SSH

# What is?

Secure Shell disingkat menjadi SSH adalah protokol komunikasi jaringan(Network) yang memungkinkan dua komputer untuk saling berkomunikasi dan berbagi data. Fitur yang melekat pada ssh adalah bahwa komunikasi antara dua komputer dienkripsi, artinya cocok untuk digunakan pada jaringan yang tidak aman. Selain itu, SSH sering digunakan untuk "log-in" ke sebuah sistem operasi dari jarak jauh dan melakukan operasi pada komputer jarak jauh(remote) tetapi juga dapat digunakan untuk mentransfer data.

Sebuah program didalam komputer kita disebut juga SSH client, untuk terhubung ke sebuah Layanan/service (server) dan mentransfer data ke/dari penyimpanan sebuah layanan/service yang tersedia yang memungkinkan client melakukan transfer ini dan beberapa sistem operasi seperti; Mac Os X dan Linux Distribution punya kemampuan default ini.

SSH client biasanya akan mendukung SCP ( [Secure Copy](https://translate.google.com/website?sl=auto&tl=en&hl=en&u=http://en.wikipedia.org/wiki/Secure_copy) ) dan/atau SFTP ( [SSH File Transfer Protocol](https://translate.google.com/website?sl=auto&tl=en&hl=en&u=http://en.wikipedia.org/wiki/SSH_File_Transfer_Protocol) ) untuk mentransfer data; Namun kebanyakan client cenderung merekomendasikan penggunaan SFTP daripada SCP tetapi keduanya akan berfungsi dengan baik. SSH memungkinan kami menyediakan layanan dengan akses terEnkripsi untuk berbagai operating system, ssh sering digunakan dalam High Performance computing community karena alasan aman dan bisa diandalkan.

# Terminology

Istilah-istilah dalam dunia IT sangat banyak dan membingungkan, maka dari itu kenalilah istilah-istilah tersebut terlebih dulu agar tidak bingung ditengah jalan, berikut beberapa islitah-istilah dasar dalam mengenal SSH;

1. **Service**; (Service) Layanan layaknya program khusus yang berjalan di belakang layar/Background komputer dan dapat digunakan oleh komputer lain tanpa harus diinstal terpisah. Misalnya, server web adalah contoh layanan yang memungkinkan berbagi halaman web. Meskipun tidak selalu memiliki tampilan grafis yang jelas.
2. **Host**: Host adalah komputer apa pun yang dapat menjadi tuan rumah/Host bagi aplikasi atau layanan yang berguna untuk komputer lain. Jadi, komputer Anda juga bisa menjadi host jika berjalan beberapa program yang membantu Anda, ponsel, atau komputer lain.
3. **Local**: local computer adalah komputer yang Anda gunakan secara langsung atau yang digunakan oleh perangkat lunak yang berjalan di komputer Anda. Setiap komputer dapat menyebut dirinya "localhost" untuk merujuk pada dirinya sendiri.
4. **Remote**: remote computer adalah komputer yang tidak Anda hadapi secara fisik atau tidak dapat Anda gunakan langsung secara fisik – komputer yang berapa jauh dihadapan anda. Ini adalah komputer yang berada di lokasi yang jauh dari tempat Anda berada – Computer jarak jauh(remote).

# Command-Line;

Secure Shell (SSH) adalah protokol keamanan yang digunakan untuk mengakses dan mengontrol perangkat jarak jauh melalui jaringan. Berikut adalah beberapa dasar-dasar perintah SSH pada Linux:

* Menggunakan SSH untuk Mengakses Perangkat Jarak Jauh:

ssh [username]@[alamat\_ip\_atau\_hostname]

Contoh:

* ssh john@192.168.1.100

Perintah di atas akan menghubungkan komputer lokal ke perangkat jarak jauh dengan alamat IP 192.168.1.100 menggunakan akun pengguna "john".

* Menggunakan Port Tertentu: Jika SSH dijalankan di port selain 22 (port default), kamu bisa menentukan port tersebut menggunakan opsi -p:

ssh -p [port] [username]@[alamat\_ip\_atau\_hostname]

Contoh:

* ssh -p 2222 john@192.168.1.100

Perintah di atas akan menghubungkan ke perangkat jarak jauh di alamat IP 192.168.1.100 dengan port SSH diatur ke 2222.

