NOTIFICATION OF SECURITY THREATS ON THE INTERNET PROXY SERVER IS A SERVER-BASED SHORT MESSAGE SERVICE (SMS)

Mario Agapito Arizald Gobel, Sumarsono, Yuliani Indrianingsih
Jurusan Teknik Informatika
Sekolah Tinggi Teknologi Adisutjipto Yogyakarta
informatika@stta.ac.id

ABSTRACT

Defense system against interference when the activity is generally done manually by the administrator. This resulted in the integrity of the system depends on the availability and speed of the administrator in response to disturbance. In the current era of information technology, almost all information is important for an institution can be accessed by its users. Disclosure of such access raises new security issues in part of a computer network system that is very important to maintain the validity and integrity of data and ensure availability of services for its users.

In this issue then designed a system to detect any interference. The detection is based filtering on the port service, using the programming language Delphi 7. The data obtained from the filtered traffic data packets passing the proxy server. By building a system that can work automatically and in real time, then perform the detection of intruders or illegal activities which will then be reported to the administrator.

With this administrator can find out the use of illegal ports through port filter techniques on the traffic data service in and out in real time, from both internal and external networks. Administrators can do the monitoring without having to depend on availability. With such a system administrator can perform tasks more efficiently in maintaining the validity and integrity of data.
***Key words: security threats, Internet Proxy Server*, SMS.**

ABSTRAK

Sistem pertahanan terhadap aktivitas gangguan saat ini umumnya dilakukan secara manual oleh *administrator*. Hal ini mengakibatkan integritas sistem bergantung pada ketersediaan dan kecepatan *administrator* dalam merespon gangguan. Dalam era teknologi informasi saat ini, hampir seluruh informasi yang penting bagi suatu institusi dapat diakses oleh para penggunanya. Keterbukaan akses tersebut memunculkan berbagai masalah. Keamanan jaringan komputer sebagian dari sebuah sistem yang sangat penting untuk menjaga validitas dan integritas data serta menjamin ketersediaan layanan bagi penggunanya.

Dalam permasalahan ini maka dirancang sebuah sistem untuk mendeteksi setiap gangguan. Pendeteksian dilakukan berdasarkan *filtering* pada *port service*, dengan menggunakan bahasa pemrograman Delphi 7. Data yang terfilter didapat dari lalulintas paket data yang lewat di *proxy* server. Dengan membangun sebuah sistem yang dapat bekerja secara otomatis dan *real time*, lalu melakukan

pendeteksian penyusup atau aktifitas-aktifitas terlarang yang kemudian akan dilaporkan kepada *administrator*.

Dengan ini *administrator* dapat mengetahui penggunaan port secara illegal melalui teknik *filter port service* pada lalulintas data yang keluar masuk secara *real time*, baik dari jaringan internal maupun eksternal. *Administrator* bisa melakukan pemantauan tanpa harus tergantung pada ketersediaan. Dengan sistem seperti ini *administrator* dapat dengan lebih efisien dalam melakukan tugas menjaga validitas dan integritas data.

Kata kunci : Gangguan keamanan, Internet Proxy Server, SMS.

1. Latar Belakang

Keamanan jaringan komputer sebagai bagian dari sebuah sistem yang sangat penting untuk menjaga validitas dan integritas data serta menjamin ketersediaan layanan bagi penggunanya. Sistem harus dilindungi dari segala macam serangan dan usaha-usaha penyusupan atau pemindaian oleh pihak yang tidak berhak. Sistem deteksi penyusup jaringan yang ada saat ini umumnya mampu mendeteksi berbagai jenis serangan tetapi tidak mampu mengambil tindak lanjut. Selain itu sistem juga tidak memiliki interaktivitas dengan *administrator* pada saat *administrator* tidak mengadministrasi sistemnya. Hal ini merupakan suatu kondisi yang tidak efektif terutama pada saat sistem berada dalam kondisi kritis.

Dalam era teknologi informasi saat ini, hampir seluruh informasi yang penting bagi suatu institusi dapat diakses oleh para penggunanya. Keterbukaan akses tersebut memunculkan berbagai masalah baru antara lain:

a. Pemeliharaan validitas dan integritas data/informasi.
b. Jaminan ketersediaan informasi bagi pengguna yang berhak.
c. Pencegahan akses informasi dari yang tidak berhak.
d. Pencegahan akses sistem dari yang tidak berhak.

Sistem pertahanan terhadap aktivitas gangguan saat ini umumnya dilakukan secara manual oleh *administrator*. Hal ini mengakibatkan integritas sistem bergantung pada ketersediaan dan kecepatan *administrator* dalam merespon gangguan. Apabila gangguan tersebut berhasil membuat suatu jaringan mengalami malfungsi, *administrator* mungkin saja tidak dapat lagi mengakses sistem secara *remote* sehingga ia tidak akan dapat melakukan pemulihan dengan cepat. Oleh karena itu di butuhkan suatu sistem yang dapat menanggulangi ancaman yang mungkin terjadi secara optimal dalam waktu yang cepat dan secara otomatis sehingga memungkinkan ketersediaan dan kecepatan *administrator* dalam merespon gangguan.

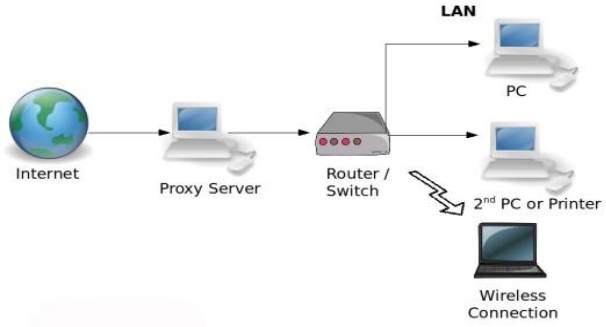
Dalam implementasinya untuk pengamana suatu sistem maka harus dilakukannya pendeteksian penyusup, deteksi penyusup sendri adalah aktivitas untuk mendeteksi penyusup melalui aktivitas yang dilakukan oleh penyusup dan secara cepat dengan menggunakan program khusus yang otomatis serta *real time* merespon gangguan atau aktivitas yang membahayakan tersebut. Dengan menggunakan metode seperti ini optimalisasi pengamanan suatu sistem dapat dilakukan dengan efisien oleh *administrator*.

