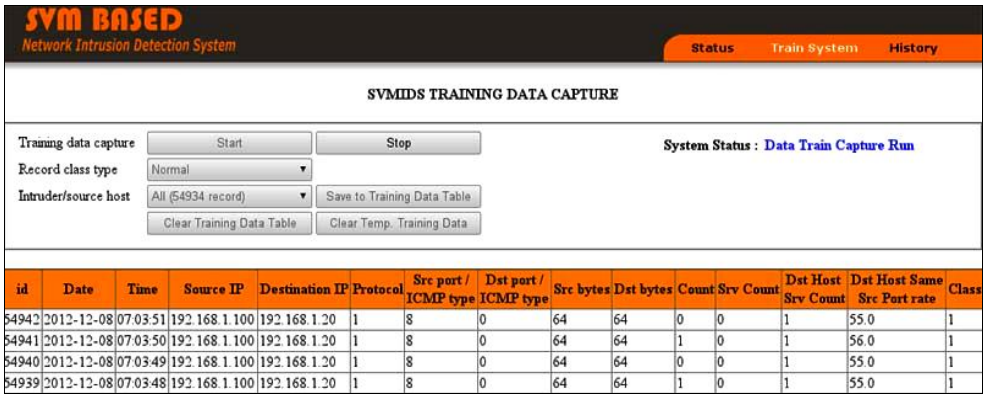
Fitur utama dari *connection record* yang terbentuk adalah fitur *src_bytes*, *dst_bytes*, *count*, *srv_count*, *dst_host_srv_count*, *dst_host_same_src_port_rate*. Fitur-fitur utama adalah fitur-fitur yang dipergunakan dalam proses *training* sistem dan pengklasifikasian data. Fitur

timestamp, *ip_src*, *ip_dest*, *protocol*, *service_port* tidak digunakan pada proses *training* sistem ataupun klasifikasi data, fitur-fitur ini merupakan fitur tambahan yang digunakan untuk kelengkapan informasi pada saat pelaporan kepada pengguna.

Pembentukan fitur-*fitur connection record* dilakukan secara *complete* dan *incomplete*, dimana *connection record* dibentuk secara *complete* ketika memiliki durasi koneksi kurang dari 2 detik dan dibentuk secara *incomplete* ketika memiliki durasi koneksi lebih dari 2 detik [7]. Untuk menyatakan 1 koneksi pada paket data dengan protokol TCP dapat berpatokan pada informasi nomor TCP *stream* dan untuk paket data dengan protokol UDP dan ICMP karena bersifat *connectionless protocol* maka setiap paket data dapat diperlakukan sebagai 1 koneksi, kecuali untuk koneksi yang memiliki *response* seperti DNS *request* dan ICMP *echo request* maka *request* dan *response* atau *reply* harus diperlakukan sebagai 1 koneksi bukan 2 koneksi. Untuk menyatakan 1 koneksi pada protokol UDP dapat berpatokan pada informasi *destination host*, *source host*, *destination port* dan *source port*, sedangkan untuk protokol ICMP dapat berpatokan pada informasi ICMP *sequence* dimana ICMP dengan nomor *sequence* yang sama dinyatakan sebagai 1 koneksi.

2.2 Implementasi Sistem

Gambar 3 adalah hasil implementasi dari *preprocessing* untuk pembentukan data *training* dimana selama proses pengumpulan data *training* bagian *System Status* akan menunjukkan "*Data Train Capture Run*" yang berarti sistem sedang berjalan dengan modus pengumpulan data *training*. Selama proses ini tidak terjadi pendeteksian intrusi, seluruh *connection record* yang terbentuk diklasifikasikan berdasar label yang ditetapkan pengguna pada bagian *Record Class type*. Data *training* yang terkumpul saat pengumpulan data *training* sifatnya masih kotor karena mengandung banyak duplikasi data. Untuk membersihkan data *training* dari duplikasi data dilakukan lewat *button Save to Training Data Tabel*, yang sekaligus menyimpan data *training* dalam tabel *data_training* secara keseluruhan atau data tertentu saja berdasar informasi yang ditetapkan pada *selection list Intruder/source host*.