* Mengirim Perintah Melalui SSH: Kamu bisa menggunakan SSH untuk menjalankan perintah di perangkat jarak jauh tanpa perlu masuk ke shell jarak jauh sepenuhnya. Gunakan opsi -c:

ssh [username]@[alamat\_ip\_atau\_hostname] [perintah]

Contoh:

* ssh john@192.168.1.100 ls

Perintah di atas akan menjalankan "ls" (list files) di perangkat jarak jauh dan menampilkan hasilnya di terminal lokal.

* Transfer File dengan SCP: SCP (Secure Copy Protocol) memungkinkan kamu untuk mengirim dan menerima file secara aman melalui SSH. Untuk mengirim file dari lokal ke perangkat jarak jauh:

scp [file\_lokal] [username]@[alamat\_ip\_atau\_hostname]: [tujuan\_di\_perangkat\_jarak\_jauh]

Contoh:

* scp data.txt john@192.168.1.100:/home/john/documents/

Perintah di atas akan mengirim file "data.txt" dari komputer lokal ke direktori "/home/john/documents/" di perangkat jarak jauh.

* Menjalankan Perintah sebagai Pengguna Lain: Jika kamu ingin menjalankan perintah sebagai pengguna lain di perangkat jarak jauh, gunakan opsi -l:

ssh -l [username\_tujuan] [alamat\_ip\_atau\_hostname] [perintah]

Contoh:

ssh -l alice 192.168.1.100 ls

Perintah di atas akan menjalankan "ls" (list files) di perangkat jarak jauh sebagai pengguna "alice".

* Mengakhiri Sesi SSH: Setelah kamu selesai menggunakan SSH, cukup ketik "exit" untuk keluar dari sesi SSH dan kembali ke terminal lokal.
* **Melihat verifikasi private key yang digunakan**: ssh-add -l -E sha256
* **Menyimpan dan restart SSH:** sudo systemctl restart sshd && echo "OK"
* **Membuat kunci ssh**: ssh-keygen -t ed25519 -f ~/.ssh/<folder yang ingin disimpan>

contoh: ssh-keygen -t ed25519 -f ~/.ssh/id\_rsa

# Push git automated:

Untuk menghindari dimintanya username dan password setiap kali kamu melakukan git push ke GitHub, kamu bisa menggunakan otentikasi melalui SSH. Berikut adalah langkah-langkah untuk mengatur otentikasi SSH dengan GitHub:

* Cek Kunci SSH yang Ada atau Buat Kunci SSH Baru: Pertama, periksa apakah sudah ada kunci SSH di komputermu atau belum. Di Linux atau macOS, cek direktori ~/.ssh/ dengan perintah:

ls ~/.ssh/

Jika sudah ada file id\_rsa dan id\_rsa.pub, berarti sudah ada kunci SSH. Jika belum ada, kamu bisa membuat kunci SSH baru dengan perintah:

ssh-keygen -t rsa -b 4096 -C "email\_anda@example.com"

Pastikan mengganti "[email\_anda@example.com](mailto:email_anda@example.com)" dengan alamat email GitHub yang digunakan.

* Tambahkan Kunci SSH ke Akun GitHub: Salin isi dari file id\_rsa.pub (kunci publik) dengan perintah berikut:
* cat ~/.ssh/id\_rsa.pub

Kemudian, buka GitHub di browsermu, masuk ke akunmu, dan buka pengaturan akun. Pilih "SSH and GPG keys" dan tambahkan kunci publik yang baru saja disalin dengan klik tombol "New SSH key".

* Tes Koneksi SSH: Sekarang, tes apakah koneksi SSH sudah berhasil dengan GitHub. Jalankan perintah berikut:

ssh -T git@github.com

Jika koneksi berhasil, kamu akan melihat pesan "Hi [username]! You've successfully authenticated..." di terminal.

* Ubah Remote URL pada Repositori Git: Terakhir, kamu perlu mengganti URL remote pada repositori git dengan menggunakan URL SSH. Pergi ke direktori repositori git yang ingin kamu push, kemudian jalankan perintah berikut untuk mengubah URL remote:

git remote set-url origin git@github.com:username/nama-repo.git

Gantilah username dengan username GitHub kamu, dan nama-repo dengan nama repositori yang sesuai.

Istilah-istilah:

* ssh-keygen for generating secure keys
* ssh-agent and ssh-add for securely storing private keys
* scp and sftp to securely copy public key files during initial use of a server

Setelah langkah-langkah di atas dilakukan, seharusnya kamu tidak lagi diminta untuk memasukkan username dan password saat melakukan git push ke GitHub. Git akan menggunakan otentikasi SSH yang sudah diatur sebelumnya untuk mengenali dan mengautentikasi akses ke akun GitHub kamu secara otomatis.