2. Landasan Teori

Proxy

Proxy server adalah sebuah komputer yang memelihara dua hubunngan jaringan komputer antara internet dan jaringan local (internal). Proxy dapat dipahami sebagai pihak ketiga yang berdiri ditengah-tengah antara kedua pihak yang saling berhubungan dan berfungsi sebagai perantara, sedemikian sehingga pihak pertama dan pihak kedua tidak secara langsung berhubungan, akan tetapi masing-masing berhubungan dengan perantara, yaitu *proxy*. Tidak ada lalulintas jaringan yang lewat secara langsung tanpa melewati *proxy server*. *Proxy server* menyampaikan komunikasi dengan internet , kemudian mentransmisikan jawaban kembali ke pemakai internal asal.

Proxy dalam pengertiannya sebagai perantara, bekerja dalam berbagai jenis protokol komunikasi jaringan dan dapat berada pada level-level yang berbeda pada hirarki lapisan protokol komunikasi jaringan. Suatu perantara dapat saja bekerja pada lapisan *Data-Link, layer Network* dan *Transport*, maupun lapisan Aplikasi dalam hirarki layer komunikasi jaringan menurut OSI. Namun pengertian *proxy server* sebagian besar adalah untuk menunjuk suatu *server* yang bekerja sebagai *proxy* pada lapisan Aplikasi (Morgan Stern, 1998).



Gambar 1 *Proxy Server*.

Notification

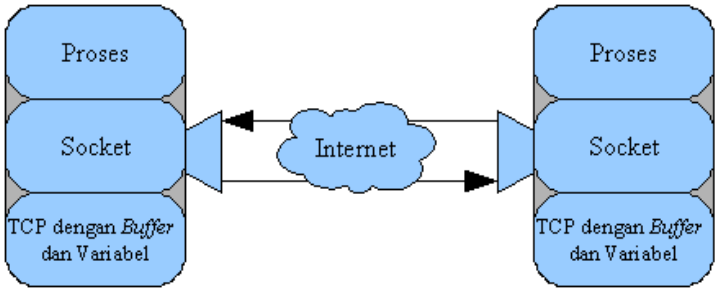
Notification adalah pemberitahuan, dalam hal ini pemberitahuan yang dimaksud adalah pemberitahuan kepada Administrator. Notifikasi dalam hal ini berisi tentang informasi-informasi yang dibutuhkan oleh pengeloa jaringan untuk tujuan-tujuan tertentu misalnya dalam pengawasan jaringan. Informasi yang terdapat didalam Notifikasi terdiri dari beberapa bagian dari paket data yang keluar masuk di jaringan komputer melalui *proxy*. Informasi yang diberitahukan adalah sebagai berikut :

a. *Time*, waktu pada saat paket data dikirimkan.
b. IP *Soure*, nomor IP pengirim paket data.
c. *Host Name*, nama komputer pengirim paket data.
d. *Service*, *port service* yang digunakan (Telnet, SSH, HTTP, dll).
e. *Port*, nomor *port* yang digunakan.

Informasi ini dikirimkan sebagai notifikasi kepada pengelola yang menjadi bahan untuk pengawasan jaringan komputer atau tujuan lainnya.

Socket

Socket adalah salah satu titik akhir dari jalur komikasi dua arah antara dua program yang berjalan pada jaringan yang berjalan pada protokol TCP/IP atau UDP. *Socket* terhubung pada sebuah nomor *port* sehingga lapisan TCP dapat mengidentifikasi dimana pengrim data ditunjukan. *Socket* merupakan sebuah aplikasi yang dibuat, dikontrol antarmuka sistem operasi dimana aplikasi dapat saling mengirim dan menerima pesan dari atau pada aplikasi yang lain. Proses kerja socket dijelaskan pada gambar 2.



Gambar 2 Proses kerja *socket*.

Definisi tentang *socket* sebagai berikut:

a. Mengizinkan suatu proses unik untuk berkomunikasi satu dengan yang lain.
b. Memberitahukan proses mana untuk menulis atau membaca.
c. Komunikasi pada mesin yang sama, jaringan dan *internet*.
d. Komunikasi dua arah membuatnya cocok untuk model *client server*.
e. dibentuk tahun 1980 oleh *Berkeley Unix Distribution*.

Pada umumnya, sebuah *server* merupakan sebuah komputer yang khusus dan memiliki sebuah *socket* yang terhubung pada sebuah nomor *port* yang khusus. *Server* hanya menunggu *socket* untuk menerima permintaan koneksi (*request*) dari sebuah *client (passive open). Client* mengetahui nama mesin tempat *server* dijalankan dan nomor *port server* tersebut dihubungkan. Untuk membuat koneksi permintaan, *client* mencoba untuk melakukan koneksi dengan *server* pada mesin *server* dan *port (active open*), seperti yang dijelaskan pada gambar 3.

Gambar 3 *Client* meminta koneksi

Apabila berjalan dengan baik maka *server* menerima koneksi (*virtual circuit)*. Saat penerimaan, *server* membuka *socket* baru yang terhubung pada *port* yang berbeda. Untuk dapat melanjutkan penerimaan dari *socket* awal untuk koneksi permintaan saat melayani kebutuhan *client* yang terhubung, dibutuhkan sebuah *socket* baru (dengan nomor *port* yang beda), seperti yang dijelaskan pada gambar 4.

Gambar 4 Respon *server*

Pada sisi *client*, jika koneksi diterima, sebuah *socket* berhasil dibentuk dan *client* dapat menggunakan *port* tersebut untuk berkomunikasi dengan *server*. *Client* dan *server* sekarang dapat berkomunikasi untuk menulis dan membaca dari *socket* tersebut.

