Analisa dan Implementasi Sistem Keamanan Jaringan Komputer dengan Iptables sebagai Firewall Menggunakan Metode Port Knocking

Irwan Sembiring, Indrastanti R. Widiasari , Sujiwo Danu Prasetyo
Fakultas Teknologi Informasi
Universitas Kristen Satya Wacana
Jl. Diponegoro 52-60, Salatiga 50711, Indonesia
Email: irwan@uksw.edu, indrastanti@yahoo.com, danu_netsoul@yahoo.com

Abstrak:
Administrator suatu *firewall* ditantang untuk menyeimbangkan fleksibilitas dan keamanan saat merancang seperangkat aturan yang komprehensif. *Firewall* harus memberikan perlindungan terhadap *malfeasants*, sementara memungkinkan pengguna yang dipercaya untuk dapat terhubung. Sayangnya, tidak selalu mungkin untuk menyaring orang-orang jahat, karena penyaringan berdasarkan alamat IP dan port tidak membedakan pengguna yang terhubung. Orang-orang yang berniat jahat dapat berasal dari alamat IP terpercaya. Port terbuka tetap menjadi kerentanan: mereka mengizinkan koneksi untuk aplikasi tetapi juga dapat berubah menjadi pintu terbuka untuk serangan. Penelitian ini menyajikan suatu sistem keamanan baru, disebut *port knocking*, di mana pengguna terpercaya memanipulasi aturan firewall dengan mengirimkan informasi di seluruh port yang tertutup.
Keywords : *port, knock, port knocking, iptables, firewall*.

1. Pendahuluan

Komponen yang paling penting dalam membangun sebuah jaringan komputer yang aman adalah seputar *firewall* beserta celah-celah keamanan jaringan itu sendiri. Celah-celah keamanan dapat terbuka dikarenakan perangkat komputer tersebut diinstall dengan suatu program aplikasi seperti aplikasi pengolah dokumen, aplikasi email klien (*client*), aplikasi antivirus, aplikasi server klien, dan lain sebagainya yang mungkin dibutuhkan bahkan mungkin juga tidak dibutuhkan. Aplikasi-aplikasi yang diinstall tersebut, terutama yang terhubung dengan jaringan baik itu jaringan lokal maupun jaringan internet akan membuka *port* komunikasi. *Port* komunikasi tersebut merupakan *port* yang ada dalam protokol TCP atau UDP yang merupakan anggota dari *transportation layer* pada standar OSI. Melalui *port* komunikasi tersebut, jaringan internet atau jaringan di luar jaringan komputer dapat menjangkau perangkat komputer. Begitu pula sebaliknya, perangkat komputer lain yang membuka *port* komunikasi tertentu dapat

dijangkau. Komunikasi dapat berjalan dengan lancar, pertukaran informasi menjadi mudah dan kenyamanan dalam berkomputer bertambah dengan terbukanya *port* komunikasi tersebut. Namun, kadang kala kenyamanan ini sering disalahgunakan oleh sebagian orang. *Port* komunikasi tersebut sering dijadikan sebagai celah untuk dimasuki secara ilegal. *Port* yang terbuka digunakan sebagai jalan menuju ke dalam jaringan internal atau ke server-server di dalamnya, kemudian mengacaukannya. *Port* komunikasi yang terbuka secara bebas tersebut juga bisa menjadi salah satu ancaman bagi keamanan data yang ada dalam sistem jaringan komputer. Sangat mungkin penyusup masuk ke dalam komputer, dan bahkan ke seluruh komputer di dalam jaringan komputer jika *port* tersebut dibiarkan terbuka secara bebas. Untuk itulah, *firewall* sangat dibutuhkan di dalam jaringan tersebut. *Firewall* memiliki tugas untuk melakukan pemblokiran terhadap *port-port* komunikasi yang terbuka di dalam sebuah jaringan komputer.

Di dalam *firewall* semua komunikasi yang keluar dan masuk dikontrol. *Port* yang tidak penting dapat diblokir (ditutup) dan *port* yang penting dan berbahaya juga dapat diblokir, sehingga hanya pihak yang diijinkan saja yang boleh masuk melalui *port* tersebut. Cara ini merupakan sistem pengamanan jaringan komputer yang paling efektif dan banyak digunakan. Akan tetapi terkadang pemblokiran yang dilakukan sering menjadi tidak fleksibel, ketika dibutuhkan untuk menjalin komunikasi dengan apa yang ada di dalam jaringan, *firewall* tidak mengijinkannya karena mungkin memang berada pada area yang tidak diijinkan. Padahal komunikasi yang ingin dilakukan sangatlah penting untuk kelancaran kerja. Misalnya melakukan koneksi dengan internet dan butuh mengakses web server melalui SSH untuk memperbaiki konfigurasinya, sementara *port* SSH pada server tersebut dilarang untuk diakses dari internet oleh *firewall*, tentu hal ini akan sangat merepotkan. Untuk menghindari hal-hal semacam ini, ada suatu metode yang sangat efektif yaitu dengan menggunakan metode *port knocking. Port knocking* adalah suatu metode untuk membangun komunikasi antar komputer dari mana pun selama masing-masing komputer tersebut terhubung dalam suatu jaringan komputer, dengan perangkat komputer yang tidak membuka *port* komunikasi apapun secara bebas, tetapi perangkat tersebut masih tetap dapat diakses dari luar, dengan menggunakan suatu format konfigurasi *port* ketukan yang berupa percobaan untuk mengirimkan koneksi pada port ketukan.

