< Linux

Debian GNU/Linux távoli elérése

- **Szerző:** Sallai András

- Web: http://szit.hu

Kapcsolódás SSH szerverhez

Parancssor

Az SSH szervernek futnia kell a szerveren. A telepítendő csomag az openssh-server:

```
# apt install openssh-server
```

Kliens oldalról a következő programot szükséges telepíteni:

```
# apt install openssh-client
```

Ez alapból telepítve van egy alaprendszeren.

Használata a következő:

```
ssh usernev@domainnev.and
```

Grafikus felületre SSH kliens a **Putty**. Telepítés Linuxon:

```
# apt install putty
```

Root tiltása

Szerkesszük sshd konfigurációs állományát:

```
mcedit /etc/ssh/sshd_config
```

Keressük meg a PermitRootLogin beállítást. Alapértelmezetten „yes” értéke van. Állítsuk „no” értékre:

```
PermitRootLogin no
```

Indítsuk újra az ssh szervert:

```
invoke-rc.d ssh reload
```

Fájlok mozgatása

Egy távoli Linuxra az **scp** paranccsal tudunk fel/le másolni állományokat.

Windowsos rendszerre a **WinSCP** programot használhatjuk.

Titkos kulcsok használata

Ha távoli gépre való belépéshez az alapértelmezett kulcs helyett más kulcsot szeretnénk használni:

```
ssh -i /home/joska/.ssh/joska_rsa user@szerver
```

Ilyen lehet, amikor kulcs alapú beléptetéshez külön kulcspárt készítettünk, és szeretnénk megadni a kulcspárt titkos változatát az ssh parancsunk számára.

SSH alagút

Az alagút kialakítás a saját gépünkön Távoli szerveren (smbszerverip) van egy Samba megostás. Ezt szeretnénk elérni az Interneten keresztül, kivülről ez nem érhető el, csak egy felhasználó szintű ssh hozzáférésünk van jozsi néven:

```
ssh -l jozsi -L 19999:smbszerverip:139 tavoligep.hu
```

Ezek után itthon elérhetjük a 19999 porton a megosztást:

```
mount -t cifs //localhost/megosztas /mnt/megosztas -o
port=19999,username=jozsi
```

A mount.cifs fájl a cifs-utils csomagban található.

Vagy így:

```
smbclient //localhost/megosztas -p 19999 -U jozsi
```

Az alagút kialakítása távolról a saját gépünkre

```
ssh -l jozsi -R 9999:smbszerverip:139 tavoligep.hu
```

Esetleg kulcs megadásával:

```
ssh -l janos -i id_rsa -f -N -g -L 9999:server:139 sulixerver.egressy.eu
```

- A -f a háttérbe teszi a folyamatot.
- A -N azt mondja nem akarunk távoli parancsot futtatni (csak porttovábbítást).
- A -g megengedi távoli géphez és helyi port összekötését.

A scriptből kezeljük az ssh kapcsolatot akkor jól jöhet a unix socket használata. Ezt a -S kapcsolóval

tehetjük meg:

```
ssh -N -f -M -S /etc/sshKapcsolat -L 9999:localhost:139 tavoliszerver.and
```

Ekkor a kapcsolat bezárható így:

```
ssh -S /etc/sshKapcsolat -O exit tavoliszerver.and
```

A kapcsolat ellenőrzése:

```
ss -lt
```

RDP szerver elérése:

```
ssh -l janos -L 33389:windowipcim:3389  sshszerveripcime
```

```
rdesktop localhost:33389
```

```
           SSH alagút               SSH alagút
        ==============            ============
LinuxPC-----------------SSHServer---------------távoliWindows
        ==============            ============
```

Ha hitelesítés, tartomány és geometria is szükséges:

```
rdesktop -d zold.and -u janos -p titok -g 1024x768 localhost:33389
```

A példában a:

- tartománynév: zold.and
- felhasználónév a tartományban: janos
- janos jelszava: titok
- felbontás: 1024×768-as felbontást szeretnénk
- a Windowsos gép IP címe

Bőbeszédű üzemmód

Ha nem tudunk kapcsolódni és nem derül ki mi a probléma, használhatjuk a -v kapcsolót az akadály felderítésére:

```
ssh -vT user@szerver
```

További ssh szerverek

LSH

```
apt-get install lsh-server
```

RSH

Függőségként openbsd-inetd csomagot függőségként felteszi, így inetd-ből fut.

```
apt-get install rsh-server
```

SSHFS

Az SSHFS segítségével a távoli hozzáférésünket felcsatolhatjuk, egy könyvtárba. Így úgy használhatjuk mintha a helyi gépen lenne a távoli elérés.

