

# Mengenal SSH Lebih Lanjut

**SSH tidak hanya sekedar dapat digunakan untuk menjalankan perintah di komputer lain dalam lingkungan kerja yang aman. SSH sendiri memiliki banyak sekali fungsi, dan dengan mengenal SSH lebih baik, kita bisa lebih memanfaatkannya untuk menjaga sistem kita dari pihak yang tidak bertanggung jawab, ataupun untuk dapat bekerja lebih mudah.**

**S**ecure shell saat ini merupakan program standar untuk menjalankan perintah di komputer lain dengan aman. Hampir semua distro Linux telah memaketkan SSH ke dalam distronya. Dan, umumnya, SSH Daemon pun telah dijalankan pada saat *booting*. Dengan demikian, pengguna di komputer lain dapat terhubung ke suatu komputer lainnya dengan mudah, tanpa harus menjalankan service tambahan. Apabila distro Anda masih belum memaketkan SSH, download-lah di website OpenSSH di [www.openssh.com](http://www.openssh.com).

SSH dibangun untuk mengatasi kelemahan aplikasi semacam telnet yang mengirimkan data dalam plain-text, sehingga dapat disniff oleh pihak yang tidak bertanggung jawab. Pada kenyataannya, tidak hanya telnet yang digantikan oleh SSH. Namun, juga termasuk rlogin, rsh, rexec, dan ftp. Hari demi hari, SSH juga kian bertumbuh semakin lengkap. Saat ini, banyak sekali service di Linux yang mengirimkan data menggunakan bantuan SSH.

Melihat sebentar ke belakang, SSH.com memulai pengembangan SSH original dan melisensikannya di bawah lisensi *open source*. SSH kemudian menuai sukses dibandingkan dengan deslogin ataupun telnet-ssl (telnet yang melewati koneksi aman) karena kecanggihannya dan kemudahan penggunaannya. Sayangnya, beberapa tahun kemudian, SSH.com membangun SSH2 dan melisensikannya di bawah lisensi yang berbeda. Lisensi yang hanya mengizinkan penggunaan bebas untuk lingkungan kerja non komersial. SSH2 ini sebenarnya sangat baik karena dibangun ulang mulai dari level protokol. Source code-nya pun ditulis ulang.

Tidak puas dengan kondisi lisensi demikian, orang-orang di belakang Open-

BSD kemudian mengambil kode SSH1 dan mengimplementasikan semua fitur SSH2 dan melisensikannya di bawah lisensi open source, dan menyebutnya sebagai OpenSSH. OpenSSH inilah yang kita pakai di Linux. Sebenarnya, selain Linux, berbagai sistem operasi juga telah memasukkan OpenSSH ke dalam instalasi default, seperti \*BSD, Solaris dan MacOS.

Satu hal yang perlu dicatat di sini adalah SSH sendiri merupakan protokol yang memiliki banyak implementasi. Ada implementasi yang proprietary dan implementasi yang free. Masing-masing memiliki kelebihan dan kekurangannya sendiri.

Bagaimana SSH bekerja? Setiap koneksi akan dienkripsi dengan *session key* yang akan berubah secara periodik. Beberapa pendekatan kriptografi dilakukan untuk memastikan bahwa kita benar-benar terhubung ke server yang diinginkan dan tidak ada orang ditengah-tengah yang mencuri data. Setiap server memiliki pasangan key public/private dan setiap kali koneksi, client akan memeriksa apakah key dari server telah berubah atau tidak. Apabila berubah, client akan menolak untuk melakukan koneksi karena ada kemungkinan percobaan tindak kejahatan.

Lebih lanjut, berikut ini, kita akan membahas beberapa isu yang cukup menarik seputar penggunaan SSH.

## SSH1 versi 1

Saat ini, SSH2 sudah cukup banyak digunakan. OpenSSH yang terinstal di sistem umumnya mendukung kedua versi SSH tersebut, namun beberapa implementasi secara default akan berbicara menggunakan protokol versi 2. Berikut ini, kita akan membahas sedikit seputar protokol versi 1.