2. Kajian Pustaka

Port knocking adalah salah satu cara berkomunikasi pada jaringan komputer, cara yang digunakan adalah berkomunikasi melalui *port* yang tertutup [3]. Dedi Dwianto menjelaskan bahwa *port knocking* merupakan salah satu metode dalam keamanan jaringan komputer yang digunakan untuk membuka sebuah *port* yang tertutup atau membuka akses *firewall* dan mengijinkan *knocker* masuk melalui *port* yang dituju melalui pengiriman paket-paket tertentu ke *port* tujuan. Cara yang digunakan oleh Dedi Dwianto adalah dengan melakukan telnet ke *port* yang tertutup pada server dengan tujuan untuk membuka *port* yang dituju [1].

3. *Firewall* dan *Port Knocking*

Format Ketukan

Format ketukan yang digunakan dalam perancangan sistem adalah format *port* tunggal dengan pemetaan tetap, dan hanya menggunakan tiga *port* ketukan sebagai tujuan pengiriman paket data untuk melakukan ketukan.

Untuk mempermudah penentuan *port* ketukan, maka dibuat aturan pemilihan *port* ketukan sesuai dengan nomor *port* tujuan, digit terakhir pada nomor *port* ketukan merujuk pada nomor *port* tujuan [2]. Sebagai contoh, seorang user ingin mengakses *port* 22, dengan range *port* ketukan yang telah ditentukan yaitu antara *port* 2000 sampai dengan 4000, maka pemilihan *port* ketukan yang digunakan adalah seperti pada Gambar 1.

Nomor *port* ketukan	*Port* tujuan
2000+a,3000+b,4000+c	abc

Gambar 1. Penentuan Format Ketukan

Nomor *port* 2000+a,3000+b,4000+c merupakan nomor *port* tujuan pengiriman paket data yang berfungsi sebagai *port* ketukan. Nomor *port* ketukan menunjukkan *port* tujuan abc yang akan dibuka atau ditutup. Maka ketukan yang dilakukan oleh pengguna jika ingin membuka 22 adalah seperti pada Gambar 2.

2000,3002,4002

Gambar 2. Format Ketukan Untuk Membuka *Port* 22.

Sedangkan jika user ingin menutup *port* 22, maka ketukan yang dilakukan oleh pengguna adalah seperti pada Gambar 3.

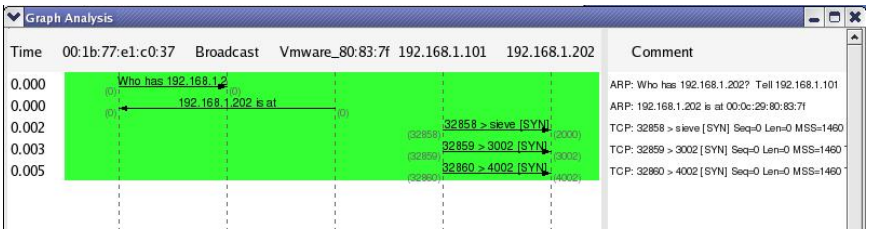
3002,4002,2000

Gambar 3. Format Ketukan Untuk Menutup *Port* 22.

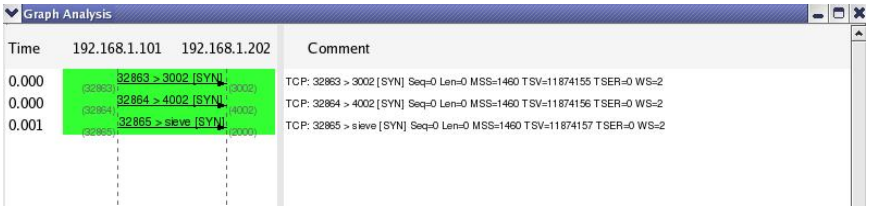
Proses Ketukan Port

Proses pengetukan *port* dilakukan dengan cara mengirimkan paket data dari alamat sumber menuju alamat tujuan. *Header* pada paket data tersebut akan diperiksa terutama

header control list SYN yang digunakan untuk mensikronisasi *sequence number*. Apabila nomor *port* ketukan yang digunakan sebagai *sequence* ketukan sesuai dengan nomor *port* tujuan yang telah ditentukan sebagi nomor *port* ketukan, maka *port* yang dituju akan terbuka/tertutup sesuai dengan format ketukan yang digunakan, seperti ditunjukkan pada Gambar 4 dan Gambar 5.