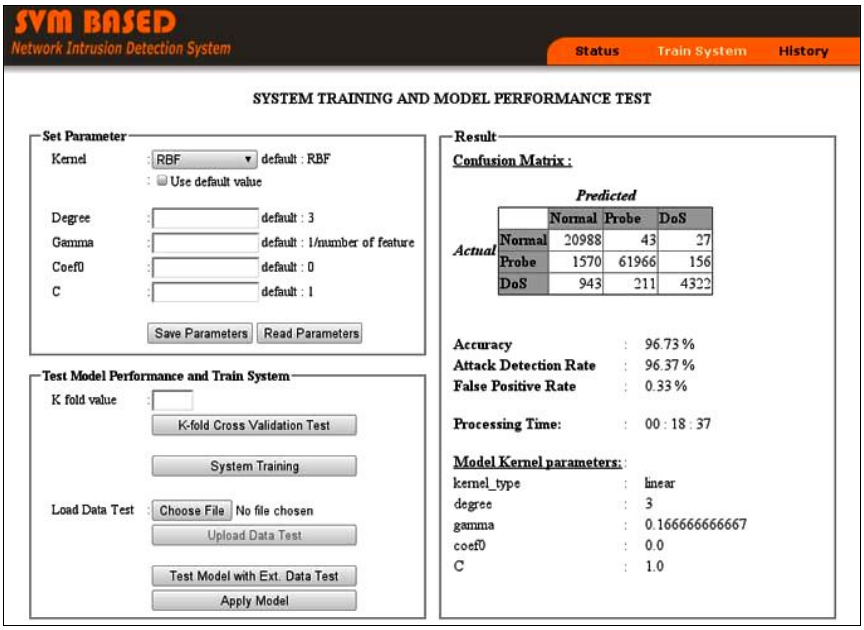
id	Date	Time	Source IP	Destination IP	Protocol	Src port / ICMP type	Dst port / ICMP type	Src bytes	Dst bytes	Count	Srv Count	Dst Host Srv Count	Dst Host Same Src Port rate	Class
54942	2012-12-08	07:03:51	192.168.1.100	192.168.1.20	1	8	0	64	64	0	0	1	55.0	1
54941	2012-12-08	07:03:50	192.168.1.100	192.168.1.20	1	8	0	64	64	1	0	1	56.0	1
54940	2012-12-08	07:03:49	192.168.1.100	192.168.1.20	1	8	0	64	64	0	0	1	55.0	1
54939	2012-12-08	07:03:48	192.168.1.100	192.168.1.20	1	8	0	64	64	1	0	1	55.0	1

Gambar 3 Tampilan proses pengumpulan data *training*.

Gambar 4 menampilkan hasil implementasi untuk proses *training* sistem dimana pada bagian ini juga dapat dilakukan pengujian dari model yang dihasilkan dari *proses training*. Proses *training* meliputi penetapan parameter *libsvm* berupa parameter *C*, tipe *kernel* dan parameter-parameter *kernel*. Pengujian dapat dilakukan dengan 2 skenario yaitu:

1. Pengujian dengan metode *k-fold cross validation*,dengan cara membagi data *training* sebanyak *k* bagian, kemudian *k-1* bagian digunakan sebagai data untuk *training* sistem dan sisanya digunakan sebagai data *test*. pengujian ini dilakukan sebanyak *k* kali dengan mengganti-ganti partisi yang berfungsi sebagai data *training* dan data *test*.
2. Pengujian dengan menggunakan eksternal data *test* dilakukan dengan menggunakan data *test* yang dibuat tersendiri diluar dari data *training*, sistem terlebih dilatih menggunakan

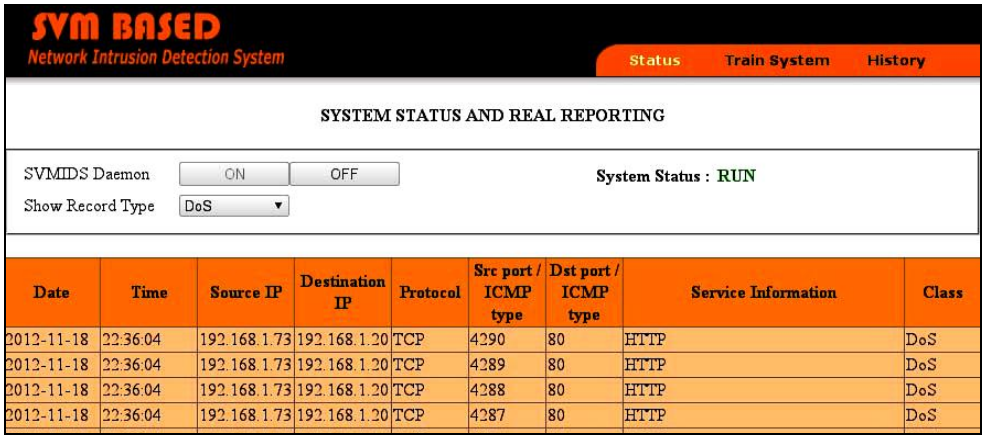
keseluruhan data *training*, dan kemudian eksternal data *test* diklasifikasi dan kemudian kelas hasil klasifikasi di bandingkan dengan kelas dari data *test* yang sebenarnya.



