

Secure Shell dan OpenSSH

Dalam bekerja di jaringan, tindakan jaga-jaga harus selalu dilakukan. Keamanan adalah hal yang harus selalu diperhatikan. Salah satu contohnya adalah penggunaan SSH dan toolnya, ketika Anda harus bekerja meremote suatu host.

Salah satu keunggulan sistem operasi UNIX adalah praktisnya penggunaan, terutama ketika bekerja dengan jaringan. Untuk bekerja pada suatu mesin, kita tidak harus langsung mendatangi mesin tersebut, melainkan cukup melakukan akses remote ke mesin tersebut. Akses remote itu mengandalkan aplikasi seperti telnet dan aplikasi berbasis teks. Dengan demikian, tidak memerlukan GUI seperti yang sering ditemukan dalam *remote desktop* saat ini. Remote desktop terlalu besar, dan tidak dapat digunakan untuk meremote mesin yang tidak menyalakan X, seperti *server*.

Dengan melakukan akses remote ke server menggunakan aplikasi seperti telnet misalnya, kita melakukan autentikasi ke mesin, dan setelah itu, kita akan memiliki akses ke shell mesin tersebut. Dengan demikian, kita dapat mempergunakan tool-tool mesin tersebut untuk melakukan apa yang kita inginkan. Setelah semuanya selesai, kita melakukan *logout* dan kembali ke mesin kita.

Bertahun-tahun yang lalu, telnet adalah aplikasi yang sangat populer untuk keperluan akses remote tersebut. Mesin yang akan di-remote menjalankan daemon telnet dan mesin yang akan me-remote cukup memanggil aplikasi *client* telnet.

Salah satu kelemahan terbesar telnet adalah mentransmisikan segalanya dalam *modus clear text*. Hal ini jelas-jelas memiliki kelemahan. Kita bisa buktikan dengan berbagai program seperti *sniffit* dan *ethereal*. Berikut ini adalah contoh bahwa password yang dilewatkan melalui telnet dalam ditangkap dengan mudah.

Jalankan telnet *server* dan lakukan koneksi ke server. Sebagai root, jalankan pula *ethereal* dan pastikan *ethereal* mendengar pada *network device* yang benar. Setelah *ethereal* siap menangkap paket, loginlah melalui telnet dan berikanlah perintah *ls*. Setelah itu,

*logout*lah. Telnet server sudah bisa dimatikan sekarang.

Kembalilah ke *ethereal* dan hentikanlah penangkapan paket. Lihatlah hasil penangkapan kita dengan memilih salah satu frame paket yang tertangkap dan klik kanan memilih menu *Follow TCP Stream*. Di sebuah dialog yang tampil, kita bisa melihat segala apa yang diketikkan selama sesi telnet, termasuk username dan password.

Anda juga bisa mempergunakan aplikasi lain seperti *tcpdump* dan *sniffit* untuk menangkap paket yang dikirimkan lewat jaringan.

Kita telah melihat bagaimana telnet ditangkap dengan mudah. Anda bisa mencoba cara serupa, namun komunikasi dilakukan lewat *ssh*. *Ethereal* hanya menangkap paket terenkripsi dan informasi penting, termasuk password tidak akan terbaca.

Hal ini sangatlah penting. Informasi yang ditangkap mungkin tidak hanya password. Bisa saja informasi lain yang lebih penting.

Secure Shell dan OpenSSH

Usaha untuk enkripsi data selama pengiriman telah diperhatikan oleh para ahli jaringan. Pada tahun 1992/1993 misalnya, sebuah usaha untuk mempergunakan secure shell telah diwujudkan dalam proyek SSH.

Sama seperti halnya telnet, Secure shell adalah program yang digunakan untuk memasuki suatu komputer dalam jaringan dan menjalankan program di mesin lain. Namun, SSH memiliki beberapa kelebihan yaitu juga dapat mengopi file suatu host ke host lain secara aman (terenkripsi). SSH adalah pengganti yang jauh lebih baik dari telnet, *rlogin*, *rsh*, *rcp* dan *sftp*.

Sebagai tambahan, Secure shell juga dapat digunakan untuk membuat koneksi X lebih aman, dan secara umum, dapat dimanfaatkan untuk meningkatkan keamanan jaringan secara umum. Karena dapat

melakukan transfer file, Secure shell juga dapat digunakan sebagai pengganti *rsync*.

Secara umum, berikut ini adalah apa yang dapat secure shell lindungi:

- IP spoofing, dimana suatu host mengirim paket dan berpura-pura sebagai host yang dipercaya (*trusted host*). Dengan *host authentication*, Secure shell dapat mencegah atau meminimasi terjadinya kejahatan ini.
- IP Source routing
- DNS Spoofing
- Penangkapan password yang dikirim dalam clear text
- Serangan pada koneksi X.