Ada 2 macam protokol yang dapat digunakan pada *socket*, yaitu:

a. TCP bersifat *connection-oriented* dan *reliable*.
   1. *Connection-oriented*, untuk melakukan komunikasi diperlukan koneksi terlebih dahulu.
   2. *Reliable*, menjamin sampainya paket data kekomputer tujuan dan melakukan pengecekan. Apabila ada paket data yang hilang atau rusak maka *socket* TCP akan meminta paket data yang rusak tersebut sampai semua paket data yang diterima dalam kondisi yang baik.

b. UDP bersifat *connectionless* dan *unreliable*.

Beberapa nomor port yang sering digunakan 21(FTP), 23(telnet), 25(SMTP), 69(TFTP), 80(HTTP). Ada beberapa langkah agar sebuah aplikasi *client server* dapat saling berkomunikasi, seperti pada Gambar 5.

Gambar 5 Komunikasi *client-server*.

Pada saat sebuah *socket* terbentuk, program harus menetapkan *address domain* dan tipe *socket*-nya. Dua buah dapat saling berkomunikasi satu dengan yang lainnya hanya jika tipe dan *domain*-nya sama. Ada dua *domain address* umum yang digunkan, yaitu:

a. *Unix domain,* adalah *socket* yang tidak dapat dijangkau dari internet sacara langsung, tetapi bersamaan dengan datangnya permintaan pada *server.*
b. *Internet domain*, adalah dua proses yang berjalan pada dua buah *host* dikomunikasikan *internet*. Alamat *socket* dari *internet domain* terdiri atas *Internet address* dari mesin *host*. Selain itu sebuah *socket* membutuhkan nomor *port* pada *host*-nya. Nomor *port* berupa 16 *bit unsigned integers*. Angka yang rendah disediakan dalam *Unix* untuk layanan standar. sebagai contoh nomor *port* untuk FTP *server* adalah 21. Hal ini penting bahwa layanan standar akan berada pada *port* yang sama pada semua komputer sehingga *client* akan mengetahui alamatnya. Ada *port* 65535 nomor *port* yang tersedia.

Untuk tipe *socket* ada 2 macam, *stream socket* dan *datagram socket*:

a. *Stream socket* adalah kominikasi *continous stream* dari karakter (menggunakan protokol TCP).
b. *Datagram socket* adalah pembacaan seluruh pesan dengan segera (mengggunakan protokol UDP) (Morgan Stern, 1998).

3. Perancangan

Analisa kebutuhan sistem

Dalam sistem yang akan dibuat ini terdapat beberapa permasalahan tentang bagaimana sistem yang dibuat bekerja serta kebutuhan *software* dan *hardware* untuk perancangan dan uji coba aplikasi.

Analisa permasalahan

Permasalahan tentang bagaimana aplikasi bekerja untuk memenuhi kebutuhan pengamanan yang diinginkan antara lain yaitu:

1 Bagaimana aplikasi melakukan *capture packet* data dari seluruh dari seluruh paket data yang masuk melalui *proxy*.
2 Bagaimana aplikasi melakukan *filtering* terhadap paket data yang dapat membahayakan jaringan lokal ataupun *server*.
3 Bagaimana aplikasi melaporkan gangguan keamanan kepada pengelola (*administrator*) secara *real time*.
4 Bagaimana aplikasi bisa bekerja sendiri tanpa harus terus diawasi *administrator* sehingga lebih efisien dalam pengawasan jaringan ataupun *server*.

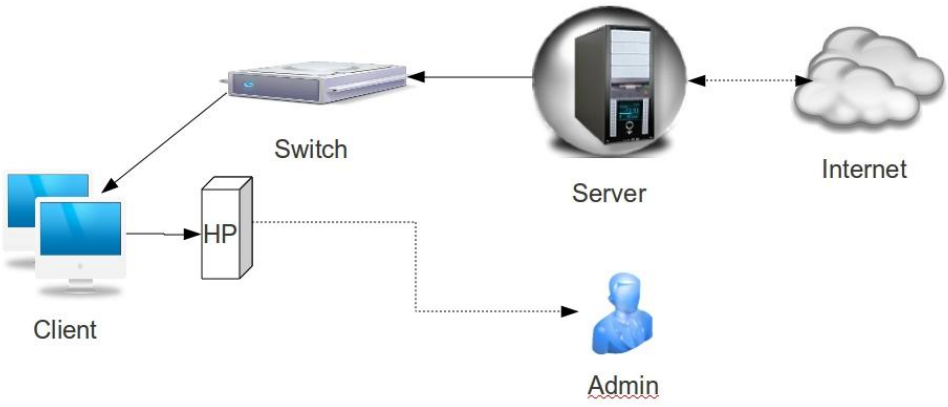
Dari empat poin diatas dapat diketahui dasar permasalahan yang akan menjadi pokok dalam perancangan sistem. Dengan ini maka perancangan sistem nantinya diharuskan mensolusikan semua permasalan diatas, sebagai dasar perancangan dari empat poin diatas adalah *packet capture, filtering, reporting, efficiency*. Sistem pertahanan terhadap aktivitas gangguan saat ini umumnya dilakukan secara manual oleh *administrator*, hal ini mengakibatkan integritas sistem bergantung pada ketersediaan dan kecepatan *administrator* dalam merespon

gangguan. Permasalahan ini yang menjadi titik berat perancangan sehingga integritas sistem tidak bergantung pada ketersediaan dan kecepatan *administrator* saja.