Gambar 4. Analisa Paket Data Saat Ketukan Pertama



Gambar 5. Analisa Paket Data Saat Ketukan Kedua

4. Implementasi *Port Knocking* pada Sistem Keamanan Jaringan Komputer

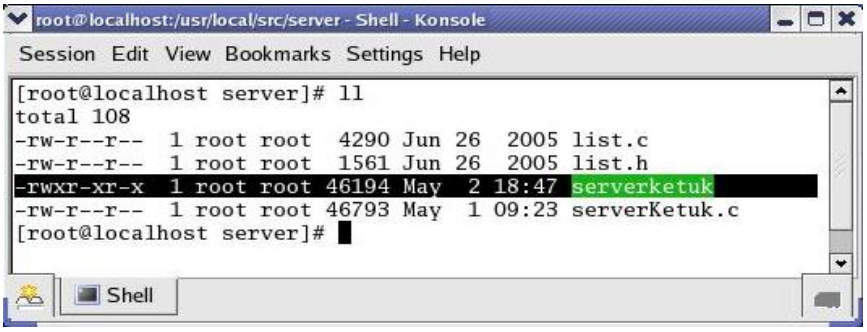
4.1 Instalasi Program

Sebelum program *port knocking* dijalankan, pertama kali dilakukan instalasi program pada komputer server yang berfungsi untuk mendengarkan ketukan *port* dan pada komputer klien yang berfungsi untuk melakukan ketukan *port* terhadap komputer server.

Pada komputer server, instalasi dilakukan terhadap dua buah file, yaitu serverketuk.c dan list.c. Instalasi file pada komputer server juga membutuhkan sebuah *library* yang bernama *lcap* yang berfungsi untuk menangkap paket yang dikirimkan oleh komputer klien pada saat melakukan ketukan *port*. Perintah yang dilakukan pada instalasi ini ditunjukkan seperti pada Gambar 6.

Gambar 6. Instalasi Serverketuk.

Hasil dari perintah yang ditunjukkan pada Gambar 6 adalah terciptanya sebuah file eksekusi yang bernama serverketuk. File serverketuk inilah yang nantinya akan dijalankan dan berfungsi untuk mendengarkan ketukan yang dilakukan oleh komputer klien, seperti terlihat pada Gambar 7.



Gambar 7. File Serverketuk.

Sedangkan pada komputer klien, instalasi dilakukan terhadap dua buah file, yaitu ketukserver.c dan list.c. Instalasi file pada komputer klien tidak membutuhkan *library* khusus seperti pada saat melakukan instalasi file pada komputer server, karena program yang dijalankan pada komputer klien hanya berfungsi melakukan ketukan dengan cara mengirimkan paket ke *port* ketukan komputer server. Perintah yang dilakukan pada instalasi ini ditunjukkan seperti pada Gambar 8.

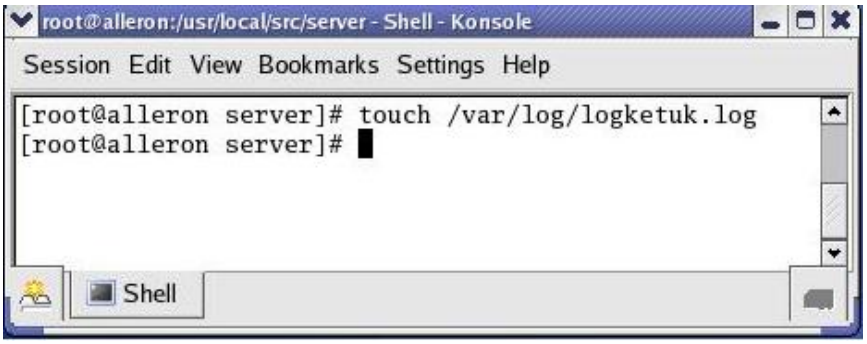
Gambar 8. Instalasi Ketukserver.

Hasil dari perintah yang ditunjukkan pada Gambar 8 adalah terciptanya sebuah file eksekusi yang bernama ketukserver. File ketukserver ini berfungsi untuk melakukan ketukan *port* terhadap komputer server, seperti terlihat pada Gambar 9.

Gambar 9 File Ketukserver.

4.2 Konfigurasi Program

Setelah dilakukan kompilasi terhadap *sourcecode* server dan klien, maka perlu dilakukan konfigurasi agar komputer klien dapat melakukan ketukan dan pada komputer server dapat mendengarkan ketukan, maka dilakukan konfigurasi terlebih dahulu pada komputer server. Untuk melakukan konfigurasi, dibuat dua buah file yaitu file logketuk.log yang berfungsi untuk menyimpan log file yang berupa aktivitas yang terjadi pada server *port knocking.* File logketuk.log dibuat dan diletakkan pada direktori /var/log yang merupakan direktori pada sistem operasi Linux yang berfungsi untuk menyimpan log sistem operasi. Perintah untuk membuat file logketuk.log ditunjukkan seperti pada Gambar 10.