Sebelum pembicaraan terjadi, SSH client dan server harus memulai koneksi yang

aman. Dalam proses ini, ada proses pertukaran *key*, *password*, dan data lainnya. Berikut ini adalah proses-proses bagaimana koneksi aman dilakukan pada SSH1:

- Client menghubungi server. Ini adalah proses yang sangat sederhana, di mana client mengirimkan request ke port 22 (port standar SSH) milik server.
- Client dan server saling melihat protokol yang digunakan. Protokol direpresentasikan dalam karakter ASCII. Untuk melihat versi protokol yang digunakan, Anda juga bisa memanfaatkan telnet ke port 22.
- Client dan server kemudian menggunakan protokol berbasis paket (packet-based protocol). Data di-transport dalam protokol TCP.
- Server memberikan informasi dirinya kepada client dan menyediakan beberapa parameter session. Lebih detail, server akan mengirim informasi berikut ini dalam kondisi tidak terenkripsi:
  - Host key server, untuk membuktikan identitas host server.
  - Server key server, untuk membantu membuat koneksi aman.
  - Urutan 8 byte random, yang disebut check bytes, Client harus mengikutsertakan check bytes ini pada respon berikutnya, atau server akan menolak respon dari client.
  - Daftar enkripsi, kompresi dan metode autentikasi yang didukung oleh server.
- Client mengirimkan secret key ke server. Secret key ini juga merupakan session key pertama. Di sini, akan terdapat enkripsi ganda untuk memastikan hanya server yang dituju yang dapat membaca apa yang dikirimkan.
- Kedua pihak, client dan server akan

```
strict checking.
Host key verification failed.
```

## Program yang datang bersama SSH

Berikut ini adalah program yang datang bersama paket openssh:

- scp, untuk melakukan pengopian file antar host dengan aman.
- Sftp, sebagai ftp client yang aman.
- Slogin, program untuk melakukan remote login. Program ini ditujukan untuk menggantikan rlogin.
- Ssh, program secure shell, berguna untuk menjalankan perintah di host lain dengan aman.
- Ssh-add, program yang berguna untuk menambahkan identitas RSA dan DSA kepada agen autentikasi, ssh-agent.
- Ssh-agent, program yang berguna untuk menjaga *private key* yang digunakan untuk autentikasi menggunakan public key.
- Ssh-copy-id, berguna untuk menginstal identity.pub ke authorized\_keys komputer lain.
- Ssh-keyconverter, berguna untuk mengkonversi public dan private key RSA yang digunakan pada SSH1 ke format yang digunakan oleh SSH2.
- Ssh-keygen, program untuk membuat key autentikasi dan melakukan pengaturan lebih lanjut.
- Ssh-keyscan, program yang berguna untuk mendapatkan public ssh host key host-host lain.
- Sshd, ssh server.
- Sftp-server, subsystem ssh yang berfungsi sebagai sftp server. Program ini tidak ditujukan untuk dipanggil langsung, melainkan sebagai subsystem sshd.

Dengan satu paket OpenSSH, Anda sudah dapat melakukan berbagai pekerjaan remote dengan cara yang sangat mudah dan sekaligus aman. Menjalankan perintah secara remote, meng-copy file antar-host dan melakukan manajemen sistem lanjutan dapat dilakukan dengan hati tenang.

Berikut ini adalah contoh penggunaan scp dan sftp, sebagai cara untuk transfer file yang aman.

## SCP

Program ini berguna untuk meng-copy file

antar host. Kita dapat mengopi dari lokal ke remote ataupun remote ke lokal.

Contoh pengopian file lokal ke remote:

```
$ scp -r a nop@192.168.0.1:/tmp/
Password:
```

Contoh pengopian file remote ke lokal:

```
$ scp -r nop@192.168.0.1:/tmp/a .
Password:
```

Sebagai catatan, opsi -r digunakan untuk pengopian secara rekursif.

