SSH

En los ejemplos “user1” será un usuario válido para la máquina “cliente” i “user2” será un usuario válido para la máquina “servidor”

Cliente	ssh [user2@]servidor [comando] Si se pone el comando, no se abre un terminal: se ejecuta el comando remotamente y ya está. Si no se pone el usuario, se entrará en el servidor con el mismo usuario, si existe. Otra manera de escribir lo mismo sería: *ssh -l user2 servidor* *Parámetros interesantes que tiene: -p nºpuerto (si el servidor no escucha en el 22 -x. Def- -v : mode verbós per veure els passos de la comunicació (-q seria lo contrario). Amb -vv és més verbós -i nomdelarxiudeclaupúblicaquesutilitzarà: (en comptes del id_dsa.pub, per exemple) -l user2 -o opcio_de_fitxer_de_configuració=valor *Arxiu de configuració del client: /etc/ssh/ssh_config , o bé /home/user1/.ssh/config La preferència és: línia dórdres, fitxer config personal, fitxer config global. L'ordre de les línies dins el fitxer és important: a la primera coincidència s'atura. -Port nºport : diu a quin port es conectarà per defecte, si el servidor no escoltés al 22. -PasswordAuthentication yes/no : ofereix la possibilitat d'utilitzar aquest mètode d'autenticació -PubkeyAuthentication yes/no : ofereix la possibilitat d'utilitzar aquest mètode d'autenticació -PreferredAuthentications publickey,password -CheckHostIP yes/no : xequejarà l'autenticitat del servidor consultant a l'arxiu known_hosts -IdentityFile... : especifica la ruta de la clau privada de l'usuari (~/.ssh/id_rsa ó id_dsa) -User usuarioautilizarenlaconexionpordefecto (para no tenerlo que decir en la línea de comandos) Per més opcions veure man ssh_config
Servidor	*Arxiu de configuració del servidor: /etc/ssh/sshd_config -Port 2341 -PermitRootLogin yes/no/without-password -PasswordAuthentication yes/no -PubKeyAuthentication yes|no -PermitEmptyPasswords yes/no -ListenAddress 0.0.0.0 : això vol dir a totes les interfícies -AllowUsers usuario??@192.168.0.2?? (els usuaris que no hi apareixen, tindran denegat l'accés) -DenyUsers fulanito[@ipmaq] menganito none (“ “ “ “ “ , tindran permís per entrar) -També està AllowGroups i DenyGroups. A més de * i ?, es pot utilitzar ! per negacions -ForceCommand /ruta/comanda/que/seexutara/al/servidor/un/cop/loguejat (se sobreescriu la possible comanda que es pugui haver escrit al client) -MaxSessions nº : obvi -MaxAuthTries 1 :número de reintents que es permeten per intentar escriure bé la contrasenya -MaxStartups nº: número de conexions simultànies permès que no han acabat d'autenticar-se -TCPKeepAlive yes : s´assegura que les sessions es tanquin correctament
Montaje de unidades remotas	En el cliente, para montar: *sshfs user2@servidor:/ruta/carpeta/remota /ruta/carpeta/local -p nºport -o -ro -o uid=nºuidlocal, gid=nºgidlocal,allow_other* donde las opciones uid y gid dicen que el propietario de los ficheros accesibles en /ruta/carpeta/local pase a ser el especificado por el uid/gid especificado (que será normalmente el del usuario local) en vez de ser el user2. Esto es para evitar inconsistencias en los permisos (si no, estaríamos viendo en un punto de montaje local unos ficheros propiedad de otro usuario no existente en el sistema) En el cliente, para desmontar: *fusermount -u /ruta/carpeta/local* Para hacer el punto de montaje permanente, se puede añadir la siguiente línea al fichero /etc/fstab: *sshfs#user2@servidor:/ruta/remota /ruta/local fuse auto,user,uid=º,gid=nº 0 0* (en las opciones no se pone “defaults”, porque equivale a: rw, auto, **nouser**, ... con lo que no dejaríamos que un usuario normal montara la carpeta. Esto no sería necesario si fstab funcionara bien, pero resulta que si no tenemos configuradas la autenticación por clave pública, necesitaremos introducir una contraseña en el arranque del sistema que no nos pregunta, así que el montaje no se hace, y deberá culminarlo el usuario local ejecutando simplemente “mount /ruta/local”.
Copia de archivos remotos	scp userX@maq:archivo/s userX@maq:directorio Si el archivo/directorio son locales no hace falta indicar usuario@maq. El directorio remoto por omisión es el /home del usuario remoto.