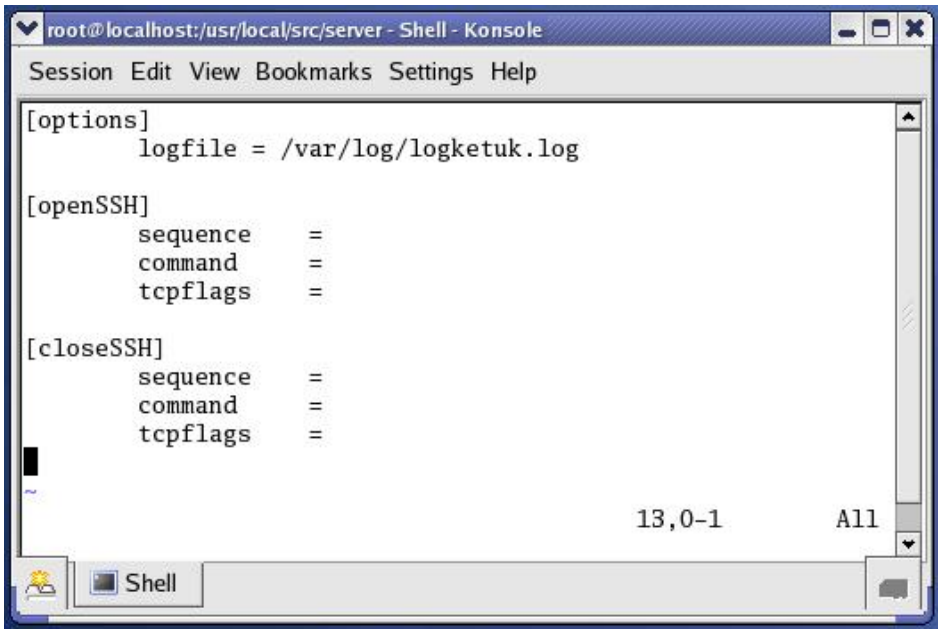


Gambar 10. Membuat File logketuk.log

File yang kedua yang perlu dibuat adalah configketuk.conf yang berfungsi untuk menentukan *port* ketukan dan menentukan perintah *iptables* yang berfungsi sebagai *firewall* dan digunakan untuk membuka atau menutup *port* tujuan. File configketuk.conf dibuat dan diletakkan pada direktori /etc yang merupakan direktori pada sistem operasi Linux yang berfungsi untuk menampung file-file konfigurasi sistem. perintah untuk membuat file configketuk.conf ditunjukkan seperti pada Gambar 11.

Gambar 11. Membuat File configketuk.conf

Setelah tercipta file configketuk.conf, maka langkah selanjutnya adalah mengisi file configketuk.conf tersebut dengan beberapa baris text konfigurasi. Isi dari file konfigketuk.conf ditunjukkan seperti pada Gambar 12.



Gambar 12. Isi File configketuk.conf.

Terdapat tiga bagian utama pada file configketuk.conf, yaitu:

Bagian pertama adalah bagian options yang merupakan bagian yang berfungsi untuk menunjukkan letak file log yang berfungsi untuk mencatat semua aktivitas ketukan.

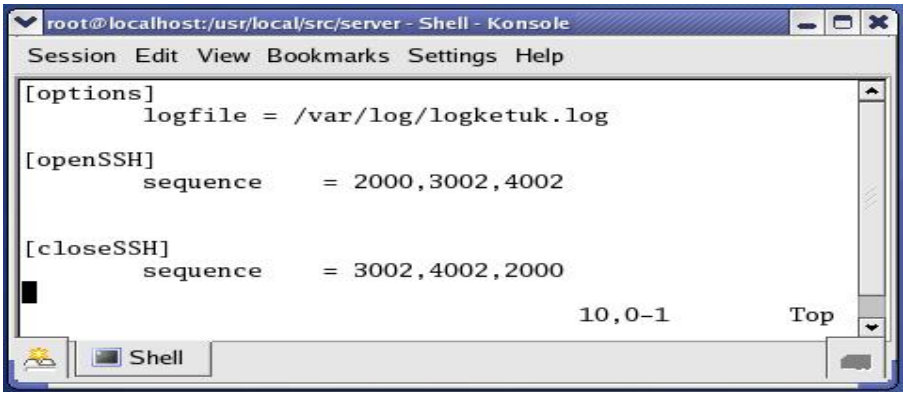
Bagian kedua adalah bagian buka*Port* yang merupakan bagian yang digunakan sebagai konfigurasi untuk membuka *port* tujuan. Pada bagian buka*Port* terdapat tiga buah baris perintah, yaitu:

- *Sequence* : berfungsi untuk menentukan *port* ketukan yang digunakan untuk membuka *port* tujuan.
- *Command* : berfungsi untuk menentukan perintah *iptables* yang dijalankan, yaitu perintah untuk membuka *port* tujuan jika terjadi ketukan pada *port* ketukan.
- *Tcpflags* : menunjukkan *header* paket yang dikirimkan sebagai ketukan *port*.

Bagian ketiga adalah bagian tutup *port* yang digunakan sebagai konfigurasi untuk menutup kembali *port* tujuan. Pada bagian tutup*Port* terdapat tiga buah konfigurasi, yaitu:

- *Sequence* : berfungsi untuk menentukan *port* ketukan yang digunakan untuk menutup *port* tujuan.
- *Command* : berfungsi untuk menentukan perintah *iptables* yang dijalankan, yaitu perintah untuk menutup *port* tujuan jika terjadi ketukan pada *port* ketukan.
- *Tcpflags* : menunjukkan *header* paket yang dikirimkan sebagai ketukan *port*.

Untuk menentukan *port* ketukan, maka ditambahkan baris perintah pada file konfigketuk.conf, seperti ditunjukkan pada Gambar 13.