## SFTP

SFTP adalah program yang berfungsi sebagai ftp client yang secure. Cara penggunaannya sama seperti halnya dengan penggunaan ftp client biasa. Sebagai contoh:

```
sftp nop@192.168.0.1
Connecting to 192.168.0.1...
Password:
sftp> ls -l
-rwxr--r-- 1 mei users
32814 Jul 8 19:36 0300469981.
jpg
drwxr-xr-x 2 nop users
80 Jan 15 2005 Documents
drwxr-xr-x 5 nop users
120 Apr 22 22:58 GNUstep
lrwxrwxrwx 1 nop users
24 Jan 3 2005 KEANTSYSTEMS
drwxr-xr-x 5 nop users
416 May 1 01:27 OpenOffice.
org1.1
-rwxr--r-- 1 nop users
600 Jan 21 09:20 PUTTY.RND
drwxr-xr-x 2 nop users
48 Jan 3 2005 bin
drwx--x--x 13 nop users
488 Jan 21 17:27 profile
drwxr-xr-x 2 nop users
80 Jan 3 2005 public_html
sftp> bye
```

Sebagai catatan, sftp menyediakan perintah dasar untuk bekerja dengan file. Perintahnya tidak akan selengkap ftp client konvensional.

## X over SSH

Istilah lain yang umum digunakan adalah SSH X forwarding. Seperti diketahui, X adalah sistem kompleks yang mengizinkan

kan remote display dari komputer satu ke komputer lain melewati jaringan secara transparan. Sayangnya, X sendiri cukup terbuka untuk diserang oleh kalangan penjahat cyber. Di sinilah peran SSH. Setiap data yang melewati SSH akan dienkripsi sebelum dikirimkan.

Bagaimana X forwarding bekerja? Sebuah ssh client meminta X forwarding pada saat terhubung ke SSH server (diasumsikan X forwarding telah diaktifkan di client). Apabila server mengizinkan X forwarding pada koneksi tersebut, maka proses login akan berjalan normal. Server akan mengatur beberapa langkah tambahan yang transparan bagi user. Sebagai tambahan dalam menangani sesi terminal, server akan mengatur dirinya sebagai X proxy server yang berjalan pada mesin remote dan akan mengatur variabel DISPLAY di remote shell yang mengacu kepada proxy X tersebut. Secara sederhana, SSH server kemudian akan berpura-pura sebagai X server.

Pada saat Anda menjalankan X client, client tersebut akan terhubung ke X proxy. Proxy kemudian akan bertindak seolah-olah seperti X server yang sesungguhnya dan akan meminta kepada SSH client untuk bertindak sebagai client X proxy untuk kemudian akan melakukan koneksi ke X server.

Client dan server SSH kemudian akan bekerjasama untuk melewatkan informasi protokol X bolak balik melalui SSH di antara dua sesi X.

Sebagai catatan, untuk mengaktifkan X forwarding pada client, editlah file /etc/ssh/ssh\_config dan berikan nilai Yes untuk opsi ForwardX11.

Bagi Anda yang lebih suka menggunakan VNC daripada X, maka seperti halnya X forwarding, kita juga bisa menerapkan hal yang sama, koneksi melalui SSH. Beberapa pihak memiliki argumen yang mengatakan bahwa VNC forwarding masih lebih aman dibandingkan dengan X forwarding. Satu catatan, selalu gunakan VNC forwarding apabila Anda harus membuka data rahasia lewat jaringan. Informasi mengenai VNC bisa didapatkan di realvnc.com.

Pembahasan mengenai SSH X forwarding akan dilakukan secara terpisah.

## Verbose untuk informasi lebih lanjut

Apabila Anda membutuhkan informasi

memulai proses enkripsi dan mulai menyelesaikan proses autentikasi.

- Koneksi aman terbentuk.

## File konfigurasi yang digunakan

- SSH server akan membaca file konfigurasi server global yang umumnya disimpan di `/etc/ssh/sshd_config`. Tanpa file ini, SSH server tidak akan berjalan.
- SSH client akan membaca file konfigurasi client global yang umumnya disimpan di `/etc/ssh/ssh_config`. Konfigurasi per user akan disimpan di dalam `~/.ssh/config`.

