

Igang med SSH

Bo Simonsen <bosim05@imada.sdu.dk>

<http://imada.sdu.dk/~bosim05/>



Den 23. februar 2006

Agenda

- Om foredragsholderen og opvarmning
- SSH klienten
- Kryptografi - SSH's virkemåde
- Avanceret SSH
- SSH dæmonen
- SSH i Windows
- Evt. demo af SSH

Spørgsmål ønskes stillet, så snart de melder sig.

Foredragsholderen ønsker at der holdes en 15 minutters pause efter 45 minutters foredrag.

Opvarmning

Om foredragsholderen

- Datalogi Studerende ved IMADA (snart Bachelor of Science)
- Linux bruger siden 1998 - brugt SSH cirka i 5 år af dem.
- Netværksadministrator på Rasmus Rask Kollegiet
- Tidligere formand for SDLUG

Hvad er SSH?

- SSH - Secure shell
- Krypteret Fjernadgang
- Krypteret Filoverførsel
- Mulighed for tunneling

Motivation

- Upålidelige netværk (Hubed netværk, Wifi)
- Tunnels
- Simplifisering af fjernadgang

OpenSSH vs. SSH

Om OpenSSH (<http://www.openssh.org>):

- Udviklet af folkene bag OpenBSD
- OpenSource (BSD Licensen)
- Gratis at benytte
- Medfølger i de fleste Linux distributioner

Om SSH (<http://www.ssh.com>):

- Lukket kildekode
- Ikke gratis - kun til evaluering

Foredragsholderen benytter udelukkende **OpenSSH**, derfor vil foredraget omhandle OpenSSH.

Programmerne

- SSH - Hjertet i OpenSSH implementationen. Etablere en krypteret forbindelse til en remote host, og giver derved shell adgang.
- SSHD - SSH daemonen der gør ssh connections mulige til en host.
- SFTP - En FTP klon til SSH, ergo krypteret filoverførsel.
- SCP - Overførsel af filer mellem 2 hosts.
- ssh-keygen - Generering af nøgler.
- ssh-add, ssh-agent - Forklaring følger.

SSH Clienten

Introduktion

Simpel brug af SSH klienten:

```
bo@indepence:~$ ssh 172.24.0.84
```

```
Password:
```

```
Linux freedom 2.6.10-5-386 #1 Tue Apr 5 12:12:40 UTC 2005 i686
```

```
You have new mail.
```

```
Last login: Wed Jan 18 18:35:50 2006 from 172.24.0.81
```

```
bo@freedom:~$ ls /
```

```
bin      dev      initrd      lost+found  mnt      root     sys      var
boot     etc      initrd.img  media       opt      sbin     tmp      vmlinuz
cdrom    home    lib          misc        proc     srv      usr
```

```
bo@freedom:~$
```

Direkte udførelse af kommandoer:

```
bo@indepence:~$ ssh 172.24.0.84 ls -C /
```

```
Password:
```

```
bin      dev      initrd      lost+found  mnt      root     sys      var
boot     etc      initrd.img  media       opt      sbin     tmp      vmlinuz
cdrom    home    lib          misc        proc     srv      usr
```

```
bo@indepence:~$
```

Kommando linie parametre

Kommandolinie parametre:

- -d <brugernavn>
- -F <konfigurations fil>
- -6/-4 tvinger ssh til at forbinde via ipv6/ipv4
- -1/-2 tvinger ssh til at benytte protocol 1/2
- -p anden port end 22
- -v verbose output, meget nyttigt hvis der er problemer

Konfigurationsfil

SSH clientens konfigurations fil er placeret i `.ssh/config` (Den globale konfigurationsfil, som er gældende for alle brugere er placeret i `/etc/ssh/ssh_config`)

```
host maskine1
hostname meget-langt-hostname.dk
user john_doe
port 255
```

```
host maskine2
hostname 207.208.209.210
user mr_missing_link
port 269
```

ssh maskine1 ⇒ ssh -p 255 -l john_doe meget-langt-hostname.dk
man ssh for alle options

