SSH - A BIZTONSÁGOS ALTERNATÍVAZ RSH HELYETT

***Kadlecsik József**, kadlec@sunserv.kfki.hu*
KFKI RMKI Számítógép Hálózati Központ

Abstract

More security on Internet usually means greater inconvenience - but ssh is an exception. The features and usage of this excellent tool are detailed with examples from real systems.

1. Miért használjunk ssh-t az rsh helyett?

A (világ) hálózaton dolgozva nap-mint-nap kell bejelentkeznünk egyik számítógéprõl egy másikra. Ehhez leggyakrabban a telnet és rsh/rlogin parancsok valamelyikét használjuk, noha azok biztonsági szempontból még csak nem is kielégítõek. Például közbülsõ gépek lehallgathatják az adatforgalmat, beleértve a password-öket is - a telnet, rsh nem titkosít - de sok más módszer is létezik jogosulatlan hozzáférés megszerzéséhez.

A finn Tatu Ylönnen által írt ssh programcsomag (ssh, slogin, scp) funkcionálisan az rsh (rsh, rlogin, rcp) helyettesítõje úgy, hogy biztonságos, erõs autentikációval ellenõrzött titkosított kapcsolatot hoz létre két, "egymásban" nem bízó gép között, amelyeket a nem-biztonságosnak tekintett hálózat köt össze. Az ssh a következõ típusú támadások ellen védett:

- IP cím hamisítás, amikor egy távoli gép olyan IP csomagokat küld, mintha azok egy másik gép csomagjai lennének.
- IP source routing, amikor egy gép úgy tesz, mintha egy távoli gép IP csomagja rajta keresztül érkezne.
- DNS hamisítás, amikor name server bejegyzéseket hamisítanak
- az adatforgalom közbülsõ gépek általi lehallgatása ellen
- az IP csomagok közbülsõ gépek általi megváltoztatása ellen
- X autentikációs adat lehallgatása és meghamisítása ellen

Az ssh rhosts-RSA autentikációs módszerét használva semmiben sem különbözik a felhasználók számára az rsh-tól - akár rsh/rlogin/rcp-ként is installálható. Használható ssh-t nem ismerõ gépekhez kapcsolódva is, mivel az ssh, amennyiben nem találja az ssh szervert a másik gépen, elindítja az rsh-t önmaga helyett.

Az ssh természetesen semmifajta védelmet nem tud nyújtani akkor, ha valaki hozzáfér az adatainkhoz a számítógépünkön, vagy ha valaki már betört a gépre és azon 'root' jogokkal rendelkezik - a hálózaton keresztüli támadások kivédésére készült.

Az ssh szerver és kliens Unix rendszereken mûködik és ingyenesen használható; kliens létezik OS/2 és MS-Windows-ra is, az utóbbira a hivatalos változat azonban kereskedelmi termék.

2. Hogyan mûködik?

Az ssh RSA kulcsokon alapul. Minden ssh-t használó gépnek van egy host-azonosító RSA kulcsa (default 1024 bit). A szerver gépen az sshd daemon ezen kívül generál egy szerver RSA kulcsot is (default 768 bit), amelyet óránként frissít és amit soha nem tárol a merevlemezen .

Amikor egy kliens (ssh) hozzákapcsolódik a szerverhez (sshd), elõször a szerver azonosítása történik meg. A szerver elküldi a host- és szerver-kulcsok publikus részét a kliensnek. A kliens összehasonlítja a host-azonosító publikus kulcsot az adatbázisában lévõvel és ellenõrzi, hogy az változatlan-e. Ezután a kliens generál egy 256 bites véletlenszámot, amit a szerver host- és szerver-kulcsával egyaránt titkosít, majd ezt visszaküldi a szervernek. A szerver az RSA kulcsai ismeretében vissza tudja fejteni a titkosított véletlenszámot, amit a továbbiakban a két oldal a forgalom titkosító kulcsául (session key) fog használni - ettõl a ponttól kezdve minden titkosított, IDEA, DES, 3DES, ARCFOUR (RC4) vagy TSS algoritmust használva.