Analisa sistem

Sistem yang akan dibuat terdiri dari dua aplikasi yang bekerja sebagai satu kesatuan, yaitu aplikasi yang bekerja di-*server* dan aplikasi *client*, aplikasi di *server* bekerja melakukan *capture packet* data dan *filtering* terhadap paket data yang masuk dari luar, sedangkan aplikasi yang bekerja di-*client* bertugas menerima data hasil *filtering* dari aplikasi di-*server* lalu melaporkannya kepada *administrator* melalui SMS (*short messages services*). Aplikasi di-*client* juga mempunyai fungsi kontrol terhadap aplikasi yang berjalan di-*server*, hal ini dimaksudkan agar *administrator* tidak perlu melakukan kontak langsung dengan *server* untuk melakukan pengecekan. Kedua aplikasi tersebut mempunyai spesifikasi sebagai berikut:

a. Aplikasi *Server*
    a. *Packet capture*, melakukan *capture packet* data yang masuk kedalam jaringan lokal.
    b. *Filtering*, melakukan *filter* terhadap semua paket data yang masuk kedalam jaringan lokal berdasarkan *port* (*port filter*)
    c. *Trace route*, melakukan pelacakan IP dan *host name* dari paket data yang membahayakan jaringan ataupun *server*.
    d. *Port setting*, menyediakan konfigurasi port yang akan dibuka di-*server*, *port* yang telah dibuka ini untuk tujuan melakukan koneksi dengan aplikasi *client* sebagai penerima data yang akan dilaporkan nantinya.
b. Aplikasi *Client*
    e. *Send SMS*, melakukan pengiriman SMS dengan tujuan pelaporan (*report*) hasil pengawasan yang dilakukan sistem kepada *administrator*.
    f. *Control capture*, melakukan kendali terhadap aplikasi *server* untuk menjalankan dan memberhentikan aplikasi *server* dalam melakukan *capture packet*.
    g. *Port connect*, melakukan koneksi ke aplikasi *server* melalui port yang telah dibuka diaplikasi *server* sehingga dapat mengontrol dan menerima data dari aplikasi *server*.
    h. *Viewer*, Menampilkan data hasil *capture* dan hasil *filter* yang didapat dari hasil di-*server*, yang nantinya data-data yg ter-*filter* tersebut akan dilaporkan. Analisa kerja sistem *notification* dijelaskan pada gambar 6.



Gambar 6 Analisa kerja sistem *notification*.

Analisa *Filtering*

Dari hasil analisa diatas menitik beratkan pendeteksian berdasarkan *filter* data. Teknik *filter* yang digunakan adalah mendeteksi berdasarkan *port* yang digunakan. Dengan pendeteksian yang berdasar pada *filtering* paket data melalui penggunaan *port service*, maka kategori aktifitas *illegal* yang dapat di-*filter* penggunaan *port service* secara *illegal*. Kategori yang dimaksudkan adalah menggunakan sebuah *port* yang tidak diijinkan digunakan selain pengelola jaringan (*administrator*). Dalam *filtering* itu sendiri pengguna dapat menyesuaikan sesuai dengan kebutuhan sehingga tidak statis dalam *filtering* tapi lebih fleksibel berdasarkan kebutuhan pengguna sistem.

4. Uji coba

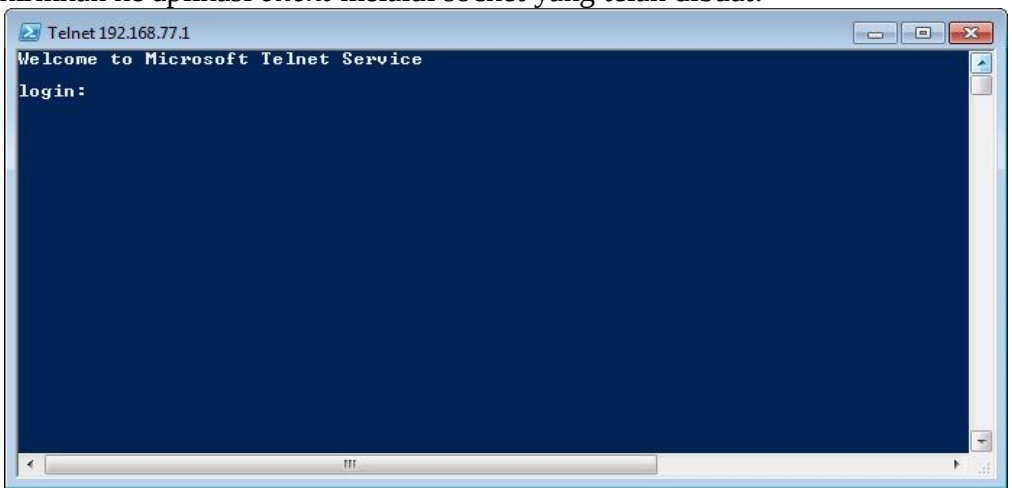
Sesuai dengan konfigurasi diatas maka akan disimulasikan percobaan untuk menjalankan aplikasi dan melakukan percobaan aktifitas yang dapat dideteksi oleh aplikasi. Dalam pengujian akan digunakan 2 komputer untuk mencoba melakukan aktifitas illegal dengan menggunakan 2 *tools* berbeda. *Tools* yang digunakan untuk percobaan yaitu dengan menggunakan Telnet dan SSH.

Percobaan 1 (Telnet)

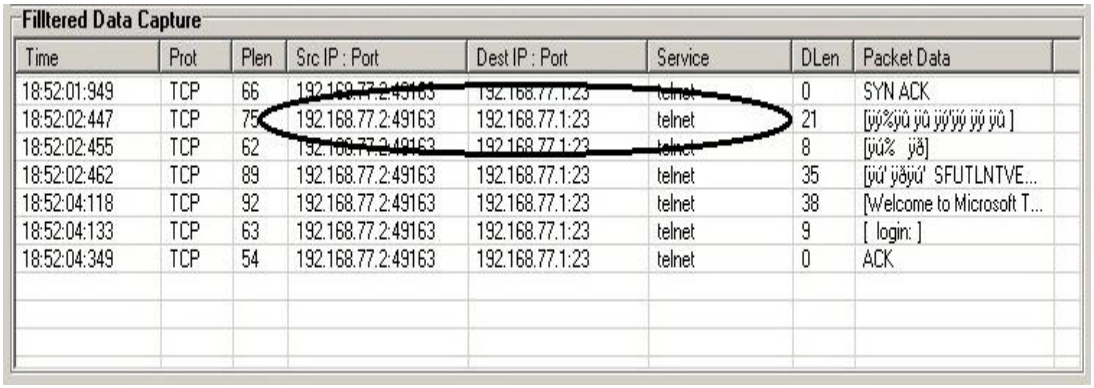
Telnet adalah sebuah aplikasi yang digunakan untuk mengendalikan komputer ataupun mengakses komputer dari jauh, aplikasi ini biasanya digunakan untuk memudahkan *administrator* dalam mengakses suatu komputer tanpa harus ada didepan komputer tersebut. Namun dengan model akses seperti dapat memungkinkan siapa saja dapat mengakses komputer apabila berhasil mengetahui akun yang digunakan *administrator* untuk masuk, dengan begini aktifitas semacam ini dapat menjadi aktifitas yang illegal.

