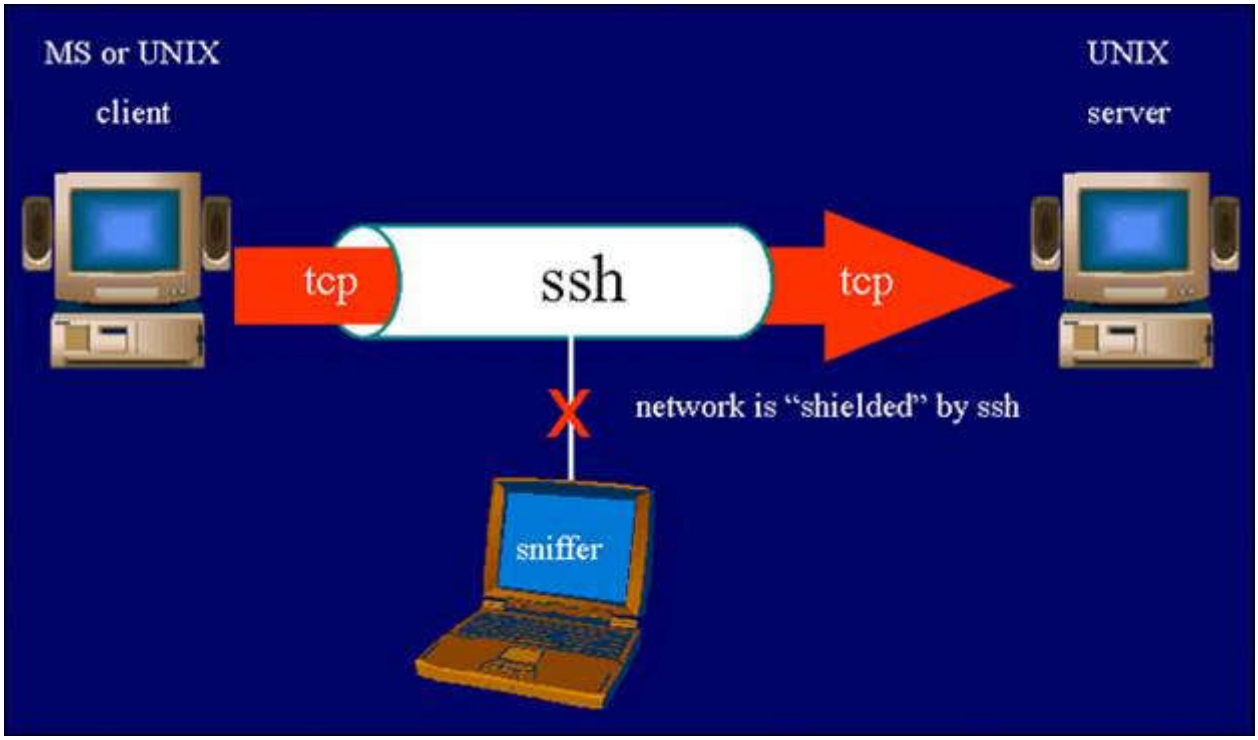
Bài 7. Quản trị từ xa

Có nhiều công cụ cho phép quản trị máy chủ Linux từ xa, trong đó phổ biến nhất là OpenSSH, Puppet, và Zentyal.

1. OpenSSH

SSH (Secure Shell) là một giao thức mạng dùng để thiết lập kết nối mạng một cách bảo mật. SSH hoạt động ở lớp trên trong mô hình phân lớp TCP/IP. Các công cụ SSH (như là OpenSSH, …) cung cấp cho người dùng cách thức để thiết lập kết nối mạng được mã hoá để tạo một kênh kết nối riêng tư.



SSH (Secure Shell) là dịch vụ hỗ trợ việc quản lý Linux/Unix từ xa qua mạng, hỗ trợ chứng thực với 3 cách khác nhau: bằng password, bằng public key, và thông qua một hệ thống quản lý account tập trung (Kerberos, LDAP, ...)

Các chương trình trước đây như telnet, rlogin không sử dụng phương pháp mã hoá. Vì thế bất cứ ai cũng có thể nghe trộm thậm chí đọc, sửa đổi được toàn bộ nội dung của phiên làm việc bằng cách sử dụng một số công cụ đơn giản. Sử dụng SSH là biện pháp hữu hiệu bảo mật dữ liệu trên đường truyền từ hệ thống này đến hệ thống khác.