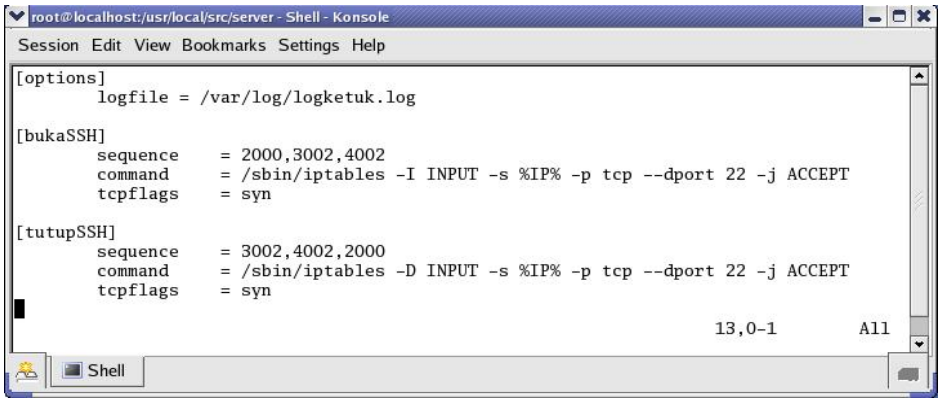


Gambar 13. Format Ketukan File configketuk.conf.

Pada bagian bukaSSH ditentukan tiga nomor *port* ketukan, yaitu *port* 2000, *port* 3002, dan *port* 4002. Jika terjadi ketukan terhadap ketiga *port* tersebut secara berurutan, maka server akan membuka *port* tujuan sesuai dengan perintah *iptables* yang ditentukan.

Pada bagian tutupSSH ditentukan tiga nomor *port* ketukan, yaitu *port* 3002, *port* 4002, dan *port* 2000. Jika terjadi ketukan terhadap ketiga *port* tersebut secara berurutan, maka server akan menutup *port* tujuan sesuai dengan perintah *iptables* yang ditentukan.

Untuk menentukan perintah *iptables* yang akan dijalankan jika terjadi ketukan *port*, maka ditambahkan baris perintah pada file konfigketuk.conf, seperti ditunjukkan pada Gambar 14.



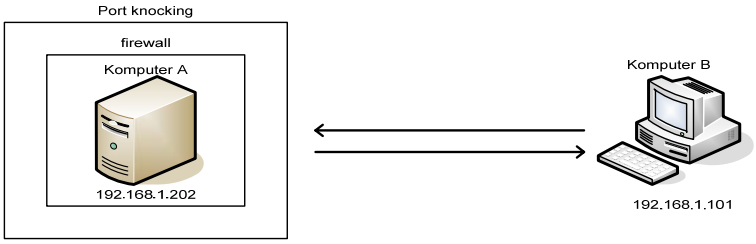
Gambar 14. Perintah *iptables* File configketuk.conf.

Pada bagian bukaSSH, perintah *iptables* yang dijalankan ketika terjadi ketukan pada *port* 2000, *port* 3002, dan *port* 4002, adalah perintah yang bertujuan untuk mengubah aturan *iptables,*yaitu dengan membuka akses terhadap alamat IP komputer klien agar dapat mengakses *port* 22 pada komputer server.

Pada bagian tutupSSH, perintah *iptables* yang dijalankan ketika terjadi ketukan pada *port* 3002, *port* 4002, dan *port* 2000, adalah perintah yang bertujuan untuk mengubah aturan *iptables,* yaitu dengan menutup kembali akses terhadap alamat IP komputer klien sehingga tidak lagi dapat mengakses *port* 22 pada komputer server.

4.3 Penerapan Program Pada Sistem Keamanan Jaringan Komputer

Setelah dilakukan konfigurasi pada file configketuk.conf, maka dilakukan implementasi pada sistem jaringan komputer. Implementasi dilakukan pada dua buah komputer yang saling terhubung pada jaringan komputer. Topologi jaringan komputer yang digunakan sebagai implementasi sistem dapat dilihat pada Gambar 15.

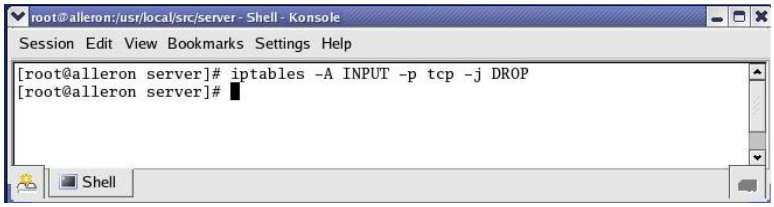


Gambar 15. Topologi Jaringan Saat Implementasi Sistem.

Pada Gambar 15, komputer A memiliki alamat IP 192.168.1.202 dan berfungsi sebagai server *port knocking* yang bertugas untuk menyediakan layanan *port knocking*,

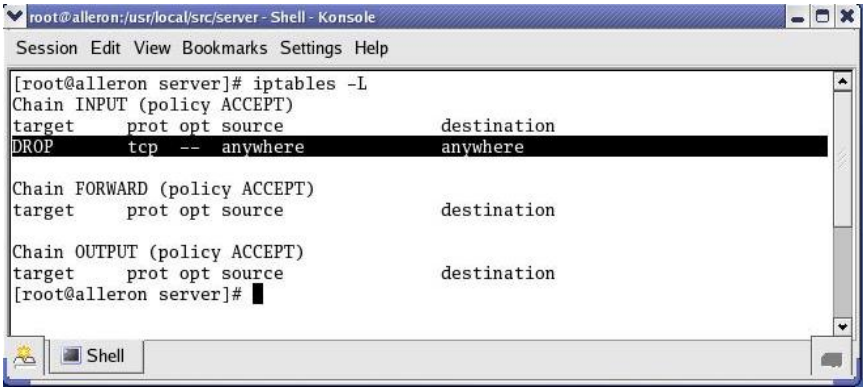
sedangkan komputer B memiliki alamat IP 192.168.1.101 dan berfungsi sebagai klien yang bertugas untuk melakukan ketukan ke komputer server.