Skenario yang digunakan dalam percobaan pertama menggunakan telnet di komputer dengan IP 192.168.77.2, disini akan dicoba melakukan Telnet ke komputer *server* (program *server* berjalan), seperti pada gambar 7. Aplikasi yang berjalan di *server* akan mendeteksi ini dan menampilkannya serta mengirimkan ke aplikasi *client* untuk ditampilkan juga (gambar 8 dan 9). Data yang terdeteksi akan disimpan untuk dilakukannya proses *trace route* pada tahap selanjutnya

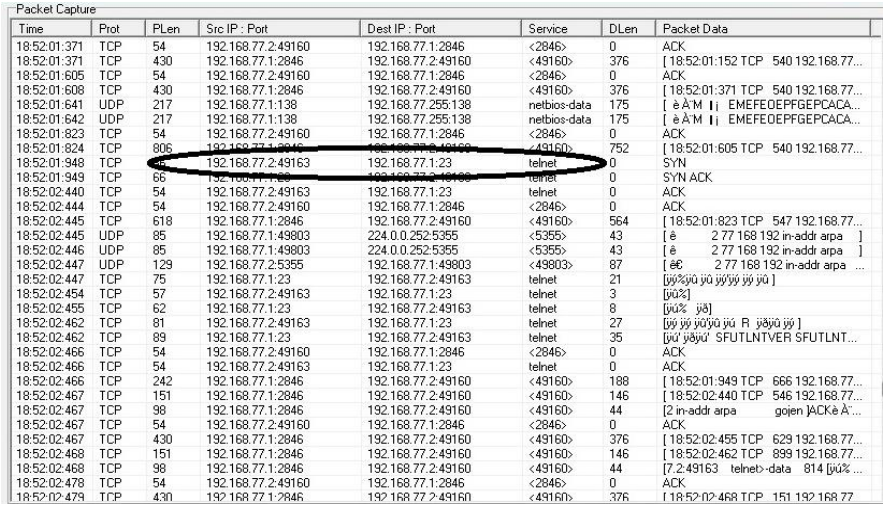
(gambar 10). Setelah terdapat beberapa data yang masuk dan telah berhasil melakukan *trace route* maka semua data yang telah berhasil dikumpulkan akan dikirmkan ke aplikasi *client* melalui socket yang telah dibuat.



Gambar 7 Telnet ke komputer *server*.

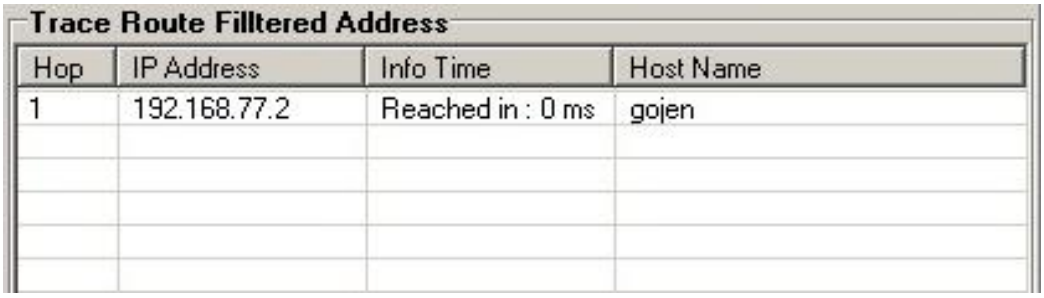


Gambar 8 Server mendeteksi aktifitas Telnet dari IP 192.168.77.2.


