Bisoños Usuarios de GNU/Linux de Mallorca y Alrededores | Bergantells Usuaris de GNU/Linux de Mallorca i Afegitons

SSH con contraseña pública/privada (43203 lectures)

Per **Carles Pina i Estany**, ***cpina*** (http://pinux.info)

Creado el 30/03/2003 23:38 modificado el 30/03/2003 23:38

Muchos de nosotros nos conectamos frecuentemente a otras máquinas por `ssh`; pero cada vez que nos conectamos tenemos que escribir la contraseña. Al final es un lio, lento, contraseñas fáciles para recordarlas, etc.

Podemos usar SSH con protocolo 2 y usar un esquema de llave pública y privada para tener acceso desde nuestra cuenta a todas nuestras máquinas poniendo la contraseña una sola vez al inicio de sesión (o sin contraseña, pero sólo desde nuestro PC), hacer un icono en nuestro escritorio que nos dé una shell remota,...

El procedimiento está probado con Debian Woody, usando el paquete `ssh` versión 3.4p1-1 (OpenSSH[1]). En otras distribuciones puede variar un poco todo, pero básicamente será lo mismo.

A la máquina a la cual nos queremos conectar le llamaré *servidor*, y la máquina que usamos para conectarnos *cliente*.

En la máquina cliente tenemos que generar el par de llaves: pública y privada. Para hacerlo hacemos:

```
ssh-keygen -t rsa
```

y nos pedirá la llave privada. Esa llave puede ser diferente a cualquiera anterior, y tiene que ser difícil. Solo lo tendremos que escribir una vez "al día", a la sesión, etc.

Sólo quien tiene la llave privada puede encriptar el "flujo" para que sea desencriptado por nuestra clave pública residente en el servidor. Esa es una forma que el servidor sabe seguro que somos realmente nosotros quien está conectado a él.

Veremos algo así:

```
carles@pinux:~$ ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key (/home/carles/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/carles/.ssh/id_rsa.
Your public key has been saved in /home/carles/.ssh/id_rsa.pub.
The key fingerprint is:
fa:a3:f3:52:ad:2a:bc:fc:f1:a4:96:b4:5e:6c:5b:c4 carles@pinux
```

(Cuando nos pide el fichero podemos presionar Enter)
Hacemos un `chmod -R .ssh` para asegurarnos que `id_rsa` sólo lo podamos leer nosotros (Debian ya pone los permisos de `id_rsa` bien por defecto).

Hecho eso tenemos que enviar el `id_rsa.pub` a la máquina *servidor* y añadirle el contenido en el fichero `$HOME/.ssh/authorized_keys`. P. ej., en la máquina *servidor* hacemos:

```
cat id_rsa.cliente >> $HOME/.ssh/authorized_keys
```

(o sencillamente con un copiar-pegar en nuestro editor de texto)

Para tener las ideas claras, en `authorized_keys` tenemos las llaves de **quien** nos puede conectar a nosotros (si nosotros queremos conectar de *servidor* a *cliente* tenemos que hacer el `key-gen -t rsa` en el *servidor* y ponerlo en el `authorized_keys` del *cliente*.

En el fichero de configuración (`/etc/ssh/sshd_config`)del *servidor* tenemos que tener al menos:

```
Protocol 2
PubkeyAuthentication yes
```

Entonces un `/etc/init.d/ssh restart` (para que relea el fichero de configuración)y desde *cliente* podemos hacer `ssh servidor`, nos pedirá la "passphrase" introducida anteriormente y entraremos en *servidor*. Si tenemos *servidor1*, *servidor2*, *servidor3*... entraremos en cualquier servidor con la misma "passphrase". Si no hemos puesto nada de "passphrase" podremos entrar en cualquier servidor sin tecler nada (`ssh servidor`)

Ahora usaremos el `ssh-agent` y `ssh-add` para no tener que poner cada vez la "passphrase".

```
carles@pinux:~$ ssh-agent /bin/bash
```

Eso lo que hace es ejecutar un `/bin/bash` pero `ssh-agent` nos guardará las llaves privadas para los hijos de /bin/bash (todo lo que ejecutemos a partir de ese Bash)

Entonces hacemos:

```
carles@pinux:~$ ssh-add
Enter passphrase for /home/carles/.ssh/id_rsa:
Identity added: /home/carles/.ssh/id_rsa (/home/carles/.ssh/id_rsa)
carles@pinux:~$
```

A partir de ese momento podemos hacer `ssh servidor` y entraremos al servidor sin que nos pida ninguna contraseña. Evidentemente tambien funciona `scp fichero servidor:~` (para copiar *fichero* al *$HOME* del servidor).

Si queremos tener esa capacidad en todo el entorno X, podemos, o bien hacer el `ssh-agent /bin/bash` despues de arrancar (o de forma automática, etc.) o bien ponemos en nuestro *$HOME/.xinitrc* algo como:

```
carles@pinux:~$ cat .xinitrc
ssh-agent /usr/X11R6/bin/icewm
```

De esa forma, dentro de todo nuestro escritorio/gestor de ventanas tendremos activado el `ssh-agent` y con un sólo `ssh-add` añadimos nuestra llave privada.

(En caso que quisieramos que un proceso que **no** es hijo del `ssh-agent` tenga acceso a las llaves tendriamos que ponerle las variables de entorno `$SSH_AUTH_SOCK` y `SSH_AGENT_PID` hacia el mismo valor que a los hijos de `ssh-agent`.)

Tambien podemos poner un icono que ejecute:

```
xterm -T "ssh servidor" -bg black -fg lightgray -e ssh servidor
```

y de esa forma tenemos con un solo click una xterm en un servidor remoto sin más preocupaciones.

Si nos ausentamos del escritorio y queremos que `ssh-agent` olvide nuestra llave, hacemos:

```
carles@pinux:~$ ssh-add -D
All identities removed.
```

`ssh-agent` soporta varias identidades y `ssh-add` tiene más opciones (bloquear las identidades, dar un período de tiempo, etc.) pero lo podéis ver en el `man` o en la misma ayuda de los programas, ya que para un uso normal no hará falta.

Lista de enlaces de este artículo:

1. http://www.openssh.org

E-mail del autor: carles _ARROBA_ pinux.info
Podrás encontrar este artículo e información adicional en: http://bulma.net/body.phtml?nIdNoticia=1722