Langkah pertama dalam implementasi program adalah menutup semua *port* yang ada pada komputer A dengan menggunakan perintah *iptables* yang bertujuan agar semua paket yang menuju komputer A akan langsung dibuang (DROP) sehingga komputer A tidak dapat diakses dari komputer lain. Perintah yang digunakan untuk menutup semua *port* yang ada pada komputer A ditunjukkan seperti pada Gambar 16.



Gambar 16. Perintah *iptables* Untuk Menutup Semua *Port* Pada Protokol TCP

Hasil dari Gambar 16 adalah tertutupnya semua *port* pada komputer A yang berarti bahwa tidak ada jalan masuk yang terbuka pada komputer A yang bisa diakses oleh komputer lain, seperti terlihat pada Gambar 17. Hal ini dibuktikan dengan melakukan *scanning port* menggunakan Nmap pada komputer A, seperti terlihat pada Gambar 17.



Gambar 17. Chain *iptables* Setelah Semua *Port* Pada Protokol TCP Ditutup.

Gambar 18. Hasil *Scanning Port* Setelah Semua *Port* Pada Protokol TCP Ditutup.

Langkah kedua adalah melakukan konfigurasi pada file configketuk.conf. Konfigurasi yang dilakukan pada file configketuk.conf antara lain adalah:

Menentukan format *port* ketukan untuk membuka *port* 22 (*port* SSH).

Menentukan aturan *iptables* yang berfungsi untuk membuka *port* 22 (*port* SSH) dan mengijinkan alamat IP sumber yang melakukan ketukan *port* untuk mengakses *port* 22 (*port* SSH) jika terjadi ketukan pada *port* 2000,3002,4002, seperti terlihat pada Gambar 19.

```
command     = /sbin/iptables -I INPUT -s %IP% -p tcp --dport 22 -j ACCEPT
```

Gambar 19. Perintah *iptables* Untuk Membuka *Port* 22

Menentukan format *port* ketukan untuk menutup kembali *port* 22 (*port* SSH), seperti terlihat pada Gambar 20.

```
[tutupSSH]
        sequence     = 3002,4002,2000
```

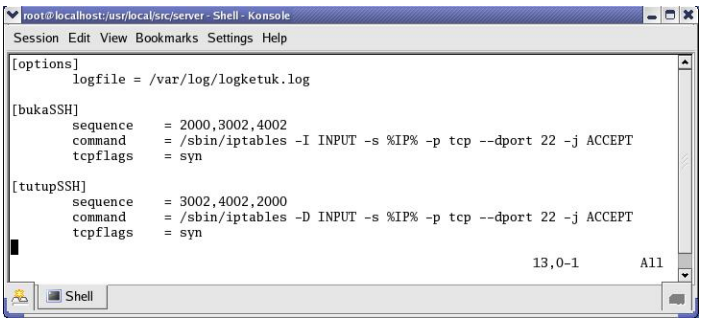
Gambar 20. Format Ketukan Untuk Menutup Kembali *Port* 22

Menentukan aturan *iptables* yang berfungsi untuk menutup kembali *port* 22 (*port* SSH) dan tidak mengijinkan kepada alamat IP manapun juga untuk mengakses *port* 22 (*port* SSH) jika terjadi ketukan pada *port* 3002,4002,2000, seperti terlihat pada Gambar 21.

```
command     = /sbin/iptables -D INPUT -s %IP% -p tcp --dport 22 -j ACCEPT
```

Gambar 21. Perintah *iptables* Untuk Menutup Kembali *Port* 22

Secara keseluruhan isi dari file configketuk.conf adalah seperti terlihat pada Gambar 22.



Gambar 22. Konfigurasi File configketuk.conf Untuk Membuka dan Menutup *Port* 22

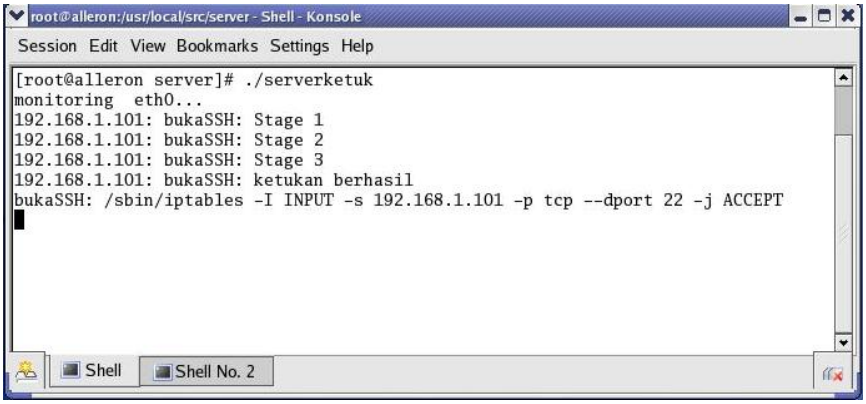
Langkah ketiga adalah menjalankan file serverketuk pada komputer A. Perintah yang digunakan adalah seperti terlihat pada Gambar 23.