Mặc định dịch vụ SSH sử dụng cổng 22

SSH làm việc thông qua 3 bước:

- **Bước 1. Định danh host** – xác định định danh của hệ thống tham gia phiên làm việc SSH:
  - Máy chủ gửi khóa public tới máy trạm
  - Máy trạm sinh ra một khóa ngẫu nhiên và mã hóa khóa này bằng khóa public do máy chủ gửi tới và gửi trả lại máy chủ
  - Máy chủ giải mã khóa do máy trạm gửi tới bằng khóa private của mình và nhận được khóa của máy trạm

- **Bước 2. Mã hoá** – thiết lập kênh làm việc mã hoá.
- **Bước 3. Chứng thực** – xác thực người sử dụng có quyền đăng nhập hệ thống:
  - Được thực hiện trên kênh trao đổi bảo mật
  - Mỗi định danh và truy cập của người dùng được cung cấp theo nhiều cách khác nhau:
    - Chứng thực **rhosts**: chỉ kiểm tra định danh máy trạm được liệt kê trong file **rhosts** (theo DNS và địa chỉ IP)
    - Chứng thực mật khẩu: rất thông dụng (**dùng tài khoản của hệ thống**)
    - Chứng thực RSA: sử dụng **ssh-keygen** và **ssh-agent** để chứng thực các cặp khóa

OpenSSH cung cấp khá nhiều tính năng để giúp cho việc truyền thông giữa 2 host trở nên an toàn. Dưới đây là một số tính năng nổi bật:

- Khả năng mã hoá mạnh bởi việc sử dụng chuẩn mã hoá 3 DES và Blowfish: Cả 2 chuẩn mã hoá trên đều được cung cấp miễn phí và sử dụng rộng rãi ở nhiều nước trên thế giới. 3DES cung cấp khả năng mã hoá chứng thực thời gian. Blowfish cung cấp khả năng mã hoá nhanh hơn. Cũng như những chuẩn mã hoá khác cả 2 chuẩn nêu trên đều cung cấp khả năng mã hoá các dữ liệu trước khi nó được đưa vào đường truyền một cách an toàn.
- Khả năng chứng thực mạnh bởi việc sử dụng các cơ chế Public Key, mật khẩu một lần (One-Time Password - OTP), Kerberos, có tác dụng bảo vệ chống lại tính dễ tổn thương trong quá trình chứng thực bởi việc khai thác và sử dụng các kỹ thuật như: IP Spoof, DNS Spoof, Fake Router…Có 4 phương pháp chứng thực được Open SSH sử dụng :
  - Chỉ chứng thực Public Key
  - Sự chứng thực host bởi việc sử dụng Public Key kết hợp với .rhost
  - Sự chứng thực dựa trên OPT kết hợp với s/key
  - Sự chứng thực dựa trên cơ chế Kerberos
- Mã hoá giao thức X11 cho việc sử dụng X Window: Mã hoá dữ liệu trong quá trình sử dụng X Window giữa 2 host. Được sử dụng để chống lại những cuộc tấn công từ xa nhằm vào xterm như Snooping, Hjacking…
- Mã hoá cho quá trình chuyển đổi cổng (Port Forwarding): Cho phép quá trình chuyển đổi các port TCP/IP tới một hệ thống khác thông qua một kênh được mã hoá. Nó được sử dụng cho những giao thức Internet chuẩn không cung cấp khả năng mã hoá dữ liệu trên đường truyền như: SMTP, POP, FTP, Telnet…
- Đại diện chuyển tiếp cho những đăng nhập vào các mạng đơn: Một Key chứng thực của người dùng có thể và thường được lưu giữ trên PC của họ, nó có thể trở thành một trạm đại diện chứng thực. Khi người sử dụng hệ thống truy cập từ một hệ thống mạng khác. Kết nối của họ sẽ được chuyển tới cho trạm đại diện chứng thực này. Nó có tác dụng cho phép người sử dụng truy cập đến hệ thống của bạn một cách an toàn từ bất kỳ hệ thống nào.
- Nén dữ liệu: Cung cấp khả năng nén dữ liệu một cách an toàn. Nó rất có ý nghĩa trên những hệ thống mạng không được nhanh.
- Chứng thực chung cho Kerberos và Andrew File System bằng cách sử dụng Ticket: Những người sử dụng Kerberos và AFS sẽ được cung cấp một password chung để sử dụng và truy cập 2 dịch vụ trên trong một thời gian nhất định.