A következõ lépés a kliens-azonosítása (autentikáció). Több módszer áll ehhez a rendelkezésünkre:

- rhosts autentikáció: Ha a kliens gép szerepel az /etc/hosts.equiv file-ban és a felhasználói azonosítók megegyeznek a két gépen, vagy a kliens gép és a felhasználó azonosítója szerepel a ~/.rhosts vagy ~/.shosts file-ok valamelyikében, akkor a bejelentkezés engedélyezett. (Ez az autentikációs módszer nem használatos, mert az ssh ekkor az rsh-nál nem nyújtana nagyobb biztonságot.)
- rhosts és host-RSA autentikáció (az elsõdlegesen használt autentikációs mechanizmus): Amennyiben a bejelentkezés engedélyezett lenne rhosts autentikációval, a szerver még ellenõrzi a kliens host-azonosító RSA kulcsát, úgy, ahogy a kliens ellenõrizte a szerverét. Ez a mechanizmus kivédi az IP hamisítás, DNS hamisítás és source routing támadási módszereket.
- user-RSA autentikáció: az ssh támogatja azt, hogy a felhasználók definiáljanak saját RSA kulcsokat, amelyeket azután távoli gépekre ssh-val való bejelentkezéskor önmaguk azonosítására használhatnak.
- password alapú autentikáció: ha az elõzõ módszerek egyikével sem sikerült azonosítani a felhasználót, az ssh kéri a felhasználó password-jét. A password titkosítottan kerül át a szerverre, így lehallgatni nem lehet.

Ha a kliens sikeresen azonosította magát, akkor a kapcsolat elõkészítéseként különbözõ szolgáltatásokat kérhet a szervertõl: pszeudo-terminál allokálását, az X11-es kapcsolatok átirányítását a biztonságos csatornára, tetszõleges TCP/IP kapcsolatok átirányítását a biztonságos csatornára vagy az u.n. authentication agent kapcsolat átirányítását a biztonságos csatornára.

Végül a kliens vagy egy shell indítását (slogin, ssh) vagy egy parancs végrehajtását (ssh, scp) kérheti a szervertõl.

A fázisokat jól illusztrálja az ssh verbose módban való indítása:

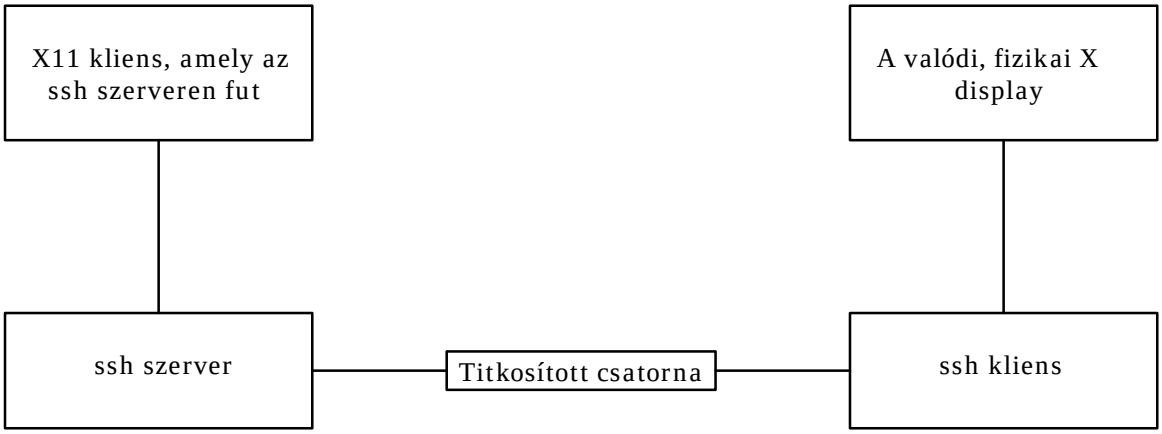
```
kadlec@blackhole:~$ ssh -v sunserv
SSH Version 1.2.13 (i486-unknown-linux), protocol version 1.3.
Standard version. Does not use RSAREF.
Reading configuration data /home/kadlec/.ssh/config
Applying options for *
Reading configuration data /usr/local/etc/ssh_config
ssh_connect: getuid 648 geteuid 0 anon 0
Connecting to sunserv [148.6.0.5] port 22.
Allocated local port 1022.
Connection established.
Remote protocol version 1.3, remote software version 1.2.13
Waiting for server public key.
Received server public key (768 bits) and host key (1024 bits).
```

```
Host `sunserv` is known and matches the host key.
Initializing random; seed file /home/kadlec/.ssh/random_seed
Encryption type: idea
Sent encrypted session key.
Received encrypted confirmation.
Trying rhosts or /etc/hosts.equiv with RSA host authentication.
Remote: Accepted by .shosts.
Received RSA challenge for host key from server.
Sending response to host key RSA challenge.
Remote: Rhosts with RSA host authentication accepted.
Rhosts or /etc/hosts.equiv with RSA host authentication
        accepted by server.
Requesting pty.
Requesting X11 forwarding with authentication spoofing.
Requesting authentication agent forwarding.
Requesting shell.
Entering interactive session.
Last login: ....
No news.
You have mail.
kadlec@sunserv:~$ exit
Connection to sunserv closed.
Transferred: stdin 14, stdout 1109, stderr 31 bytes
        in 40.4 seconds
Bytes per seconds: stdin 0.3, stdout 27.5, stderr 0.8
Exit status 0
```

3. Mit tud még az ssh?

3.1 X11 átirányítás

Az egyik legkényelmesebb tulajdonsága az ssh-nak, hogy X Windows grafikus felület használata esetén (a lokális gépen a DISPLAY változó be van állítva) a távoli X11-es kapcsolatokat automatikusan átirányítja a biztonságos titkosított csatornára. Továbbá, hogy az X11 szerver autentikációs adatai ne hagyják el a szervert, az ssh véletlenszerû "cookie"-t generál, és azt küldi át a hálózaton, természetesen titkosítottan:



Nem kell bajlódnunk sem a DISPLAY változó beállításával, sem xhost vagy MAGIC-COOKIE X Windows autentikációkkal, mert az ssh mindent elvégez helyettünk. Egyszerûen végrehajthatjuk a következõ parancsot,

kadlec@blackhole:~$ *ssh sunserv xterm*

és az X terminál a képernyõnkön minden további nélkül megjelenik.

3.2 TCP/IP port átirányítás

Az X11-es kapcsolatok automatikus átirányítása a TCP/IP portok átirányításának egy speciális esete. Például az ftp parancsot használhatjuk úgy, hogy a password-ünk ne titkosítatlanul haladjon át a hálózaton a gépünk és a távoli gép között, hanem az ssh által titkosítottan. Ehhez grafikus felületen csak a következõt kell tennünk:

1. Egy ablakban végrehajtjuk a következõ parancsot (a lokális port száma tetszõleges):

   kadlec@blackhole:~$ *ssh -L 1234:sunserv:21 sunserv*

Ezzel a paranccsal bejelentkeztünk a sunserv gépre és ugyanakkor a sunserv 21-es portját, amely az ftp parancs-portja, átirányítottuk a lokális gép 1234-es portjára a titkosított csatorna fölött.

2. Egy másik ablakban pedig a következõt:

   kadlec@blackhole:~$ *ftp localhost 1234*
   Connected to localhost.
   220 sunserv FTP server (Version wu-2.4(6) Fri Jun 16 11:25:25

MET DST 1995) ready.
Name (localhost:kadlec):

Kész! Az ftp parancs-portja így titkosított, password-ünk nem-lehallgathatóan fog áthaladni a hálózaton.