## Kompresi

Koneksi SSH mendukung fitur kompresi, yang akan mengompres data sebelum dikirim sehingga akan memperkecil ukuran data yang dikirim. Ini sangat berguna ketika Anda mengirimkan data memanfaatkan dialup, misalnya. Untuk mengaktifkan kompresi, berikan opsi `-C`, sebagai contoh:

```
$ ssh -C nop@192.168.0.1 ls /opt
Password:
MozillaFirefox
OpenOffice.org
gnome
kde3
lftp
```

Kompresi mencakup semua data. Mulai dari stdin, stdout, stderr dan semua data lainnya. Kompresi yang digunakan adalah kompresi yang sama dengan yang digunakan oleh gzip. Seperti disebutkan sebelumnya, pada koneksi yang lambat, kompresi akan sangat membantu. Namun, pada jaringan yang cepat, kompresi bisa saja memperlambat.

## Masalah known host

Pertama kali Anda mencoba melakukan koneksi ke suatu host, informasi host yang Anda kunjungi akan disimpan ke dalam file `~/.ssh/known_hosts`. Daftar host yang dikenal juga bisa disimpan di dalam `/etc/ssh/known_hosts` (oleh sysadmin Anda).

Sebagai contoh, komputer B (192.168.0.110) melakukan koneksi ke Komputer A (192.168.0.1). Komputer A kemudian akan disimpan sebagai known host komputer B. Suatu hari, komputer A rusak dan digantikan dengan komputer baru, masih menggunakan IP 192.168.0.1. Ketika

komputer B ingin melakukan koneksi kembali ke komputer A, maka pesan peringatan akan ditampilkan, dan ssh akan menolak melakukan koneksi. Program ssh akan memberitahu bahwa kemungkinan akan ada usaha untuk menyerang sistem (man-in-the-middle attack).

Hal ini adalah wajar karena komputer baru yang menggantikan posisi A memiliki informasi IP yang dimiliki oleh komputer yang digantikan, sementara key-nya berbeda. Apabila Anda yakin bahwa hal tersebut bukan kesalahan, maka Anda bisa menghapus file `~/.ssh/known_hosts` ataupun menghapus entri host komputer A (192.168.0.1) di file tersebut.

Contoh file `~/.ssh/known_hosts`:

```
192.168.0.1 ssh-rsa AAAAB3NzaC1yc2EAAAABIwAAAIEAv6kCmEtZ0x6cuuC+Ld0FKSUv3+o+8hiI9N1oy8p65o6EQK+YC0aPw2/AND6UMSJAZHdK1bvDHXm2i7Umj0cRFh+TjXgvzE5SQSYjxFrGo4RB1LGzAa3kRow4d/givET8X1soARtrWvFzgYXWwZ9coAZnYsPB0RUd1orG3E1pZ40=
192.168.0.28 ssh-rsa AAAAB3NzaC1yc2EAAAABIwAAAIEA5s+zA5NPBcspMPKN4+tswkFbPJVJq4symo9EmcP7HWDGRGjarepNawV+KIDkFk7wWYjegkpVbc8GRZT3IcXfB4D2/8GqPS2obpC3knHECaJVKbCakk1yp4iWUyoR72mT+KWbbaEH
```

```
6E2ZmPWCQ96P0mZKGM9QqSYu2n67Sr7UU=
```

Contoh pesan kesalahan:

```
ssh nop@192.168.0.1
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@    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
61:43:82:c6:45:87:6a:9a:0a:d8:6a:41:21:1a:af:42:
Please contact your system administrator.
Add correct host key in /home/nop/.ssh/known_hosts to get rid of this message.
Offending key in /home/nop/.ssh/known_hosts:1
RSA host key for 192.168.0.1 has changed and you have requested
```



Situs Resmi OpenSSH.




lebih lanjut tentang apa yang terjadi ketika suatu aksi dilakukan, Anda bisa meminta SSH untuk menampilkan informasi secara lebih verbose. Cukup berikan opsi `-v` pada saat menjalankan SSH. Pemberikan `-vv` dan `-vvv` akan meningkatkan tingkat verbositas.