Packet Capture

Time	Prot	PLen	Src IP : Port	Dest IP : Port	Service	DLen	Packet Data
18:52:01:371	TCP	54	192.168.77.2:49160	192.168.77.1:2846	<2846>	0	ACK
18:52:01:371	TCP	430	192.168.77.1:2846	192.168.77.2:49160	<49160>	376	[18:52:01:152 TCP 540 192.168.77...
18:52:01:605	TCP	54	192.168.77.2:49160	192.168.77.1:2846	<2846>	0	ACK
18:52:01:608	TCP	430	192.168.77.1:2846	192.168.77.2:49160	<49160>	376	[18:52:01:371 TCP 540 192.168.77...
18:52:01:641	UDP	217	192.168.77.1:138	192.168.77.255:138	netbios-data	175	[è Ä`M ¦¡ EMEFEOEPFGEPCACA...
18:52:01:642	UDP	217	192.168.77.1:138	192.168.77.255:138	netbios-data	175	[è Ä`M ¦¡ EMEFEOEPFGEPCACA...
18:52:01:823	TCP	54	192.168.77.2:49160	192.168.77.1:2846	<2846>	0	ACK
18:52:01:824	TCP	806	[illegible]	[illegible]	<49160>	752	[18:52:01:605 TCP 540 192.168.77...
18:52:01:948	TCP	66	192.168.77.2:49163	192.168.77.1:23	telnet	0	SYN
18:52:01:949	TCP	66	[illegible]	[illegible]	telnet	0	SYN ACK
18:52:02:440	TCP	54	192.168.77.2:49163	192.168.77.1:23	telnet	0	ACK
18:52:02:444	TCP	54	192.168.77.2:49160	192.168.77.1:2846	<2846>	0	ACK
18:52:02:445	TCP	618	192.168.77.1:2846	192.168.77.2:49160	<49160>	564	[18:52:01:823 TCP 547 192.168.77...
18:52:02:445	UDP	85	192.168.77.1:49803	224.0.0.252:5355	<5355>	43	[ê 2 77 168 192 in-addr arpa]
18:52:02:446	UDP	85	192.168.77.1:49803	224.0.0.252:5355	<5355>	43	[ê 2 77 168 192 in-addr arpa]
18:52:02:447	UDP	129	192.168.77.2:5355	192.168.77.1:49803	<49803>	87	[ê€ 2 77 168 192 in-addr arpa ...
18:52:02:447	TCP	75	192.168.77.1:23	192.168.77.2:49163	telnet	21	[ÿÿ%ýú ýú ÿÿ'ÿÿ ÿÿ ýú]
18:52:02:454	TCP	57	192.168.77.2:49163	192.168.77.1:23	telnet	3	[ýú%]
18:52:02:455	TCP	62	192.168.77.1:23	192.168.77.2:49163	telnet	8	[ýú% ýð]
18:52:02:462	TCP	81	192.168.77.2:49163	192.168.77.1:23	telnet	27	[ÿÿ ÿÿ ýú'ýú ýú R ÿðýú ÿÿ]
18:52:02:462	TCP	89	192.168.77.1:23	192.168.77.2:49163	telnet	35	[ýú' ÿðýú' SFUTLNTVER SFUTLNT...
18:52:02:466	TCP	54	192.168.77.2:49160	192.168.77.1:2846	<2846>	0	ACK
18:52:02:466	TCP	54	192.168.77.2:49163	192.168.77.1:23	telnet	0	ACK
18:52:02:466	TCP	242	192.168.77.1:2846	192.168.77.2:49160	<49160>	188	[18:52:01:949 TCP 666 192.168.77...
18:52:02:467	TCP	151	192.168.77.1:2846	192.168.77.2:49160	<49160>	146	[18:52:02:440 TCP 546 192.168.77...
18:52:02:467	TCP	98	192.168.77.1:2846	192.168.77.2:49160	<49160>	44	[2 in-addr arpa gojen]ACKè À`...
18:52:02:467	TCP	54	192.168.77.2:49160	192.168.77.1:2846	<2846>	0	ACK
18:52:02:467	TCP	430	192.168.77.1:2846	192.168.77.2:49160	<49160>	376	[18:52:02:455 TCP 629 192.168.77...
18:52:02:468	TCP	151	192.168.77.1:2846	192.168.77.2:49160	<49160>	146	[18:52:02:462 TCP 899 192.168.77...
18:52:02:468	TCP	98	192.168.77.1:2846	192.168.77.2:49160	<49160>	44	[7.2:49163 telnet>-data 814 [ýú% ...
18:52:02:478	TCP	54	192.168.77.2:49160	192.168.77.1:2846	<2846>	0	ACK
18:52:02:479	TCP	430	192.168.77.1:2846	192.168.77.2:49160	<49160>	376	[18:52:02:468 TCP 151 192.168.77

Gambar 9 *Client* menampilkan kiriman data dari *server*.


Trace Route Filltered Address

Hop	IP Address	Info Time	Host Name
1	192.168.77.2	Reached in : 0 ms	gojen

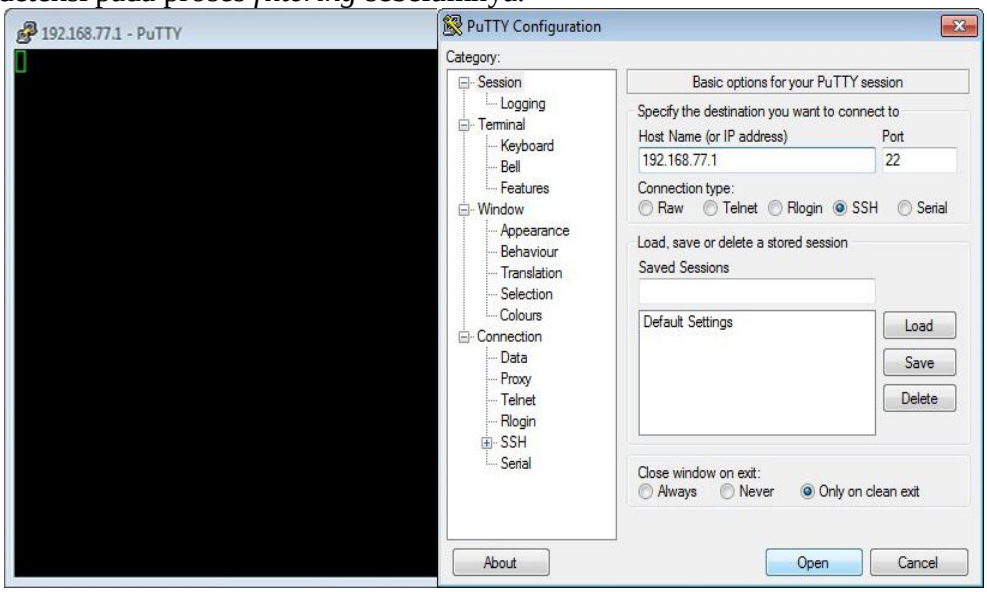
Gambar 10 Hasil *Trace route* ke IP 192.168.77.2.

Percobaan 2 (SSH)

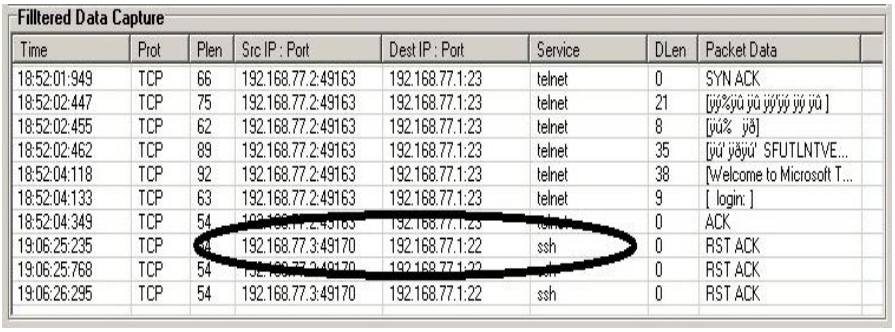
SSH (*Secure Shell Hosting*) adalah aplikasi *remote computer*, sama halnya dengan telnet namun SSH tidak mengirimkan data dalam bentuk *plain text* yang mudah dibaca oleh orang. SSH melakukan pengiriman data *remoting* dalam keadaan telah terenkripsi sehingga membuat SSH lebih aman dibandingkan telnet. Tetapi dengan keamanan yang dimiliki SSH juga akan menjadi masalah apabila ada akses ke komputer yang tidak di inginkan menggunakan SSH. Dengan begitu penggunaannya akan sangat menguntungkan tetapi juga sebaliknya apabila digunakan untuk akses illegal.