Gambar 4 Tampilan proses *training* dan pengujian model.

Hasil pengujian model ditampilkan pada bagian *Result* dalam bentuk *confusion matrix* untuk menunjukkan perbandingan jumlah *connection record* hasil klasifikasi (*predicted*) dan kelas yang sebenarnya (*actual*), tingkat akurasi pengklasifikasian (*accuracy*), tingkat deteksi serangan (*attack detection rate*) dan tingkat *false positive* (*false positive rate*). Selain informasi utama berupa hasil pengujian ditampilkan juga informasi tambahan berupa waktu proses (*processing time*) dan informasi parameter dari model yang diuji.

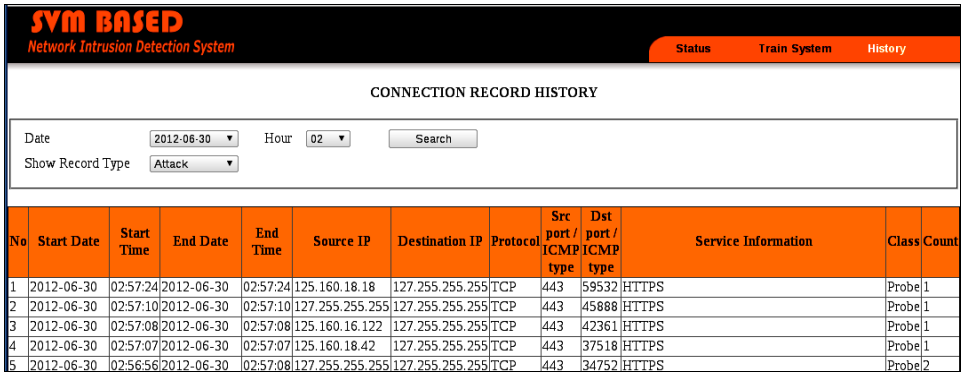
Gambar 5 adalah tampilan dari hasil implementasi laporan *real-time*, dimana bagian *system status* akan menunjukan indikasi RUN ketika sistem sedang berjalan untuk mendeteksi serangan, *button* ON/OFF berfungsi untuk menjalankan dan menghentikan sistem dan *selection list Show Record Type* berfungsi untuk melakukan penyaringan *record* yang ditampilkan.



Gambar 5 Laporan *real-time*

Gambar 6 memperlihatkan hasil implementasi untuk pelaporan historis dimana *query* yang diinputkan oleh pengguna adalah berupa tanggal (*Date*) dan jam (*Hour*) dari waktu

kejadian yang ingin dilihat, dan sama seperti pada pelaporan *real-time show record type* digunakan untuk melakukan penyaringan dari *record* yang ditampilkan.



No	Start Date	Start Time	End Date	End Time	Source IP	Destination IP	Protocol	Src port / ICMP type	Dst port / ICMP type	Service Information	Class	Count
1	2012-06-30	02:57:24	2012-06-30	02:57:24	125.160.18.18	127.255.255.255	TCP	443	59532	HTTPS	Probe	1
2	2012-06-30	02:57:10	2012-06-30	02:57:10	127.255.255.255	127.255.255.255	TCP	443	45888	HTTPS	Probe	1
3	2012-06-30	02:57:08	2012-06-30	02:57:08	125.160.16.122	127.255.255.255	TCP	443	42361	HTTPS	Probe	1
4	2012-06-30	02:57:07	2012-06-30	02:57:07	125.160.18.42	127.255.255.255	TCP	443	37518	HTTPS	Probe	1
5	2012-06-30	02:56:56	2012-06-30	02:57:08	127.255.255.255	127.255.255.255	TCP	443	34752	HTTPS	Probe	2