Telepítés:

```
# apt install sshfs
```

Csatolás:

```
$ sshfs usernev@hostnev: /home/usernev/csatolasipont
```

Leválasztás:

```
fusermount -u /home/usernev/csatolasipont
```

Grafikus kliens az SSHFS számára:

- https://github.com/spantaleev/sftpman-gtk (Linux; 2018)
- https://github.com/billziss-gh/sshfs-win (Windows; 2018)

SSH known_hosts állomány

Néha előfordulhat, hogy ~/.ssh/known_hosts állományban olyan kulcs tartozik az IP címhez ami már elavult.

Ehhez hasonló figyelmeztetést látunk:

```
ssh joska@192.168.5.100
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@    WARNING: REMOTE HOST IDENTIFICATION HAS CHANGED!     @
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
IT IS POSSIBLE THAT SOMEONE IS DOING SOMETHING NASTY!
Someone could be eavesdropping on you right now (man-in-the-middle attack)!
It is also possible that a host key has just been changed.
The fingerprint for the ECDSA key sent by the remote host is
9a:5b:1d:8f:9f:d0:04:96:62:ef:68:58:59:e6:07:9c.
Please contact your system administrator.
Add correct host key in /home/joska/.ssh/known_hosts to get rid of this
message.
Offending ECDSA key in /home/joska/.ssh/known_hosts:1
```

```
  remove with: ssh-keygen -f "/home/joska/.ssh/known_hosts" -R 192.168.5.100
ECDSA host key for 192.168.5.100 has changed and you have requested strict
checking.
Host key verification failed.
```

Fontoljuk meg, hogy valóban megváltozott a gép, vagy illetéktelenek változtatták meg. Ha változás normális, szeretnénk törölni az ismert gépek közül.

A figyelmeztetőüzenetben benne van a megoldás is.

Törölhetjük az adott IP címet a következő paranccsal.

```
ssh-keygen -f "/home/andras/.ssh/known_hosts" -R 192.168.5.100
```

A példában a 192.168.5.100 IP címet töröljük.

Bejelentkezés előtti üzenet

Szerkesszük a következő állományt:

```
# nano /etc/ssh/sshd_config
```

Keressük meg a következőt:

```
#Banner /etc/issue.net
```

Vegyük ki a sor elején a megjegyzés, „#” karakterét.

```
Banner /etc/issue.net
```

Mentsünk, majd indítsuk újra az SSH szervert:

```
# service ssh restart
```

Jelszavak

A jelszavakat ha scriptből automatizáltan szeretnénk megadni:

```
apt install sshpass
```

```
sshpass -p "titok" scp -r /hely/utvonal janos@zold.and:/utonval
```

Ne szerepeljen a jelszó a bash history-ban:

```
sshpass -f pass scp -r /hely/utvonal janos@zold.and:/utonval
```

Az sshpass használható az ssh paranccsal is.

Parancsok végrehajtása a távoli gépen

Az SSH továbbra is szükséges, de most nem szeretnénk belépni a távoli gépre, csak parancsokat végrehajtani ott.

Egy script futtatása a távol gépen, amikor a script a távoli gépen van:

```
ssh janos@pluto ./ad.sh
```

vagy:

```
ssh janos@pluto /home/janos/ad.sh
```

Parancs futtatási távoli gépen, de az eredményt helyben látjuk:

```
ssh janos@pluto "ls -la"
```

Parancs leírása, majd futtatása:

```
$ ssh janos@pluto <<'END'
echo 1 > adat.txt
END
```

Helyi parancs futtatása távoli gépen:

```
$ ssh janos@pluto 'bash -s' < csinald.sh
```

A Windowson dolgozunk, akkor a Putty webhelyéről letölthető plink.exe programra lesz szükségünk, amit így használunk:

```
C:\>plink janos@pluto -m csinald.sh
```