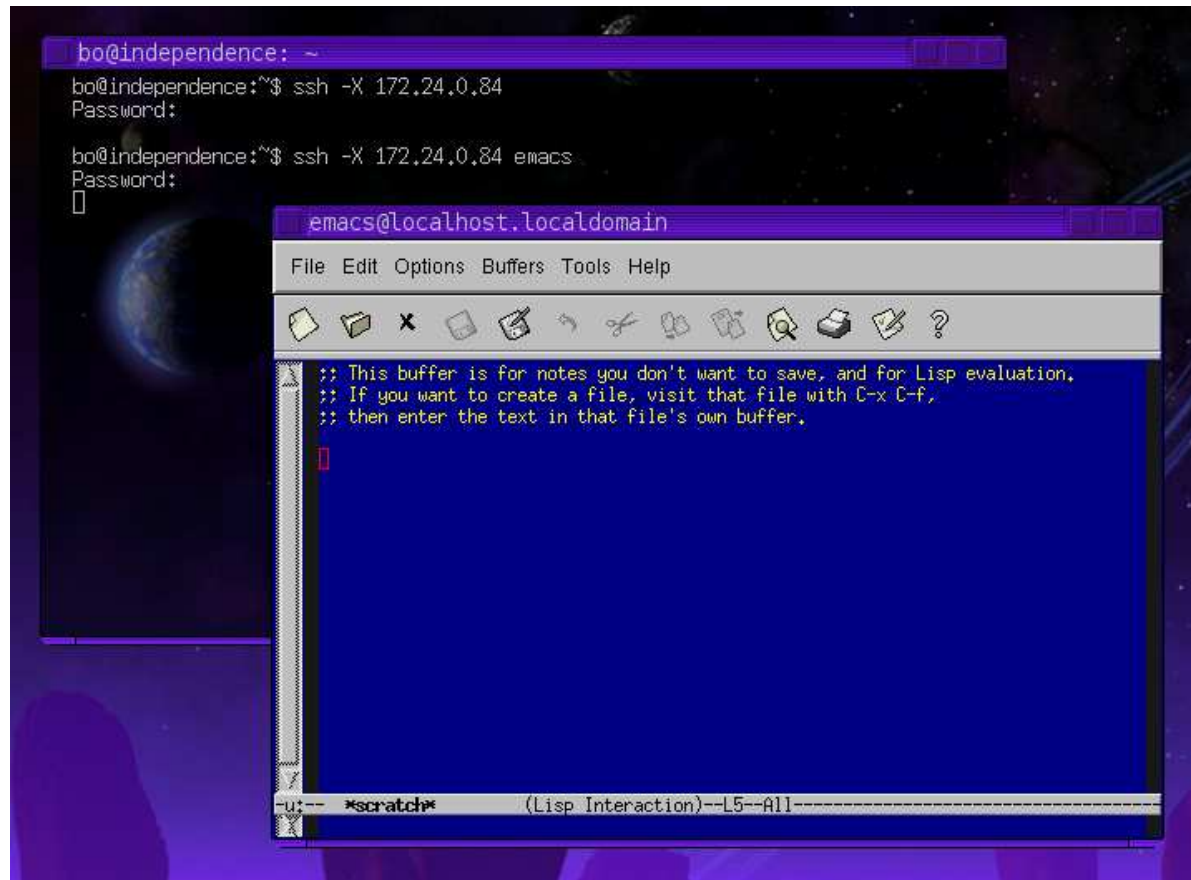
X forwarding

SSH forwarding benyttes således:

```
ssh -X <host> [kommando]
```

Eller ved et entry i ssh config tilføjes

```
ForwardX11 yes
```



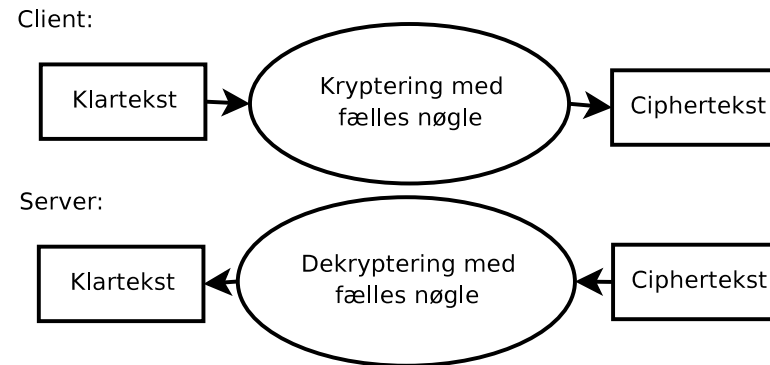
Kryptografi - SSHs virkemåde

Kryptografiske begræber

- Klartekst - Tekst der ønskes overført krypteret
- Nøgle - Krypteringsnøgle, kun ved brug af samme - eller “inverteret nøgle” - kan der dekrypteres.
- Ciphertekst - Krypteret tekst

Vi vil se nærmere på 2 kryptografiske metoder - Symmetrisk kryptering og public/private key kryptering.