Gambar 6 Tampilan halaman *connection record history*

3. HASIL DAN PEMBAHASAN

Pengujian dari sistem yang dibangun dilakukan dengan cara melakukan aksi intrusi dan komunikasi secara normal ke sebuah sistem yang menjadi target. *Connection record* yang terbentuk selama aksi intrusi digunakan untuk data *training* dan pengujian secara *offline* maupun *real-time*. Performansi dari sistem didasarkan pada tiga kriteria yaitu: tingkat akurasi yang ditunjukan pada persamaan (1) , tingkat deteksi (ADR) pada persamaan (2) dan tingkat *false positiv*e (FPR) ditunjukan pada persamaan (3). Performansi dikatakan baik apabila memiliki tingkat akurasi dan tingkat deteksi yang tinggi, serta tingkat *false positive* yang rendah.

$$Akurasi = \frac{Jumlah\ data\ terklasifikasi\ benar}{Total\ jumlah\ data\ test} 100\% \quad (1)$$

$$ADR = \frac{Jumlah\ data\ serangan\ terdeteksi}{Total\ jumlah\ data\ serangan} 100\% \quad (2)$$

$$FPR = \frac{Jumlah\ data\ kelas\ normal\ terklasifikasi\ salah}{Total\ jumlah\ data\ kelas\ normal} 100\% \quad (3)$$

3.1 Pengujian Secara Offline

Pengujian secara *offline* bertujuan mencari model terbaik yang dapat digunakan untuk mendeteksi intrusi. Pengujian secara *offline* dengan metode *k-fold cross validation* mempergunakan *dataset* yang terbentuk dari aksi-aksi intrusi yang bertujuan membentuk data *training* lewat fasilitas *training data capture* dan eksternal data *test* yaitu *dataset* lain yang dibentuk dengan cara yang sama dengan pembentukan data *training* sebelumnya. Adapun *dataset* yang terbentuk dalam penelitian ini (*dataset* Simulasi) memiliki 90224 *connection record* yang terdiri atas: 21058 *record* kelas normal, 63690 *record* kelas *probe*, dan 5476 *record* kelas DoS, sedangkan untuk data *test* Simulasi memiliki 17535 *connection record* yang terdiri atas 14217 *record* kelas normal, 533 *record* kelas *probe*, dan 2785 *record* kelas DoS.

Tabel 3 memperlihatkan hasil pengujian *dataset* simulasi dengan *metode k-fold cross validation*. Dari hasil pengujian *offline* dengan *metode k-fold cross validation* dipilih model terbaik adalah model dengan tipe *kernel* polinomial C=100, karena memiliki tingkat akurasi

tertinggi dan tingkat deteksi yang tinggi juga serta tingkat *false positive* yang rendah pada nilai C=100.

Tabel 3 Performansi model pada *dataset* simulasi

Parameter C	***Kriteria***	***Tipe kernel***			
		Linear	**RBF**	***Polinomial***	***Sigmoid***
1	Akurasi	96,73	95,86	96,0	94,74
	ADR	96,37	95,01	95,18	95,30
	FPR	0,33	0,27	0,20	5,99
5	Akurasi	97,52	97,17	97,25	94,14
	ADR	97,42	96,82	96,99	94,87
	FPR	0,34	0,18	0,23	7,73
10	Akurasi	97,72	97,74	97,58	93,82
	ADR	97,70	97,22	97,37	94,85
	FPR	0,47	0,25	0,30	9,07
25	Akurasi	97,94	98,09	98,16	93,25
	ADR	98,01	97,63	97,74	94,36
	FPR	0,55	0,36	0,42	9,89
50	Akurasi	98,05	98,29	98,34	92,50
	ADR	98,17	97,90	97,98	94,28
	FPR	0,61	0,41	0,50	12,42
100	Akurasi	98,13	98,43	98,46	92,98
	ADR	98,30	98,11	98,16	94,12
	FPR	0,64	0,51	0,56	10,24

Tabel 4 menunjukan performansi model dari pengujian dengan menggunakan eksternal data *test*.

Tabel 4 Performansi model pada pengklasifikasian data *test* simulasi

Parameter C	***Kriteria***	***Tipe kernel***			
		Linear	**RBF**	***polinomial***	***Sigmoid***
1	Akurasi	90,14	89,40	89,81	88,08
	ADR	66,60	68,80	67,10	72,23
	FPR	3,48	5,04	4,19	7,45
5	Akurasi	85,22	89,48	89,68	83,76
	ADR	48,93	64,40	63,40	71,35
	FPR	5,94	4,11	3,714	12,49
10	Akurasi	79,18	88,29	84,12	84,03
	ADR	44,23	64,22	66,16	68,97
	FPR	11,67	5,48	6,55	11,78
25	Akurasi	66,87	82,89	75,98	83,72
	ADR	19,46	65,89	66,72	68,50
	FPR	20,57	12,37	14,63	12,06
50	Akurasi	63,87	75,78	69,02	83,61
	ADR	19,87	66,66	67,27	68,42
	FPR	24,32	21,17	22,21	12,18
100	Akurasi	61,90	70,61	64,16	83,57
	ADR	20,11	66,48	68,56	68,42
	FPR	26,81	27,40	27,62	12,22