Automatikus bejelentkezés

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

vagy

```
ssh-keygen -t dsa
```

```
janos@jupiter:~$ ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key (/home/janos/.ssh/id_rsa): ./id_rsa
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in ./id_rsa.
Your public key has been saved in ./id_rsa.pub.
```

```
The key fingerprint is:
SHA256:Vsu1R9afuXuUIWf7KoPuoCgE0Z0gKKLIiPK0Ttv/TA0 janos@jupiter
The key's randomart image is:
+---[RSA 2048]----+
|o..o .           |
|=.. o          . |
|O.         . . o .|
|B..       o o = ++|
|.+ .     E o . =o=|
|  =     . o   . oo|
| + o     o ..   o.|
|  + .. + .. o   +|
|   ...o.ooo  o.o.|
+----[SHA256]-----+
janos@jupiter:~$
```

Megadhatjuk a kulcs méretét:

```
ssh-keygen -t rsa -b 4096
```

Létrejön két állomány (publikus és nyilvános kulcs), a felhasználó .ssh könyvtárában. Jelszót ne adjunk meg.

```
$HOME/.ssh/id_dsa
$HOME/.ssh/id_dsa.pub
```

Azon a gépen ahol be szeretnénk jelentkezni automatikusan:

```
su -
```

```
vi /etc/ssh/sshd_config
```

A konfigurációs fájlban a következő sorra van szükségünk:

```
AuthorizedKeysFile %h .ssh/authorized_keys
```

A fenti sor elől kiveszem a „#” karaktert, hogy ne legyen a sor megjegyzésbe:

A publikus kulcsot felmásoljuk arra a számítógépre ahova be akarunk jelentkezni:

```
scp $HOME/.ssh/id_dsa.pub   user@hova:.ssh/authorized_keys
```

Fontos, hogy a felmásolt név authorized_keys legyen az .shh könyvtárban.

Kulcsok konvertálása

Putty kulcspár konvertálása OpenSSH kulcspárrá

Windowson

- Indítsuk el a PuttyGen programot
- Load gombbal töltsük be a privát kulcsot.
- Konvertáljuk:
  - Conversions → Export OpenSSH
  - A privát kulcsot kell (és csak azt tudjuk exportálni
- Másoljuk a ~/ssh/id_rsa állományba
- Készítsük el az RFC 4716 verziót az ssh-keygen paranccsal:
  - ssh-keygen -e -f ~/.ssh/id_rsa > ~/.ssh/id_rsa_com.pub
- Készítsük el az RFC 4716-ból az OpenSSH formátumot:
  - ssh-keygen -i -f ~/.ssh/id_rsa_com.pub > ~/.ssh/id_rsa.pub

Linuxon

Szükséges csomag telepítése:

```
apt install putty-tools
```

Putty kulcspár létrehozása:

```
puttygen -t rsa -b 2048 -C "janos@zold.and" -o id_rsa.ppk
```

Privát kulcs készítse:

```
puttygen id_rsa.ppk -O private-openssh -o id_dra
```

Publikus kulcs készítése:

```
puttygen id_rsa.ppk -O public-openssh -o id_dra.pub
```

OpenSSH kulcsból Putty kulcs

OpenSSH kulcsból Putty kulcs Linuxon.

Szükséges eszköz telepítése:

```
apt install putty-tools
```

Putty kulcs készítése:

```
puttygen id_rsa -o id_rsa.ppk
```

SFTP és SSHFS

A példában webmestereknek szeretnénk egy könyvtárat elérhetővé tenni, hogy oda feltölthessék állományaikat, mindezt valamilyen biztonságos protokollon keresztül. Nem szeretnénk, hogy a webmester betudjon lépni linuxos felhasználóként programindítás céljából.

Szerveroldalon

Egy csoport a webmseterk számára:

```
addgroup webmaster
```

Ez lesz a gyökérkönyvtára:

```
mkdir -p /home/www/zoldand
```

A zoldand felhasználó fog feltölteni:

```
useradd -d / -s /bin/false zoldand
usermod -a -G webmaster zoldand
```

A gyökérkönyvtárat nem írhatja a felhasználó, csak a chroot kedvéért csináltuk. Egy újabb könyvtárat hozunk létre számára:

```
mkdir -p /home/www/zoldand/zoldand
chgrp webmaster /home/web/zoldand/zoldand
chmod 775 /home/www/zoldand/zoldand
```

Az apache webszerver egy htdocs könyvtárat fog kiszolgálni:

```
mkdir /home/www/zoldand/zoldand/htdocs
```