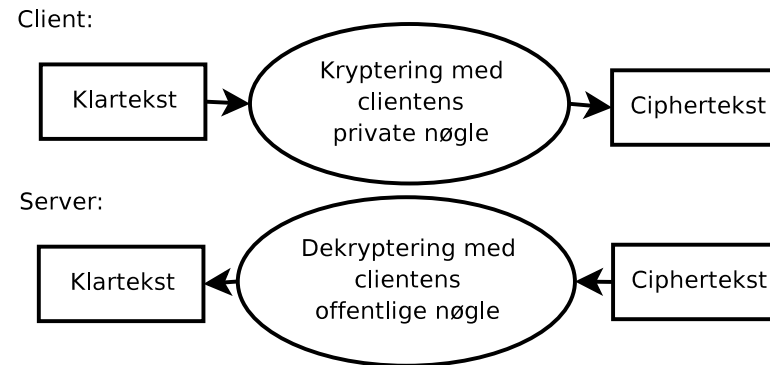
Symmetrisk kryptering



Princippet:

- Begge systemer skal kende nøglen / nøglerne (man kan sagtens bruge 2 forskellige nøgler, de skal dog være kendt af begge systemer).
- Svaghed i det der er single point of failure - hvis client/serveren bliver hacket vil det ramme begge systemer.

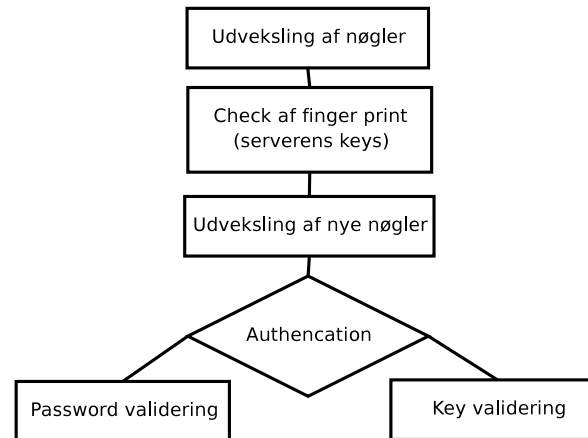
User authentication delen - Public/Private key kryptering



Princippet:

- Krypteret med public key - kan kun dekrypteres af private key.
- Krypteret med private key - kan kun dekrypteres af public key.

Opsætning af SSH forbindelsen



- Udveksling af nøgler - Her udveksles et sæt nøgler v.h.a. Diffie-Hellman som herefter benyttes til at overføre data, krypteret med AES eller en anden symmetrisk krypteringsalgoritme.
- Finger print check - Serverens host key checkes, udfra tidligere oplysninger (.ssh/known_hosts)
- Key validering - Her benyttes en public private kryptering til at sikre sig det er den korrekte user.
- Udveksling af nye nøgler - Serverens host og server key benyttes til at kryptere et tilfældigt nummer, som er key til yderligere kommunikation.

Under sessionen benyttes der en symmetrisk krypteringsalgoritme.

Åbentlyse fordele ved SSH protocol 2

- Integritets check vh.a. md5, sha, ripemd160
- Stærkere krypteringsalgoritmer dvs. AES, 3DES, Blowfish, CAST128, Arcfour (protocol 1 kan AES og Blowfish).
- Bedre handshake med key auth.

Forskelle på key auth:

- SSH protocol 1 - Hvis et nøgle par eksistere på klienten, vil der blive sendt en CHALLENGE krypteret på ovenstående måde, som serveren så dekryptere og kryptere igen og sender tilbage. Hvis dette lykkedes nøgle sættene korrekte. CHALLENGEN vælges tilfældigt.
- SSH protocol 2 - Næsten på samme måde, bortset fra klienten og serveren kender en session identifier, som der "signeres" vh.a. nøgle parene.

Hvorfor bruge offentlig/privat nøgle kryptering

- Meget mere sikkert (da der er både nøgle og passphrase)
- Erstatte password authentication
- Privat nøgle sikret med en passphrase, således at den kun er gyldig ved en korrekt passphrase
- Unix password kan gøres overflødigt ved login
- Ved skift af passphrase ændres der kun i den private nøgle, ergo virker login stadig på remote systemer

Avanceret SSH

Generering af nøgler

Et nøglepar generes ved

```
test@independence:~$ ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key (/home/test/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/test/.ssh/id_rsa.
Your public key has been saved in /home/test/.ssh/id_rsa.pub.
The key fingerprint is:
14:ef:13:04:e2:a9:af:db:cf:98:b8:2c:1d:79:bb:d9 test@independen
```

Vælg en fornuftig passphrase på cirka 10-20 tegn, jo længere passphrase jo bedre.