	*Parámetros que tiene: -r : copia recursiva (como primer parámetro se puede poner una carpeta, y se copiará todo su contenido) -p : mantiene los propietarios, permisos y las fechas de {c|m|a} -P nºport -l nºkb/smáximosanchobandapermitidos

Claus d´usuari

Existeix una altra forma de loguejar-se en el servidor Ssh que implica no haver d´escriure user/password cada cop. Es tracta d´utilitzar un parell de claus pública/privada d´usuari. La idea és que cada usuari generi el seu parell propi, i col.loqui la seva clau pública al servidor (la privada romandrà a la màquina client, protegida per contrasenya).
Sent user1 al client, volem connectar-nos com user2 a la màquina servidora. Per fer això, a la màquina client s´ha de crear un parell de claus per user1, així:

ssh-keygen -b 1024 -t rsa [-f nomarxiuclaus]

on -b és el nº de bits de la clau (pot ser 512, 1024 o 2048) i -t pot ser també dsa. Preguntarà llavors pel passphrase, que és una contrassenya LOCAL integrada íntimament amb les claus -concretament, amb la clau privada-, per poder fer-les servir (per protegir les claus per si algú les agafa per poder-les utilitzar pel seu compte). Si es posa, ens la preguntarà després cada cop que ens volguem conectar. ¿Per a canviar un passphrase un cop estigui definit, amb aquesta comanda: *ssh-keygen -p -q -t rsa -f ruta/id_rsa.pub -P passantic -N passnou* ? (el paràmetre -q evita els missatges per la sortida estàndar; el paràmetre -p es per canviar la passphrase sense crear un nou fitxer de clau privada)

Es creen dos arxius en /home/user1/.ssh que són id_rsa i id_rsa.pub. A la màquina servidora copiem dins de l´arxiu /home/user2/.ssh/authorized_keys el contingut de l´arxiu id_rsa.pub. Per fer això, es pot fer de vàries maneres:

cat ~/.ssh/id_dsa.pub | ssh user2@servidor "cat - >> ~/.ssh/authorized_keys"
scp ~/.ssh/id_dsa.pub user2@servidor:.ssh/authorized_keys (si només farem servir aquest client !?)
ssh-copy-id -i ~/.ssh/id_dsa.pub user2@servidor

Nota: els permisos de ~/.ssh del servidor han de ser 700 i els de l´arxiu authorized_keys han de ser 600.

A més a més, al servidor s´ha de configurar una sèrie de coses per a què això funcioni. En concret, al seu arxiu de configuració s´han d´escriure els següents valors:

PermitRootLogin without-password (o "no", directament)
PasswordAuthentication no (no s'utilitzarà ja més el sistema d'autentificació per contrasenyes)
PubkeyAuthentication yes (activem el sistema d'autentificació per claus)
AuthorizedKeysFile %h/.ssh/authorized_keys (per indicar l'arxiu on s'enmagatzemaran les claus públiques dels clients)

Anell de claus

Per poder fer servir passphrases i poder fer que el procés d'autenticació continuï sent totalment automàtic, podem fer servir el dimoni ssh-agent. Ssh-agent normalment és un programa que a les distribucions actuals es posa en marxa automàticament amb les X (en /etc/X11/xinitrc ó /etc/X11/Xsession.options..., depén), fent així que tots els programes X tinguin una conexió amb ssh-agent en segon pla. Però el podríem executar nosaltres directament, i fins i tot li podríem associar un programa on hi estigués vinculat (i tot els programes fills d'aquest). Per exemple: *ssh-agent gnome-terminal &* Que Ssh-agent estigui funcionant vinculat a algun entorn (X, gnome-terminal, etc) implica que dins d'aquest entorn es podrà usar el programa ssh-add per afegir la passphrase una vegada a l´agent i l´agent passarà al seu torn aquesta informació d´autenticació automàticament cada cop que tu necessitis aquest passphrase. Així que el pròxim cop que s´executi el client ssh no preguntarà ni tan sols el passphrase. Suposant que ssh-agent funciona (ps auxw | grep "ssh-agent") , la manera d´afegir la clau a l´anell és executant ssh-add , a seques. Si el programa troba la clau DSA, preguntarà la passphrase, i ja està.

Ssh-add té paràmetres també:
nomclaupúblicaconcretaaafegir
-t nºsegons : temps que romandrà la clau dins de l'anell (per defecte, per sempre)
-l : llista les claus que estan dins l´anell
-L : para ver si ssh-agent está funcionando o no
-d nomclau : esborra la clau de l´anell
-D : esborra totes les claus de l´anell

Incluso si cerramos la shell, el agente no finalizará por sí mismo. Para finalizar el programa, necesitamos llamar a ssh-agent -k en la shell en la que arrancamos ssh-agent. Si no podemos hacerlo, porque la shell llamada no está ya ejecutándose, por ejemplo, la única opción que nos queda es el comando kill.

Para ejecutar ssh-agent con una aplicación antes que como un demonio, ejecutamos el nombre del ejecutable en la línea de comandos,ssh-agent gnome-terminal & el cual restringe el agente a esta aplicación y a los programas arrancados en él. Si abandonamos la terminal, shh-agent finaliza automáticamente y elimina cualquier información de clave que utilizó. Cargar las claves en el agente sin restricciones de tiempo y luego olvidarnos de cerrar la pantalla puede significar abrir una vulnerabilidad en la seguridad.