/etc/ssh/sshd_config

```
[...]
Subsystem sftp /usr/lib/openssh/sftp-server
[...]
Match Group webmester
    ChrootDirectory /home/www/zoldand
    AllowTCPForwarding no
    X11Forwarding no
    ForceCommand internal-sftp
```

```
systemctl restart sshd
```

Kliens oldalon

Elérés SFTP protokollal:

```
sftp zoldand@server
```

Elérés SSHFS protokollal:

```
apt install sshfs
mkdir ~/c
sshfs zoldand@server: ~/c
```

Linkek

Forrás

- https://www.server-world.info/en/note?os=Debian_9&p=ssh&f=5 (2018)

Programok

- https://mhogomchungu.github.io/sirikali/ (Win; 2018)
- https://github.com/billziss-gh/sshfs-win (Win; 2018)

Kapcsolódás GDM-hez

Windowson szükséges program

```
Xming
```

Debian GNU/Linux alatt be kell állítani

```
/etc/gdm3/daemon.conf
```

```
[xdmcp]
Enable=true
```

Kapcsolódás

Az Xlaunch programot kell indítanunk.

Az előugró varázslóban a következőket válasszuk:

- One Window (Tovább)

- Open session via XDMCP (Tovább)
- Connect to host: 192.168.16.21 (A Linuxos gép IP címe; Tovább)
- (Tovább)
- Save configuration (Bejezés)

VNC

VNC szerver

```
apt-get install tightvncserver
```

A felhasználó beállít egy jelszót az eléréshez:

```
vncpasswd
Using password file /home/joe/.vnc/passwd
Password:
Warning: password truncated to the length of 8.
Verify:
Would you like to enter a view-only password (y/n)? n
```

Elindítjuk a szervert:

```
vncserver

New 'X' desktop is servername:1

Starting applications specified in /home/joe/.vnc/xstartup
Log file is /home/joe/.vnc/servername:1.log
```

Kapcsolódás egy VNC szerverhez

```
apt-get install xtightvncserver
```

```
xtightvncviewer 192.168.1.4:1
```

Kapcsolódhatunk a VNC port megadásával:

```
192.168.5.104:5900
```

Témához kapcsolódó csomagok

- tightvncserver - VNC szerver
- xtightvncviewer - VNC kliens
- directvnc - VNC kliens, a framebuffer-t használja megjelenítőnek
- gnome-rdp - Távoliasztal kliens GNOME számára
- gvncviewer - VNC viewer, amely a gtk-vnc programot használja

- gtkvncviewer - Kicsi GNOME alapú VNC kliens
- remmina - Távoliasztal kliens GNOME számára
- tsclient - Távoli asztal kliens GNOME-hoz (front-end)
- vinagre - Távoli asztal kliens GNOME-hoz
- vnc4server - VNC szerver
- xvnc4viewer - VNC kliens
- xrdp - Remote Desktop Protocol (RDP) szerver
- rdesktop - Windows elérése
- krfb - A KDE VNC szervere; bármely VNC klienssel kapcsolódhatunk
- x11vnc - X és parancssoros VNC szerver; sok kapcsoló: man x11vnc

x11vnc

Szerver:

```
x11vnc -rfbauth ~/.vnc/passwd
```

```
x11vnc -forever -bg
```

-forever	Várjon több kapcsolatra is, ne csak az elsőre
-bg	A képernyő beállítása után a háttérbe.
-rfbauth /útvonal/jelszófájlhoz	
-allow host1[, host2..]	Csak adott kliensek kapcsolódhatnak.
-unixpw [lista]	Unix azonosítás, felhasználónév és jelszó. \ A lista opcionális, vesszővel tagolt engedélyezett felhasználók.
-passwd jelszó	azonosítás plain-text jelszóval
-geometry FxV	függőleges és vízszintes geometria
-viewonly	A kliens csak nézhet

systemd szolgáltatás létrehozása

```
cp ~/.vnc/passwd /etc/vnc.passwd
chmod 600 /etc/vnc.passwd
```

```
nano /etc/systemd/system/x11vnc.service
```

```
[Unit]
Description=x11vnc szerver
After=multi-user.target

[Service]
Type=simple
ExecStart=/usr/bin/x11vnc -auth guess -forever -loop -noxdamage -repeat -
rfbauth /etc/vnc.passwd -rfbport 5900 -shared
ExecStop=/usr/bin/x11vnc -R stop
Restart=on-failure
RestartSec=2
```

```
[Install]
WantedBy=multi-user.target
```

```
systemctl enable vnc.service
systemctl daemon-reload
```

Esetleg lehet egy ilyen sor:

```
ExecStart=/usr/bin/x11vnc -display :0 -rfbauth /etc/x11vnc.pwd -shared -
forever -o /var/log/x11vnc.log
```

```
systemctl start x11vnc
```

RSSH

Az rssh egy speciális shell.