Mail-szervert használva levelezõ programjaink minden egyes kapcsolatnál titkosítatlanul küldik át password-ünket a szervernek. Az ssh port-átirányítását használva a titkosítás ezeknél is megoldható. Stephane Bortzmeyer Perl-scripje POP, míg Holger Trapp program-csomagja pine és IMAP esetén oldja meg a problémát ssh segítségével; sem a mail-szerver, sem a kliens nem igényel semmilyen módosítást egyik esetben sem:

ftp://ftp.pasteur.fr/pub/Network/gwpop/
ftp://ftp.kfki.hu/pub/packages/security/
ssh/contrib/imap_sec.tgz

3.3 Tömörítés

Lassú vonalak fölött igen hasznos lehet az ssh azon képessége, hogy képes tömöríteni a teljes forgalmat, beleértve az átirányított X11 és TCP/IP portok forgalmát is.

3.4 Felhasználói RSA kulcsok

Az ssh esetében felhasználói RSA kulcsok az ssh-keygen paranccsal hozhatók létre. Egy RSA kulcshoz "password" helyett tetszõleges hosszúságú "passphrase" rendelhetõ. Az rhost és RSA autentikációhoz képest az RSA autentikáció a felhasználótól több (de egyszeri) konfigurációs munkát igényel:

- rhost és RSA autentikáció
  a) A távoli gépen az ~/.rhosts vagy ~/.shosts file-hoz megfelelõ

     host *user*

     ejegyzéseket kell adni

- RSA autentikáció
  a) A lokális gépen:
     i) Az ssh-keygen paranccsal RSA kulcso(ka)t kell létrehozni
     ii) Módosítani kell a bejelentkezési folyamatot úgy, hogy az az ssh-agent programmal kezdõdjön és minden más program (shell) abból induljon:
     Például X session (xdm) esetén:

     ~/.xsession file:

     ```
     #!/bin/sh
     exec ssh-agent $HOME/.xsession.real
     ```

     iii) Vagy a prompt-nál, vagy az ~/.xsession.real-ból, vagy menübõl stb. végrehatjuk az ssh-add parancsot, amellyel a lokális gépen futó ssh-agent-hez hozzáadjuk az általunk használt RSA kulcsot a megfelelõ passphrase megadásával.
  b) A távoli gép(ek)en az ~/.ssh/authorized_keys file-(ok)hoz hozzáadjuk az RSA kulcsunk publikus részét, amely a lokális gépen az ~/.ssh/identity.pub file-ban van.

Az elképzelés az RSA autentikáció mögött a következõ: az ssh-agent a felhasználó lokális gépén fut, és kezdetben semmilyen RSA kulcsot nem ismer - azokat az ssh-add programmal kell hozzáadni. (Tetszõleges számú kulcs használható.) Mivel az RSA kulcsokhoz a kulcsmondatokat a lokális gépen kell megadni, azok

soha nem haladnak át a hálózaton. A távoli gépek az RSA kulcsunk publikus részével azonosítják magukat és a kapcsolatok az ssh-agent-hez automatikusan a titkosított csatornára irányítódnak.

Az RSA autentikáció különösen akkor hasznos, amikor az ssh-t olyan gépen szeretnénk használni, amelyen nincs ssh és a gép adminisztrátora valamiért nem hajlandó azt installálni. Az ssh ekkor közönséges felhasználóként lefordítható, installálható és RSA valamint password autentikációval használható. (Ahhoz, hogy az rhost-RSA autentikáció mûködjön, az ssh-t 'root'-ként kell telepíteni.)

4. Honnan tölthetõ le Magyarországon?

ftp://ftp.kfki.hu/pub/packages/security/ssh

5. Hol található a témáról további információ az Interneten?

Ssh Home Page	http://www.cs.hut.fi/ssh/
Ssh FAQ	http://www.uni-karlshruhe.de/~ig25/ssh-faq/
RSA FAQ	http://www.rsa.com/rsalabs/fa q/