Gambar 23. Perintah Untuk Menjalankan Program serverketuk.

Langkah keempat adalah menjalankan program ketukserver pada komputer B yang berfungsi untuk melakukan pengetukan *port* oleh komputer B ke komputer A pada *port* 2000,3002,4002 dengan tujuan agar komputer A membuka *port* 22 dan memberikan ijin agar komputer B dapat mengakses *port* 22 pada komputer A. Perintah yang digunakan adalah seperti terlihat pada Gambar 24.

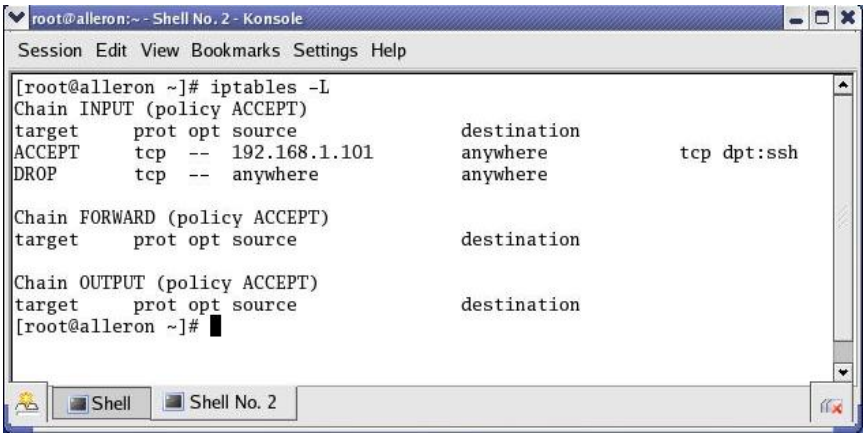
Gambar 24. Menjalankan Program ketukserver Untuk Membuka *Port* 22.

Setelah komputer B melakukan ketukan *port* 2000, 3002, 4002 pada komputer A, maka pada komputer A akan melakukan *monitoring port* apakah *port* yang diketuk sesuai dengan format *port* ketukan bukaSSH, seperti terlihat pada Gambar 25.

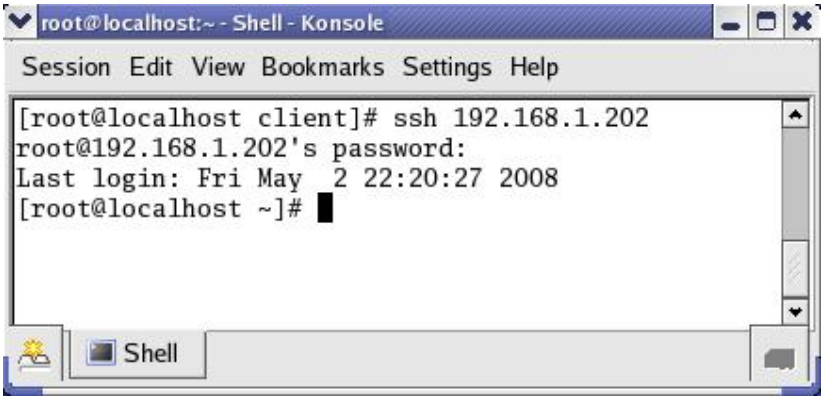


Gambar 25. *Monitoring Port* yang Dilakukan Oleh serverketuk Setelah Ketukan bukaSSH

Pada Gambar 26, jika ketukan yang dilakukan oleh komputer B sesuai dengan format ketukan bukaSSH, maka program serverketuk pada komputer A akan menjalankan perintah *iptables* untuk membuka *port* 22 (*port* SSH) pada komputer A dan hanya akan memberikan akses kepada komputer B untuk menggunakan *port* 22 (*port* SSH) tersebut.



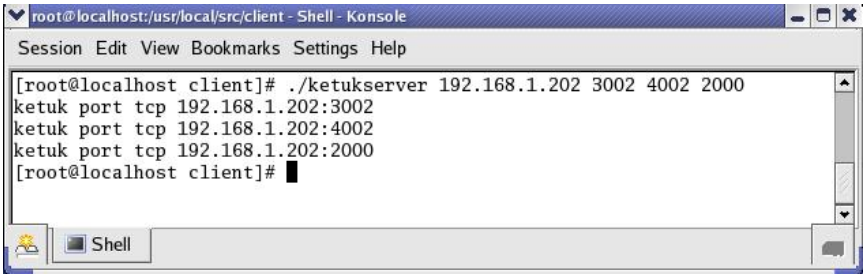
Gambar 26. Chain *iptables* Setelah Terjadi Ketukan Untuk Membuka *Port* 22