OpenSSH không phải là một chương trình. Nó là một bộ các chương trình kết nối an toàn:

- **OpenSSH Client** (ssh): Chương trình được sử dụng cho các đăng nhập từ xa. Với sự an toàn và mã hoá trong mỗi phiên đăng nhập ở mức độ cao.

- **Secure Copy Program** (scp): Được sử dụng cho việc copy file từ xa, copy các file từ các host khác nhau trên Internet. Nó hỗ trợ username và password.
- **Secure File Transfer Program** (sftp): Được sử dụng để phục các yêu cầu FTP một cách an toàn.
- **OpenSSH Deamon** (sshd): OpenSSH Server chạy ở chế độ daemon.

Cài đặt

Cài đặt cả OpenSSH client và server đều dễ dàng. OpenSSH client thường được cài đặt theo mặc định. Để cài đặt OpenSSH client trên Ubuntu, chạy lệnh sau:

```
sudo apt install openssh-client
```

Để cài đặt OpenSSH server, cùng các tệp phụ trợ, chạy lệnh:

```
sudo apt install openssh-server
```

Các files cấu hình cơ bản của dịch vụ OpenSSH

Tất cả các tập tin cấu hình của ssh đều nằm trong thư mục **/etc/ssh**. Ta sẽ khảo sát sơ lược một số file trong thư mục ssh này:

- **moduli**: Chứa một nhóm Diffie-Hellman được sử dụng cho việc trao đổi khóa Diffie-Hellman, nó thực sự quan trọng để xây dựng một lớp bảo mật ở tầng vận chuyển dữ liệu. Khi các khóa được trao đổi với nhau bắt đầu ở một phiên kết nối SSH, một share secret value được tạo ra và không thể xác định bởi một trong hai bên kết nối, giá trị này sau đó sẽ được dùng để cung cấp chứng thực cho host.
- **ssh_config**: file cấu hình mặc định cho SSH client của hệ thống.
- **sshd_config**: File cấu hình cho **sshd** deamon.
- **ssh_host_dsa_key**: DSA private key được sử dụng với **sshd** deamon.
- **ssh_host_dsa_key.pub**: DSA public key được sử dụng bởi **sshd** deamon.
- **ssh_host_key**: RSA private key được sử dụng bởi **sshd** deamon cho phiên bản 1 của giao thức SSH.
- **ssh_host_key.pub**: RSA public key được sử dụng bởi **sshd** deamon cho phiên bản 1 của giao thức SSH.
- **ssh_host_rsa_key**: RSA private key được sử dụng bởi **sshd** deamon cho phiên bản 2 của giao thức SSH.
- **ssh_host_rsa_key.pub**: RSA public key được sử dụng bởi **sshd** deamon cho phiên bản 2 của giao thức SSH.

Cấu hình OpenSSH chứng thực bằng tài khoản hệ thống

Các lựa chọn chính trong file **/etc/ssh/sshd_config**:

- Chấp nhận cho tài khoản người dùng **root** đăng nhập:

  PermitRootLogin yes
- Không chấp nhận (no) hoặc chấp nhận (yes) tài khoản có mật khẩu rỗng:

  PermitEmptyPasswords no
- Yêu cầu phải nhập mật khẩu khi đăng nhập:

  PasswordAuthentication yes
- Cấm người dùng truy cập:

  DenyUsers user1 user2 user3
- Cấm nhóm người dùng truy cập:

  DenyGroups group1 group2

- Cho phép người dùng truy cập:

  AllowUsers user1 user2

- Cho phép nhóm người dùng truy cập:

  AllowGroups group1 group2

Sau khi sửa tệp cấu hình, chạy lệnh sau để khởi động lại OpenSSH Server:

```
sudo systemctl restart sshd.service
```

Để kiểm tra cấu hình sshd đã hoạt động, từ client, tiến hành kết nối ssh đến server bằng lệnh

```
ssh usersvr@server
```

và nhập mật khẩu của người dùng usersvr.