Percobaan ke-2 dengan skenario yang sama tetapi menggunakan komputer berbeda dengan IP 192.168.77.3. Dipercobaan ke-2 *tools* yang digunakan adalah SSH. Sama halnya dengan percobaan pertama, SSH akan digunakan untuk mencoba melakukan *remoting* ke komputer *server*. Percobaan ke-2 seperti pada

gambar 11, hasil dari percobaan ini seperti pada gambar 12. Sama seperti halnya tahap terakhir adalah program melakukan *trace route* terhadap hasil yang terdeteksi pada proses *filtering* sebelumnya.



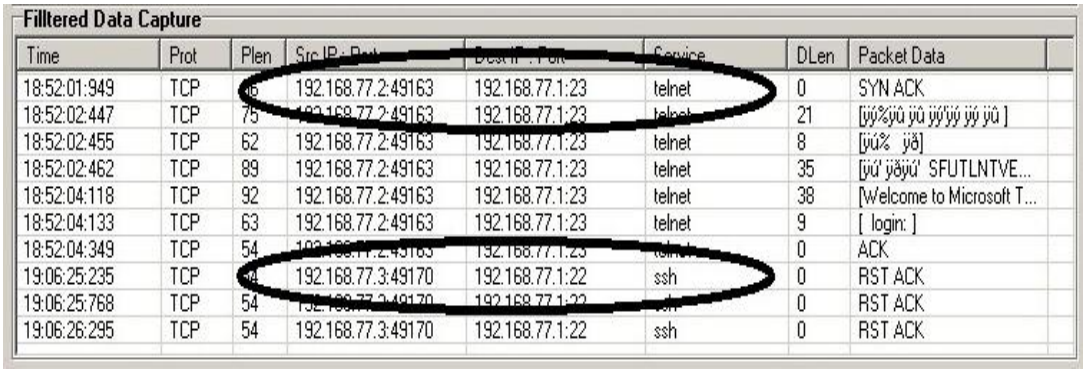
Gambar 11 SSH ke komputer *server*.



Gambar 12 *Server* mendeteksi aktifitas SSH dari IP 192.168.77.3.

Notification

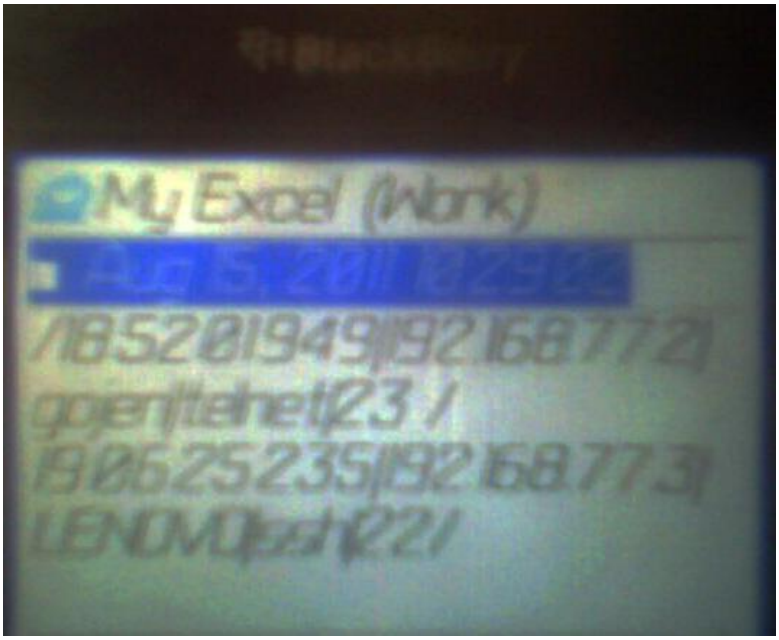
Hasil dari semua uji coba diatas akan memenuhi kuota untuk melakukan pemberitahuan (*Notification*) kepada *administrator* tentang aktifitas yang terdeteksi pada server melalui sms. Aktifitas yang dilaporkan ke *administrator* adalah aktifitas pertama komputer tersebut ke komputer server (Gambar 13). Dari aktifitas tersebut akan dilaporkan melalui SMS dengan format : **waktu, IP, nama komputer, services, port** .



Filltered Data Capture

Time	Prot	Plen	Src IP : Port	Dest IP : Port	Service	DLen	Packet Data
18:52:01:949	TCP	62	192.168.77.2:49163	192.168.77.1:23	telnet	0	SYN ACK
18:52:02:447	TCP	75	192.168.77.2:49163	192.168.77.1:23	telnet	21	[ÿý%ýú ÿû ÿý'ÿý ÿý ÿû]
18:52:02:455	TCP	62	192.168.77.2:49163	192.168.77.1:23	telnet	8	[ÿú% ÿä]
18:52:02:462	TCP	89	192.168.77.2:49163	192.168.77.1:23	telnet	35	[ÿú' ÿäÿú' SFUTLNTVE...
18:52:04:118	TCP	92	192.168.77.2:49163	192.168.77.1:23	telnet	38	[Welcome to Microsoft T...
18:52:04:133	TCP	63	192.168.77.2:49163	192.168.77.1:23	telnet	9	[login:]
18:52:04:349	TCP	54	192.168.77.2:49163	192.168.77.1:23	telnet	0	ACK
19:06:25:235	TCP	62	192.168.77.3:49170	192.168.77.1:22	ssh	0	RST ACK
19:06:25:768	TCP	54	192.168.77.3:49170	192.168.77.1:22	ssh	0	RST ACK
19:06:26:295	TCP	54	192.168.77.3:49170	192.168.77.1:22	ssh	0	RST ACK

Gambar 13 Aktifitas pertama komputer penyerang.