Gambar 27. Akses yang Dilakukan Klien ke *Port* 22 yang Telah Terbuka

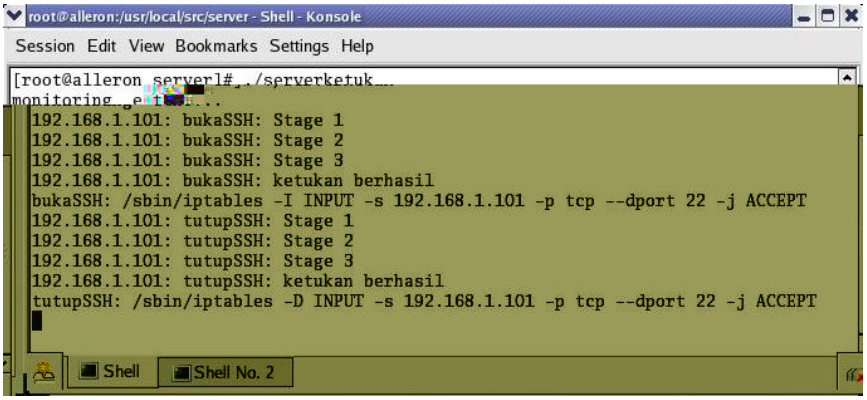
Pada Gambar 27, komputer B melakukan koneksi SSH ke komputer A, setelah komputer B melakukan ketukan *port,* dan komputer A membuka *port* 22 hanya untuk diakses oleh komputer B.

Setelah komputer B selesai mengakses *port* 22 (*port* SSH) pada komputer A, maka langkah kelima yang dilakukan adalah komputer B melakukan ketukan *port* kembali dengan format ketukan yang digunakan untuk menutup kembali *port* 22 (*port* SSH) pada komputer A, ketukan yang dilakukan adalah ketukan terhadap *port* 3002, 4002, 2000. Perintah yang digunakan adalah seperti pada Gambar 28.

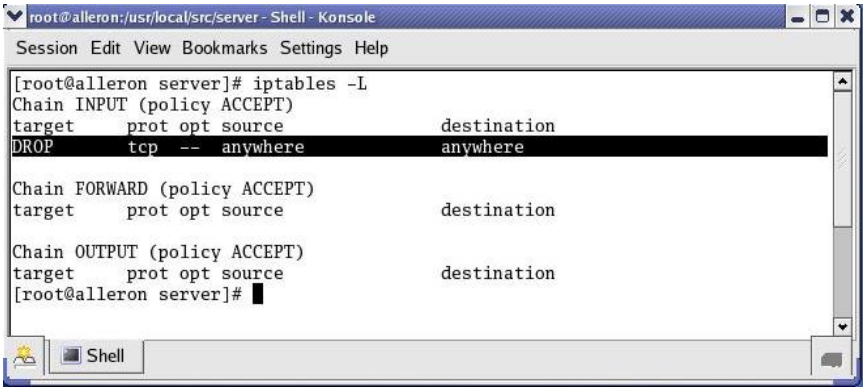


Gambar 28. Menjalankan Program ketukserver Untuk Menutup Kembali *Port* 22

Ketika komputer B melakukan ketukan *port* 3002, 4002, 2000 pada komputer A, maka program serverketuk akan melakukan monitoring *port* apakah *port* yang diketuk sesuai dengan format *port* ketukan tutupSSH. Jika ketukan tersebut benar, maka program serverketuk akan menjalankan perintah *iptables* untuk menutup kembali *port* 22 (*port* SSH) pada komputer A, sehingga tidak ada satupun komputer termasuk juga komputer B yang dapat mengakses kembali *port* 22 pada komputer A, seperti ditunjukkan pada Gambar 29 dan Gambar 30.



Gambar 29. Monitoring serverketuk Setelah Terjadi Ketukan tutupSSH.



Gambar 30. Chain *iptables* Setelah Terjadi Ketukan tutupSSH

5. Penutup

1. Metode *port knocking* sangat berguna diterapkan pada sistem keamanan jaringan komputer.
2. Metode *port knocking* sangat berguna bagi administrator jaringan atau server yang harus mengurusi jaringan atau server secara terus menerus dari mana saja, karena *port knocking* aman digunakan untuk membuat komunikasi antar komputer pada jaringan komputer.
3. Dengan metode *port knocking*, komunikasi antar komputer dapat dilakukan meskipun melalui *port* yang tertutup.
4. Metode *port knocking* cukup baik diimplementasikan pada jaringan yang lalu lintasnya tidak terlalu padat, karena pada jalur lalu lintas jaringan yang padat, *port* yang digunakan sebagai komunikasi harus terbuka dan di akses secara terus menerus.
5. Dengan metode *port knocking* orang lain yang tidak berhak tidak mampu mengetahui apakah ada *port* yang terbuka dan memberikan layanan.

6. Daftar Pustaka

[1] Dwianto, D. (2007). Membuka akses firewall melalui Networks. echo|zine, volume 5 issue 7. Diambil dari: http://ezine.echo.or.id/ezine17/09.txt.

[2] Jeanquier, S. (2006). An analysis of port knocking and single packet authorization. Information Security Group : Royal Holloway College, University of London.

[3] Krzywinski, M. (Jun 16, 2003). Port knocking. Linux Journal. Diambil dari: http://www.linuxjournal.com/article/6811.