Lưu ý, usersvr l à người dùng trên server.

Cấu hình OpenSSH chứng thực qua SSH key không hỏi mật khẩu

Khác với chứng thực bằng mật khẩu, ở đây ta sẽ cấu hình SSH Server cho phép chứng thực người dùng thông qua khóa.

Ta sẽ tạo ra cặp khóa Public key & Private key bằng thuật toán RSA hoặc DSA.

- Public key: Sử dụng cho Server
- Private key: Sử dụng cho Client

Thuật toán này hổ trợ cặp khóa tạo ra cho độ dài max là 2048 bit

Cách thực hiện, gồm 4 bước:

1. Kích hoạt chức năng chứng thực bằng **key** trên server, chỉnh sửa file **/etc/ssh/sshd_config**, thêm 2 dòng sau:

   PubkeyAuthentication yes
   AuthorizedKeysFile .ssh/authorized_keys
   Khởi động lại dịch vụ sshd:

   ```
   sudo systemctl restart sshd.service
   ```

2. Tạo cặp khóa:
   a. Trên client Linux/Unix:
      Chạy lệnh sau:

      [root@localhost ~]# **ssh-keygen** -t rsa
      (hoặc ssh-keygen -t dsa)
      dùng đường dẫn mặc định và enter khi được hỏi password

      Generating public/private rsa key pair.
      Enter file in which to save the key (**/home/user/.ssh/id_rsa**):
      Enter passphrase (empty for no passphrase):
      Enter same passphrase again:
      Your identification has been saved in /home/user/.ssh/id_rsa.
      Your public key has been saved in /home/user/.ssh/id_rsa.pub.
      The key fingerprint is:
      2b:92:ad:e1:5c:65:80:96:7b:d1:cb:4a:a4:4a:37:29 user@client.home.lan

Ở client, để bảo vệ file **id_rsa**, ta nên chuyển vị trí của nó, đồng thời phải thay đổi luôn đường dẫn của chúng (**IdentityFile ~/.ssh/id_rsa**) trong file **/etc/ssh/ssh_config**.

3. Đưa public pubkey của client lên server

   Để làm được việc chứng thực không cần password, ta cần cung cấp cho server public_key của client, và phải được ghi vào trong file **~ /.ssh/authorized_keys** (tương ứng trong **sshd_config**)

   a. Có 2 cách để thực hiện
      - Cách 1: dùng lệnh:

        # ssh-copy-id -i /home/user/.ssh/id_rsa.pub usersvr@server.home.lan
      - Cách 2: dùng lệnh:

        # ssh usersvr@server.home.lan cat < /home/user/.ssh/id_rsa.pub ">" /home/usersvr/.ssh/authorized_keys

4. Kiểm tra kết quả

Từ client, tiến hành kết nối ssh đến server bằng lệnh:

ssh usersvr@server.home.lan
kết quả:

[root@localhost ~]# ssh usersvr@server.home.lan
Last login: Sun Jun 21 00:31:50 2009 from client.home.lan
[root@localhost ~]#

2. Puppet và Zentyal

Tự học:

http://docs.puppetlabs.com/

http://doc.zentyal.org/

3. Thực hành

- Giả sử, quản trị mạng A được giao nhiệm vụ quản trị các máy server **R01**, **R02**. Tuy nhiên, A không thể vào phòng server để sử dụng bàn phím/chuột trực tiếp. A quyết định sử dụng SSH để quản trị các máy tính này từ xa. Để đảm bảo an toàn, tránh các phần mềm KeyLog có thể vô tình bị nhiễm, A quyết định sử dụng cơ chế xác thực thông qua bộ khóa công khai của người dùng theo cơ chế RSA.
  - Trên server **R01**, tạo tài khoản **user01** và cấu hình server này sao cho từ các máy tính khác có thể đăng nhập từ xa chỉ bằng tài khoản của **user01** thông qua dịch vụ SSH.
  - Hãy thiết lập server **R02** sao cho máy tính PC có khả năng đăng nhập từ xa như root thông qua SSH vào server trên mà không cần mật khẩu.