Kerugian penggunaan Secure shell adalah lebih lambat daripada telnet karena harus dienkripsi terlebih dahulu. Enkripsi membutuhkan kerja CPU dan memori yang besar. Namun, SSH2 telah menyempurnakan banyak hal sehingga boleh dikatakan, perbedaannya kini sangatlah kecil.

Seiring dengan berjalannya waktu, proyek SSH yang standarnya (*Secsh*) disubmit ke IETF tersebut memiliki banyak implementasi. Proyek yang awalnya bebas mulai terikat. Beberapa pihak juga membangun implementasi *Secsh* sendiri. Jadilah kita mengenal berbagai macam implementasi *ssh*.

Salah satu yang paling populer adalah OpenSSH. OpenSSH adalah implementasi bebas dari standar secure shell. Penggunaan algoritma yang dipatenkan juga dihindari. Kini, OpenSSH adalah proyek secure shell yang terdapat dalam hampir semua distribusi Linux (dan secara default, umumnya juga telah diaktifkan).

OpenSSH mendukung protokol 1.3, 1.5 dan 2.0. OpenSSH juga mendukung kompresi data (sangat berguna bagi yang melakukan remote connection melalui modem).

Protokol Secure Shell

Setelah melewati perjalanan yang panjang, kita mengetahui terdapatnya dua protokol mayor SSH. Yaitu protokol versi 1 dan versi 2. Kedua protokol tersebut sama-sama melakukan pengiriman data secara aman, hanya algoritma dan enkripsi yang digunakan berbeda. Kedua protokol tersebut berbeda, dan tidak kompatibel. Umumnya, yang digunakan pada distro Linux modern adalah protokol versi 2.

Berikut ini, kita akan melihat perbedaan algoritma enkripsi yang dipergunakan pada protokol secure shell 1 dan 2.

Cipher	SSH1	SSH2
DES	Ya	Tidak
3DES	Ya	Ya
IDEA	Ya	Tidak
Blowfish	Ya	Ya
Twofish	Tidak	Ya
Arcfour	Tidak	Ya
Cast128-cbc	Tidak	Ya

Sementara, untuk autentikasi, SSH 1 mempergunakan RSA dan SSH 2 mempergunakan DSA.

Satu catatan yang perlu diperhatikan sehubungan dengan implementasi OpenSSH yang bebas adalah tidak tersedianya kode-kode yang dipatenkan. Para developer OpenSSH berusaha untuk menjaga OpenSSH bebas dari kode-kode paten. Oleh karena itu, beberapa algoritma terpaksa tidak dimasukkan ke dalam OpenSSH.

Pada SSH1, OpenSSH hanya mendukung 3DES dan Blowfish. Pada SSH2, hanya

3DES, Blowfish, Cast128, Arcfour, dan AES yang didukung. IDEA tidak didukung karena merupakan algoritma yang dipatenkan. Sementara, RSA, karena patennya telah berakhir, maka tidak ada larangan untuk digunakan.

Lebih lanjut tentang protokol. SSH1 datang dalam dua varian, 1.3 dan 1.5. Keduanya menggunakan algoritma kriptografi asimetris RSA untuk key negotiation, dan mempergunakan algoritma kriptografi simetris untuk penyembunyian data yaitu 3DES dan Blowfish. Protokol SSH1 mempergunakan pengecekan CRC yang sederhana untuk memeriksa integritas data.

SSH2 ditulis untuk menghindari paten RSA (telah berakhir) dan menyempurnakan pemeriksaan integritas data, disamping berbagai alasan teknis lain. Penggunaan DSA dan DH digunakan untuk menghindari paten. Dan, sebagai penyempurnaan pemeriksaan berbasis CRC, algoritma HMAC digunakan untuk penyembunyian informasi.

Untuk penggunaan banyak algoritma kriptografi, OpenSSH mengandalkan OpenSSL.

Tool-tool OpenSSH

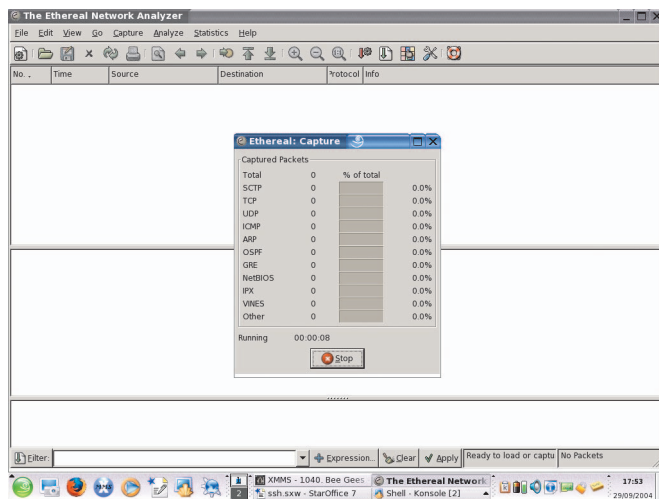
OpenSSH datang dengan berbagai tool, sebagai berikut:

- sshd (8). Program ini adalah server secure shell. SSHD akan mendengar koneksi dari client, melakukan autentikasi, dan melayani permintaan lain-lain client. Kita hanya perlu menjalankan satu server untuk menikmati koneksi secara aman. Konfigurasinya (terletak di

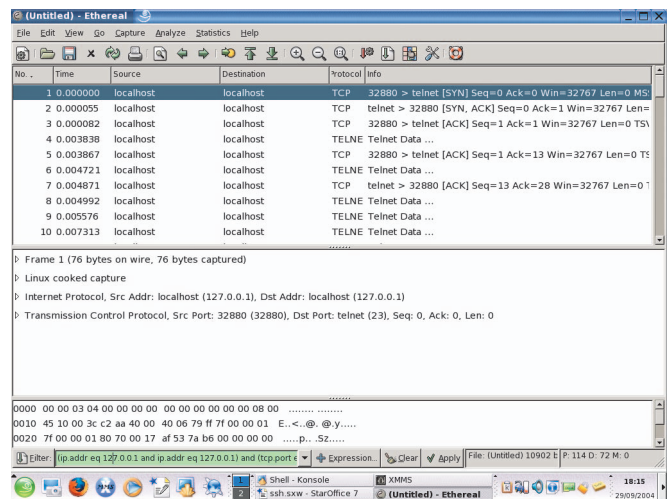
/etc/ssh/sshd_config) juga mudah sekali untuk dimengerti.

- Ssh (1). Program ini adalah client OpenSSH. Gunakan program ini seperti halnya telnet. Program ini juga memiliki nama lain, yaitu slogin. File konfigurasi program ini terletak pada /etc/ssh/ssh_config ataupun ~/.ssh/config.
- Scp (1). Program ini adalah pengopi file (pengganti rcp) yang dapat mengopi file dari satu host ke host lain secara aman. Pengopian tidak harus dari host yang digunakan ke host lain. Bisa juga dari host lain ke host lainnya. Misal, kita berada di host A, kita dapat mengopi file dari host B ke host C, tanpa masalah.
- Ssh-keygen (1). Program ini dapat digunakan untuk membuat key bagi suatu host. Key yang dibuat berdasarkan algoritma RSA ataupun DSA (tergantung protokol).
- Ssh-agent (1). Authentication agent.
- Ssh-add (1). Program ini dapat digunakan untuk mendaftarkan key baru ke agent.
- Sftp-server (8). Program ini berfungsi sebagai ftp server yang aman.
- Sfp (1). Program ini adalah client ftp yang aman.
- Ssh-keyscan (1). Program ini berguna untuk mendapatkan / scan berbagai public key.
- Ssh-keysign (8). Berguna sebagai pembantu untuk hostbased authentication (SSH protokol 2).

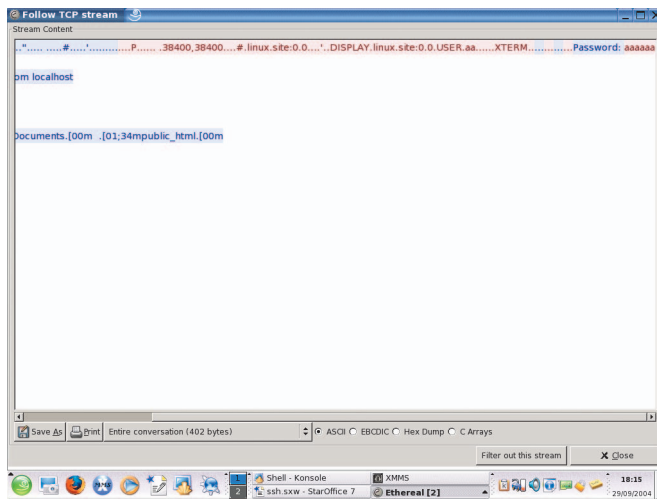
Tool-tool tersebut mudah sekali untuk digunakan. Sshd misalnya. Umumnya, kita



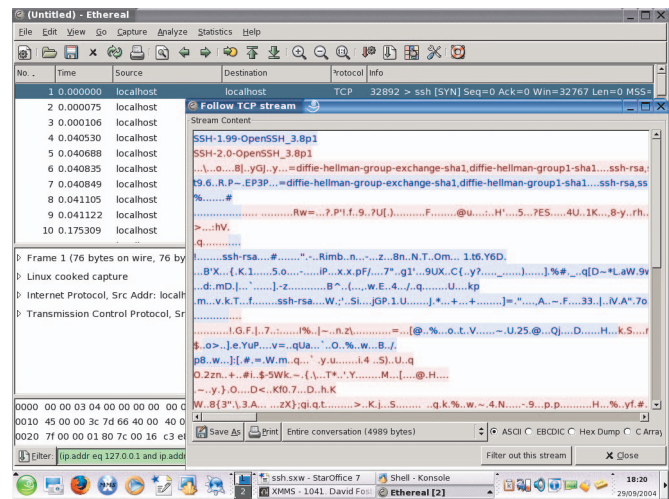
Menangkap paket dengan Ethereal.