Állítsuk be például a joska felhasználó számára:

```
chsh -s /usr/bin/rssh joska
```

A rssh.conf fájlban megmondhatjuk, hogy ez a felhasználó nem tudjon, belépni, csak az scp paranccsal tudjon dolgozni.

```
mcedit /etc/rssh.conf
```

Keressük meg az allowscp-t tartalmazó sort, majd vegyük ke a sor elejéről a megjegyzést:

```
allowscp
```

ClusterSSH

A ClusterSSH lehetővé teszi, hogy több távoli gépen egyszerre dolgozzunk. Maga a ClusterSSH egy kliens program, amivel egyszerre több kapcsolatot nyitok SSH szerverek felé.

Telepítés:

```
apt install clusterssh
```

Legyen három szerver, pluto, neptun és jupter. Ezeken a neveken érhető el, ha használhatnánk akár IP címet is. Mindhárom gépen van egy janos nevű felhasználó.

Ezek után belépés a három szerverre egyszerre:

```
cssh janos@pluto janos@neptun janos@jupiter
```

Négy újabb ablak nyílik. Ebből három az adott szerverekhez tartozó terminálablak. Ugyanakkor egy

kis grafikus ablak is nyílik, ahol a beviteli mezőbe gépelve, a begépelt utasítások mindhárom gépen megjelennek.

A ClusterSSH használatához grafikus felületre van szükség kliens oldalon.

Egyéb információk a ClusterSSH weblapján:

- https://github.com/duncs/clusterssh/wiki

Windowsos távoli-gép

Telepítsük az rdesktop csomagot:

```
apt install rdesktop
```

```
rdesktop 192.168.10.11
```

Ha hitelesítés, tartomány és geometria is szükséges:

```
rdesktop -d zold.and -u janos -p titok -g 1024x768 192.168.10.11
```

A példában a:

- tartománynév: zold.and
- felhasználónév a tartományban: janos
- janos jelszava: titok
- felbontás: 1024×768-as felbontást szeretnénk
- a Windowsos gép IP címe

stunnel4

Az stunnelről

Az stunnel egy több platformos, nyílt forráskódú szoftver, amely lehetővé teszi hálózati protokollok alagutaztatását, TLS kapcsolaton keresztül. Az adott hálózati alkalmazásnak egyáltalán nem kell ehhez ismernie a TLS protokollt. Az hálózati forgalom így biztonságosan, titkosítva haladhat az Interneten. Használható Linux, Windows és Android rendszeren.

Az stunnel segítségével egy privát hálózatban elérhető szolgáltatást az Interneten keresztül biztonságosan elérhetünk bárhonnan.

A projekt weboldala:

- https://www.stunnel.org/

Ehhez hasonló megoldás az SSH alagút.

Cél

Szeretnénk egy belső hálózat webszerverét elérni, titkosított alagúton keresztül. A webszerver csak a belső hálózatról érhető el. Az is célunk, hogy a szervert csak azonosítás után lehessen elérni.

Szerver oldalon

Telepítjük az stunnel4 csomagot:

```
apt install stunnel4
```

Hozzunk létre egy tanúsítványt a titkosított kapcsolat számára:

```
openssl req -new -x509 -nodes -days 365 \
-out /etc/stunnel/stunnel.pem
-keyout /etc/stunnel/stunnel.pem
```

Hozzuk létre a stunnel.conf állományt a /etc/stunnel könyvtárban:

/etc/stunnel/stunnel.conf

```
[http]
client = no
accept = 8000
connect = 192.168.10.2:80
cert = /etc/stunnel/cert.pem
ciphers = PSK
PSKsecrets = /etc/stunnel/psk.txt
```

- client
  - A client beállítással megmondhatjuk szervert vagy klienst hozunk létre
- accept
  - Milyen porton fogadunk el a kliensektől kapcsolatot
- connect
  - Milyen szolgáltatást akarunk alagutaztatni, kiszolgálni
  - a 192.168.10.2 a privát hálózat szolgáltatása
- cert
  - A TLS tanúsítvány helye
- ciphers
  - A hitelesítéshez módja
- PSKsecrets
  - A hitelesítéshez használt állomány helye

A ciphers és a PSKsecrets nélkül is működik, de akkor bárki beléphet.