Dari pengujian ini diperoleh model terbaik adalah model dengan tipe *kernel linear* pada nilai *C*=1 dengan pertimbangan tingkat akurasi dari model ini adalah yang tertinggi dari model yang lain disertai tingkat deteksi yang tinggi serta tingkat *false positive* yang rendah pada nilai C=1.

3.2 Pengujian Secara Real-Time

Pengujian secara *real-time* dilakukan dengan menguji 2 model yang diperoleh dari pengujian secara *offline* pada situasi yang nyata. Sistem dilatih dengan menggunakan parameter-parameter dari 2 model terbaik hasil pengujian *offline*, kemudian diaplikasikan untuk mengklasifikasikan *connection record* atau mendeteksi aksi intrusi yang dilakukan pada sebuah sistem komputer, 2 model tersebut adalah: model dengan tipe kernel polinomial *C*=100 (M1) dan model dengan tipe kernel linear *C*=1 (M2).

Tabel 5 menunjukan perbandingan performansi dari kedua model ketika dilakukan aksi intrusi dan komunikasi normal ke sebuah sistem komputer. Dari Tabel 5 dapat dilihat bahwa performansi dari model 2 lebih baik daripada model 1 dari segi tingkat akurasi dan tingkat deteksi serangan, akan tetapi dari segi tingkat *false positive* model 1 memiliki tingkat *false positive* yang lebih baik dari pada model 2.

Tabel 5 Perbandingan performansi model pada pengujian *real-time*

Model	Tingkat Akurasi (%)	Tingkat Deteksi Intrusi (%)	Tingkat False Positive (%)
Model 1	86,38 %	82,05	7,17
Model 2	89,68 %	88,37	8,63

Tabel 6 menunjukkan perbandingan kemampuan pendeteksian serangan pada tiap-tiap aksi intrusi yang dilakukan.

Tabel 6 Perbandingan *attack detection rate* secara *real-time*

No	Intrusi	Tipe Intrusi	Jumlah Record	Hasil klasifikasi				Attack Detection Rate (%)	
				Benar		Salah			
				M1	M2	M1	M2	M1	M2
1	Intense Scan	*Probe*	545	541	533	4	12	99,27	97,80
2	Intense Scan Plus UDP		1945	624	1530	1321	415	32,08	78,66
3	Intense Scan All TCP Ports		3534	3481	3483	53	51	98,50	98,56
4	Intense Scan No Ping		391	387	379	4	12	98,98	96,93
5	Reguler Scan		457	447	442	10	15	97,81	96,72
6	Quick Scan		227	216	211	11	16	95,15	92,95
7	Quick Scan Plus		284	257	229	27	55	90,49	80,63
8	Slow Comprehensive Scan		2645	453	1869	2192	776	17,13	70,66
9	Ping -l 7500	DoS	3030	2807	2834	223	196	92,64	93,53
10	Hping		2896	2508	2482	388	414	86,60	85,70
11	Letdown		3368	2876	2848	492	520	85,39	84,56
12	UDP.pl		3852	3676	3729	176	123	95,43	96,81
13	Slowloris		2003	1546	1507	457	496	77,18	75,24

Dari Tabel 6 dapat dilihat bahwa model 1 memiliki kemampuan pendeteksian serangan yang lebih baik daripada model 2 pada sebagian besar intrusi yang dilakukan, akan tetapi pada beberapa intrusi model 2 menunjukkan performansi yang jauh lebih baik daripada model 1 yaitu pada aksi intrusi *Intense Scan Plus UDP* dan *Slow Comprehensive Scan*, hal ini yang menyebabkan turunnya performansi dari model 1 dibandingkan model 2.