Gambar 14 SMS yang diterima oleh *administrator*.

Pengujian Gangguan dari dalam dan luar

Pada tahap percobaan seperti pada sub-bab sebelumnya, dilakukan juga percobaan tidak hanya dari jaringan dalam (jaringan lokal) tetapi juga dari jaringan luar (Internet). Percobaan yang dilakukan untuk jaringan luar dan dalam, dilakukan dengan cara melakukan *filter* terhadap semua opsi yang ada pada konfigurasi *filtering port*. Percobaan percobaan ini dimaksudkan untuk memastikan bahwa aplikasi dapat melakukan *capture* terhadap semua aliran data yang keluar dan masuk jaringan yang diuji. Hasil dari percobaan ini dapat dilihat pada tabel pengujian gangguan dari dalam dan luar.

Tabel 1 Tabel pengujian gangguan dari dalam dan luar

No.	Gangguan / Alamat IP	Jenis Sambungan	Ter-Capture
1.	Telnet / 192.168.77.2	Sekelas (Lokal)	Ya

2.	SSH / 192.168.77.3	Sekelas (Lokal)	Ya
3.	HTTP / 74.125.235.50	Tidak Sekelas (Internet)	Ya
4.	POP3 / 202.182.170.10	Tidak Sekelas (Internet)	Ya

Pengujian pada operator selular

Pengujian pada operator selular yang dimaksudkan adalah pengujian terhadap biaya dan kecepatan dalam mengirimkan pesan. Pengujian pada operator selular bertujuan agar aplikasi dapat memenuhi kebutuhan sistem yang efisien dan *real time*. Pengujian dilakukan dengan menggunakan beberapa operator berbeda. Hasil dari pengujian ini dapat dilihat pada tabel 2 tabel pengujian pada operator selular.

Tabel 2 Tabel pengujian pada operator selular

No.	Operator Selular	Biaya (Rp)	Waktu (detik)
1.	Telkomsel	150/sms	4
2.	Indosat	150/sms	6
3.	Excelindo	150/sms	5

Analisa sistem

Hasil dari semua percobaan dapat di analisa dari implementasi dan simulasi percobaan adalah tentang proses mekanisme kerja dari sistem secara keseluruhan. Secara keseluruhan hasil analisa terdiri dari konsep dasar yang dibangun untuk menjadi sebuah sistem sesuai dengan yang telah dibahas dan analisa dari hasil percobaan adalah sebagai berikut :

a. *Filtering* (*port filter*) yang dikerjakan program dapat disesuaikan dengan keinginan pengguna, dengan ini sistem *filtering* data dapat lebih fleksibel.
b. *Notification* akan dikirimkan setelah adanya beberapa aktifitas gangguan dari beberapa IP berbeda. Proses ini membuat pengiriman *notification* lebih efisien.
c. Dari point b, dapat dianalisa bahwa setiap IP (Komputer penyerang) dapat melakukan beberapa aktifitas illegal sekaligus yang ter-*filter* sehingga yang dilaporkan tidak berdasarkan jumlah aktifitas gangguan tetapi berdasarkan IP.
d. Proses dari *trace route* hanya dapat mengambil nama komputer (*Host Name*) dari komputer penyerang. Semua proses *hop to hop* dari *trace* dapat terlihat tetapi hanya ujung dari proses yang digunakan untuk *notification*.
e. Proses *filtering* dapat menangkap semua aliran paket data dari dalam (lokal area) maupun dari luar (internet) yang melewati *server*.

5. Kesimpulan

a. Sistem yang telah dibangun *administrator* dapat mengawasi penggunaan *port* ataupun *server* dengan lebih efisien dan *real time* sehingga pengawasan *administrator* tidak lagi harus tergantung pada keberadaan.
b. *Administrator* dapat mengetahui penggunaan *port* secara illegal (percobaan penyerangan / gangguan) melalui teknik *filter port service* pada lalulintas data yang keluar masuk, baik dari jaringan internal maupun eksternal.
c. Kehandalan aplikasi *server* dalam melakukan *capture* terhadap seluruh paket yang melintas membuat proses *filtering* dapat lebih akurat sehingga setiap aktifitas dapat terpantau dengan baik.

d. Kecepatan pengiriman *notification* secara *real time* tergantung kepada operator selular yang digunakan untuk mengirimkan SMS.

Saran

1. Pada penelitian selanjutnya menambahkan informasi dalam *notification* tentang dampak (*impact*) dari gangguan serta status terakhir dari *server* ataupun jaringan yang dikelola.
2. Memperluas kemampuan *filter* agar opsi *filtering* yang lebih banyak dan tidak hanya melakukan *filter* pada *port service* sehingga kemungkinan pendeteksiannya lebih banyak.
3. Penambahan fitur *Whois* yang akan ditambahkan juga sebagai salah satu basis informasi pada *notification* sehingga lebih memudahkan *administrator* dalam menindak lanjuti informasi yang didapatkan.

6. DAFTAR PUSTAKA

Budi Sutedjo Dharma Oetomo, S.Kom.,MM, Eddy Hartono, S.Kom,.MT, Ester Wibowo, BA.,MM.,MT, Samuel Prakoso, S.Kom, 2006, Konsep dan *Aplikasi Pemprograman Client Server dan Sistem Terdistribusi*, Andi Offset (Andi), Yogyakarta.

Dony Ariyus., M.Kom, 2007, *Intrusion Detection System,* Andi Offset (Andi), Yogyakarta.

Edhy Sutanta, 2005, *Komunikasi Data dan Jaringan Komputer,* Graha Ilmu, Yogyakarta.

Jhonathan Lukas, 2006, *Jaringan Komputer*, Graha Ilmu, Yogyakarta.

Melwin Syafrizal, 2005, *Pengantar Jaringan Komputer,* Andi Offset (Andi), Yogyakarta.

William Stallings, 2007, *Komunikasi dan jaringan Nirkabel,* jilid 1, Edisi Kedua, Erlangga, jakarta.