Paket yang tertangkap.



Clear Text password dapat dilihat dengan mudah.



Penggunaan SSH, lebih aman!

tidak perlu menjalankan server ssh ini secara manual karena sudah dimasukkan sebagai servis ketika booting. Hampir semua distro telah melakukannya. Dengan demikian, untuk menjalankan sshd, umumnya Anda cukup memberikan perintah:

```
/etc/init.d/sshd restart
```

atau, pada SUSE, lebih umum ditemukan:

```
rcsshd restart
```

Sementara, penggunaan ssh sendiri, sebagai ssh client, juga benar-benar mudah. Anda hanya perlu memberikan perintah seperti berikut ini:

```
ssh user@host
```

Sebagai contoh:

```
ssh nop@192.168.0.1
```

Apabila suatu host belum dikenal, maka ssh akan memberikan pesan kepada kita. Pesan tersebut juga berfungsi sebagai konfirmasi untuk memasukkan suatu host sebagai host yang dikenal.

SSH benar-benar memperhatikan aspek keamanan di sini. Sebagai contoh, katakanlah Anda memiliki sebuah server dengan IP 192.168.0.1. *Workstation* Anda, yang memiliki IP 192.168.0.50 misalnya, telah menganggap server tersebut sebagai host yang telah dikenal. Suatu hari, Anda menginstall ulang server 192.168.0.1 tersebut. Ketika Anda melakukan koneksi ssh lagi dari 192.168.0.50, Anda mendapatkan pesan bahwa host tujuan mungkin telah dicompromise, seperti berikut ini:

```

@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@    WARNING: REMOTE HOST
IDENTIFICATION HAS CHANGED! @
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
IT IS POSSIBLE THAT SOMEONE IS
DOING SOMETHING NASTY!
Someone could be eavesdropping
on you right now (man-in-the-
middle attack)!
It is also possible that the RSA
host key has just been changed.
The fingerprint for the RSA key
sent by the remote host is
b1:00:2b:eb:52:ce:62:4b:fb:20:
af:94:bd:d5:f9:19.
Please contact your system
administrator.
Add correct host key in /root/.
ssh/known_hosts to get rid of
this message.
Offending key in /root/.ssh/
known_hosts:1
RSA host key for 192.168.0.1 has
changed and you have requested
strict checking.
Host key verification failed.

```

Sebenarnya, ini hanyalah tindakan jaga-jaga dari SSH. Informasi host telah berubah. Cara mudahnya adalah dengan menghapus entri IP server tersebut di file `~/.ssh/known_hosts`. Setelah itu, lakukanlah koneksi kembali. Cara ini bukan cara satu-satunya, namun bekerja.

Program pengopi file, scp, juga mudah

sekali digunakan. Cukup berikan saja perintah dengan pola berikut ini:

```
scp <SRC> <DST>
```

misal:

```
scp file1 nop@192.168.0.1:~
```

Perintah ini akan mengopi file1 ke home directory nop di 192.168.0.1.

```
scp nop@192.168.0.1:~/file1 .
```

Perintah ini akan mengopi file1 ke home directory nop di 192.168.0.1 ke direktori aktif.

```
Scp nop@192.168.0.1:~/file1
nop@192.168.0.50:~/
```

Perintah ini akan mengopi file1 ke home directory nop di 192.168.0.1 ke home directory nop di 192.168.0.50.

Program ftp client yang secure, sftp, bisa Anda gunakan sama seperti layaknya Anda menggunakan ftp biasa.

PuTTY: Front End dan SSH client di Windows dan Linux

PuTTY tersedia untuk Windows dan Linux. Namun, di untuk Linux, GNOME terminal atau Konsole sudah sangat mencukupi. Dapatkan PuTTY di <http://www.chiark.greenend.org.uk/~sgtatham/putty/>.

Kita telah membahas beberapa aspek dari SSH, OpenSSH dan pengunanya. Mulai sekarang, lakukanlah selalu tindakan jaga-jaga. Bukankah mencegah lebih baik dari mengobati?

Noprianto (noprianto@infolinux.co.id)