4. KESIMPULAN

Berdasar hasil-hasil pengujian dari sistem yang telah dibangun dapat ditarik kesimpulan bahwa:

1. Penerapan metode klasifikasi *support vector machine* dalam sistem deteksi intrusi yang telah dibangun dapat membantu analis dalam pembentukan *profile*, skenario intrusi atau model secara otomatis, dimana dari hasil pengujian model yang dihasilkan oleh sistem ini dapat mendeteksi aksi intrusi yang dilakukan dengan tingkat akurasi dan tingkat deteksi yang tinggi, serta tingkat *false positive* yang rendah.
2. Pengujian secara *real-time* membuktikan bahwa penerapan fungsi *complete/incomplete connection record* dalam proses *preprocessing* pembentukan data audit atau *connection record* dapat menjadi solusi permasalahan keterlambatan pendeteksian akibat durasi koneksi yang terlalu lama.
3. Dari hasil pengujian secara *real-time*, sistem belum dapat mendeteksi secara efektif aksi intrusi ketika aksi tersebut baru mulai dilakukan, *connection record* yang terbentuk pada saat aksi intrusi baru dilakukan masih diklasifikasikan dengan kelas normal, kondisi ini tidak hanya mempengaruhi tingkat pendeteksian serangan tapi juga memberi celah bagi aksi-aksi intrusi yang dilakukan dengan mengirim paket data secara lambat.

5. SARAN

Disadari bahwa aplikasi ini masih memiliki kelemahan-kelemahan yang membutuhkan pengembangan lanjutan untuk membuatnya menjadi lebih baik, sehingga untuk selanjutnya diharapkan dapat dilakukan hal-hal berikut untuk memperbaikinya:

1. Perbaikan pada proses pembentukan *connection record*. Pada proses pembentukan fitur *connection record* masih belum bisa ditentukan dengan benar mana yang menjadi *source host* dan *destination host*. *Source host* dan *destination host* masih didasarkan pada informasi yang ditemukan pertama kali, sehingga akibat penerapan *incomplete connection record* kemungkinan untuk salah mendefinisikan *source host* dan *destination host* menjadi besar yang berakibat pada kesalahan klasifikasi.
2. Perlu dilakukan penelitian untuk metode penentuan parameter SVM. Berdasar hasil pengujian dapat dilihat bahwa penentuan parameter SVM yang tepat dapat memberikan model dengan performansi yang baik dan untuk mendapatkannya perlu dilakukan pengujian berulang-ulang dengan kombinasi parameter yang berbeda. Pencarian parameter yang dilakukan dengan mencoba kombinasi parameter satu persatu sangat tidak efisien dari segi waktu dan belum tentu akan mendapatkan model dengan performansi terbaik.
3. Pengembangan penelitian untuk ekstraksi informasi *payload* atau isi dari data yang ditransmisikan. Sistem yang dibangun baru bisa mendeteksi intrusi berdasar informasi *header* paket data, dengan adanya penelitian ekstraksi informasi *payload* diharapkan dapat dibentuk fitur-fitur baru untuk pendeteksian intrusi pada level aplikasi seperti *cross site scripting* dan SQL *injection.*

DAFTAR PUSTAKA

[1] Lee, W., 1999. A Data Mining Framework for Constructing Features and Models for Intrusion Detection Sistems, *Tesis*, School of Arts and Sciences Columbia University, New York

[2] Bloedorn E., Christiansen,D., Hill, W., Skorupka, C., Talbot, L., dan Tivel, J., 2001, Data Mining for Network Intrusion Detection: How to Get Started, *MITRE Technical Report*, Agustus 2001

[3] Agathou A. dan Tzouramanis, T., 2008, *The Role Of Data mining in Intrusion Detection Technology*, Garson, G.D. dan Khosrow-Pour,M (ed.), *Handbook of Research on Public Information Technology*, Vol.1, Information Science Reference, Hershey,New York

[4] Tavallaee, M., Bagheri, E., Lu, W., dan Ghorbani, A. A., 2009, A Detailed Analysis of the KDD CUP 99 Data Set, *Proceedings of the 2009 IEEE Symposium on Computational Intelligence in Security and Defense Applications* (CISDA 2009)

[5] Shrivastava, K. S., dan Jain, P., 2011, Effective Anomaly Based Intrusion Detection using Rough Set Theory and Support Vector Machine, *International Journal of Computer Applications*, Vol.18-No 3, hal 35-41

[6] Han, J., dan Kamber, M., 2006, *Data mining: Concepts and Techniques*, Ed.2 Morgan Kaufmann Publishers, San Francisco, USA

[7] Benferhat, S., Sedki, K., dan Tabia, K., 2007. Preprocessing Rough Network Traffic For Intrusion Detection Purposes. *IADIS International Telecommunications, Networks and* Sistems *2007*, hal.105-109.