Sebagai contoh, berikut ini adalah penggunaan opsi `-v` pada saat penggunaan SSH:

```
nop@nop:/home/DATA/NOP/home>
ssh -v nop@192.168.0.110
OpenSSH_3.9p1, OpenSSL 0.9.7e
25 Oct 2004
debug1: Reading configuration
data /etc/ssh/ssh_config
debug1: Applying options for *
debug1: Connecting to
192.168.0.110 [192.168.0.110]
port 22.
debug1: Connection established.
debug1: identity file /home/
nop/.ssh/identity type -1
debug1: identity file /home/
nop/.ssh/id_rsa type -1
debug1: identity file /home/
nop/.ssh/id_dsa type 2
```

```
debug1: expecting SSH2_MSG_KEX_
DH_GEX_REPLY
debug1: Host '192.168.0.110' is
known and matches the RSA host
key.
debug1: Found key in /home/nop/
.ssh/known_hosts:3
debug1: ssh_rsa_verify:
signature correct
debug1: SSH2_MSG_NEWKEYS sent
debug1: expecting SSH2_MSG_
NEWKEYS
debug1: SSH2_MSG_NEWKEYS
received
debug1: SSH2_MSG_SERVICE_REQUEST
sent
debug1: SSH2_MSG_SERVICE_ACCEPT
received
[...DIPOTONG...]
debug1: Authentications that can
continue: publickey,keyboard-
interactive
debug1: Next authentication
method: keyboard-interactive
Password:
```

```
debug1: Authentication succeeded
(keyboard-interactive).
debug1: channel 0: new [client-
session]
debug1: Entering interactive
session.
debug1: Sending environment.
debug1: Sending env LANG =
en_US.UTF-8
Last login: Mon Jul 18 15:20:36
2005 from nop.keant.com
Have a lot of fun...
nop@nop:~> ls
bin Documents GNUstep
OpenOffice.org1.1.4 public_html
temp TransGaming_Drive
nop@nop:~>
```

SSH adalah sistem yang sangat kompleks. Pembahasan kali ini hanya menyentuh sangat sangat kecil bagian dari SSH. Tapi, di luar itu, SSH adalah sistem yang sangat menarik. Berbagai solusi bisa dibangun memanfaatkan SSH. Selamat mencoba!   
Noprianto (noprianto@infolinux.co.id)

## Fakultas Ilmu Komputer Universitas Indonesia bekerjasama dengan KPLI Jakarta



Distro untuk Warnet

### Persyaratan kompetisi:

1. Kompetisi ini terbuka untuk perorangan atau kelompok, tidak terbatas kepada institusi maupun lembaga riset.
2. Hasil karya belum pernah diikutsertakan dalam lomba lainnya.
3. Distro yang dibuat dapat berupa *live CD* atau *installer*.
4. Distro yang dibuat harus merupakan turunan dari Debian atau turunannya (Knoppix, Ubuntu, dll).
5. Untuk mendukung perangkat lunak bebas (*free software*), panitia menganjurkan untuk sebisa mungkin menghindari penggunaan aplikasi yang tidak sesuai dengan ketentuan FOSS (*proprietary license* tidak dianjurkan).
6. Hasil karya yang diserahkan kepada panitia merupakan milik panitia dengan tetap menghargai bahwa hak cipta merupakan milik peserta lomba sepenuhnya.
7. Hasil karya yang diperlombakan dibawah lisensi GNU Public License (GPL).
8. Bagi pemenang pertama kompetisi ini diberikan kesempatan untuk mendemonstrasikan hasil karyanya pada acara seminar Debian Conference II.
9. Hasil karya pemenang I, II, III akan dipublikasikan di majalah komputer.
10. Keputusan panitia tidak bisa diganggu gugat.

Pendaftaran dilakukan dengan cara mengirimkan email ke: **kompetisi\_debconf@mhscs.ui.ac.id**, dan mencantumkan: **nama kelompok**, **nama anggota**, dan **nama universitas/instansi**. Petunjuk lebih lanjut akan diberikan setelah pendaftaran.

Hasil karya diterima panitia paling lambat tanggal **19 November 2005**.

keterangan lebih lanjut kunjungi <http://debconf.vism.org>

menangkan total hadiah  
**6 juta rupiah**

be creative....

debian untuk semua