Hozzuk létre a psk.txt állományt, amiben leírjuk azokat a felhasználókat, akik beléphetnek.

/etc/stunnel/psk.txt

```
janos:01234567890123456789
mari:01234567890123456789
```

A kulcs rész (a kettőspont utáni rész), legyen véletlen karakterek sorozat. Itt most túl egyszerű számsort választottunk. A kulcsnak minimum 20 karakternek kell lennie.

Minden kliens számára létrehozhatunk egy ilyen sort. A psk.txt állomány egy sorát átmásoljuk a kliensre. Lehet például az első sor psk1.txt fájlba.

Állítsuk be, hogy csak a tulajdonos láthassa a fájlt:

```
chmod 600 /etc/stunnel/psk.txt
```

Az stunnelt ezek után kézzel indítjuk:

```
stunnel
```

Szerveroldalon célszerű beállítani, hogy alapértelmezetten elinduljon rendszerindításkor. Ha ezt szeretnénk, szerkesszüka /etc/default/stunnel4 állományt:

```
nano /etc/default/stunnel4
```

Cseréljük le a ENABLED értékét 1-re:

```
ENABLED=1
```

Ezek után a systemd rendszerben engedélyezzük:

```
systemctl enable stunnel4
```

Ellenőrzés:

Ellenőrzéshez használjuk a ps axf és/vagy a ss -lt parancsot:

```
ps axf
```

```
ss -lt
```

Kliens oldalon

A példában az stunnel szerver címe: 195.100.100.2, ahova saját címet kell behelyettesíteni. A beállításokat Linuxon írom le, de Windowson is megvalósítható.

Telepítjük az stunnel4 csomagot:

```
apt install stunnel4
```

Hozzunk létre egy tanúsítványt a titkosított kapcsolat számára:

```
openssl req -new -x509 -nodes -days 365 \
-out /etc/stunnel/stunnel.pem
-keyout /etc/stunnel/stunnel.pem
```

/etc/stunnel/stunnel.conf

```
[http]
client = yes
accept = 9000
connect = 195.100.100.2:8000
cert = /etc/stunnel/cert.pem
PSKsecrets = /etc/stunnel/psk1.txt
```

A PSKsecrets csak akkor szükséges, ha szerver oldalon is be van állítva, de nélküle bárki elérheti a szervert.

A psk1.txt állományban a szerveren lévő psk.txt nevű állomány egyik sorának kell szerepelnie.

/etc/stunnel/psk1.txt

```
janos:01234567890123456789
```

Ne feledkezzünk meg, bonyolultabb kulcsot megadni.

Állítsuk be, hogy csak a tulajdonos láthassa a fájlt:

```
chmod 600 /etc/stunnel/psk1.txt
```

Ellenőrzés:

Ellenőrzéshez használjuk a ps axf és/vagy a ss -lt parancsot:

```
ps axf
```

```
ss -lt
```

Használat

Kliens oldalon indítsuk el az stunnel programot parancssorból:

```
stunnel
```

A kliens böngészőjébe írjuk be:

```
localhost:9000
```

Tanúsítványokról

A tanúsítványok természetesen többféle képen is elkészíthetők. Itt egy másik példát látunk.

```
openssl genrsa 1024 > stunnel.key
openssl req -new -x509 -key stunnel.key -days 3650 -out stunnel.crt
cat stunnel.crt stunnel.key > stunnel.pem
```

XRDP

Az XRDP az RDP protokollt támogató szerver program.

Használatáról leírást az alábbi oldalon találunk:

- xrdp

Függelék

Egyéb csomagok

A dropbear egy SSH szerver. Telepítése:

```
# apt install dropbear
```

Kapcsolódó dolgok

- http://www.teamviewer.com
- https://github.com/creaktive/tsh (Tiny SHell; hátsóajtó)
- https://anydesk.com/hu/ (nem nyílt)
- Guacamole
    - https://guacamole.apache.org/
    - apt install guacamole
- https://remotedesktop.google.com/

SFTP kliens

- https://code.google.com/archive/p/win-sshfs/
- https://github.com/billziss-gh/sshfs-win
    - https://mhogomchungu.github.io/sirikali/
- https://www.nsoftware.com/sftp/netdrive/