Herefter vil der blive dannet 2 nye filer i `.ssh` kataloget.

- `id_rsa` - Den private nøgle - sørg for der er permissions 600 på den, ellers kan det få katastrofale følger!
- `id_rsa.pub` - Den offentlige nøgle - kan sagtens ligges på en hjemmeside e.l.

Placering af nøgler

Den offentlige nøgle kopieres over på serveren, før der kan logges ind.

Den placeres i `.ssh/authorized_keys`, med permissions 644, og ejet af brugeren.

Nøglen kan frit overføres ved en plaintext protocol da den ikke er “hemlig”.

Herefter vil et login se således ud

```
bo@independence:~$ ssh apollo
```

```
You are trying to connect to apollo.geekworld.dk, this is a pri  
and any try to compromise security will be logged. So if you ty  
please disconnect.
```

```
Enter passphrase for key '/home/bo/.ssh/id_rsa':
```

```
Linux apollo.geekworld.dk 2.4.20-021stab028.17.777-enterprise #
```

```
....
```

```
No mail.
```

```
Last login: Thu Jan 19 04:50:08 2006 from cpe.atm4-0-7935.0x535
```

```
bo@apollo:~$
```

SSH agenten

SSH agenten gør det muligt at passfrasen huskes, somregl i forbindelse med en X session.

På mange distributioner starter ssh agenten op når X startes, hvis der er dannet et nøglepar.

Dette tjekkes således

```
$ ps aux|grep ssh-agent
bo          12601  0.0  0.2   3128  1136 ?          Ss   14:00
0:00 /usr/bin/ssh-agent sh /home/bo/.Xsession
```

Hvis ssh agenten ikke starter op når X startes, skrives der blot i .Xsession eller .xinitrc

```
exec /usr/bin/ssh-agent /home/bo/.Xsession2
```

Hvor .Xsession2 er et shell script i stil med .Xsession

(I ældre versioner af ssh kan det være lidt kompliseret at få agenten startet, på min hjemmeside findes et script der kan starte den)

ssh-add

Når ssh-agenten er startet skrives der blot

```
$ ssh-add
```

Og passfrasen indtastes, her opnås så at ssh-agenten husker passfrasen og benytter den hvergang ssh har behov for den. Derved skal der ikke indtastes passphrase hvergang der logges ind. Således at:

```
bo@independence:~/tmp2$ ssh apollo who
```

```
You are trying to connect to apollo.geekworld.dk, this is a pri  
and any try to compromise security will be logged. So if you ty  
please disconnect.
```

```
bo          pts/2          Jan 19 08:00 (cpe.atm4-0-7936.0x535f1d06.
```

```
bo@independence:~/tmp2$
```

ssh agent kan dog være problematisk da den interagerer gennem en unix socket der ligger i `/tmp/ssh-<something>/agent.<something>`

ssh-askpass

ssh-askpass er et grafisk program der kan benyttes til at indtaste passfrasen når X startes.

I .Xsession skrives der:

```
export SSH_ASKPASS=/usr/bin/ssh-askpass
```

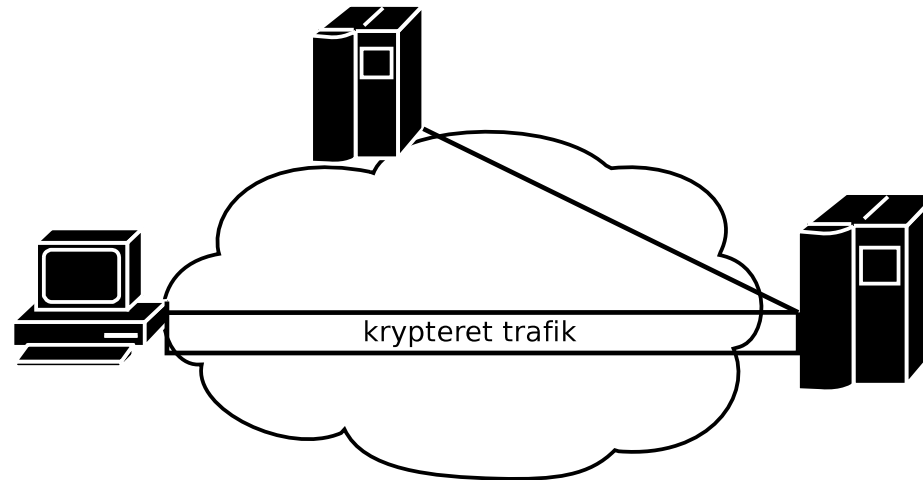
```
ssh-add &
```

ssh-askpass ser således ud



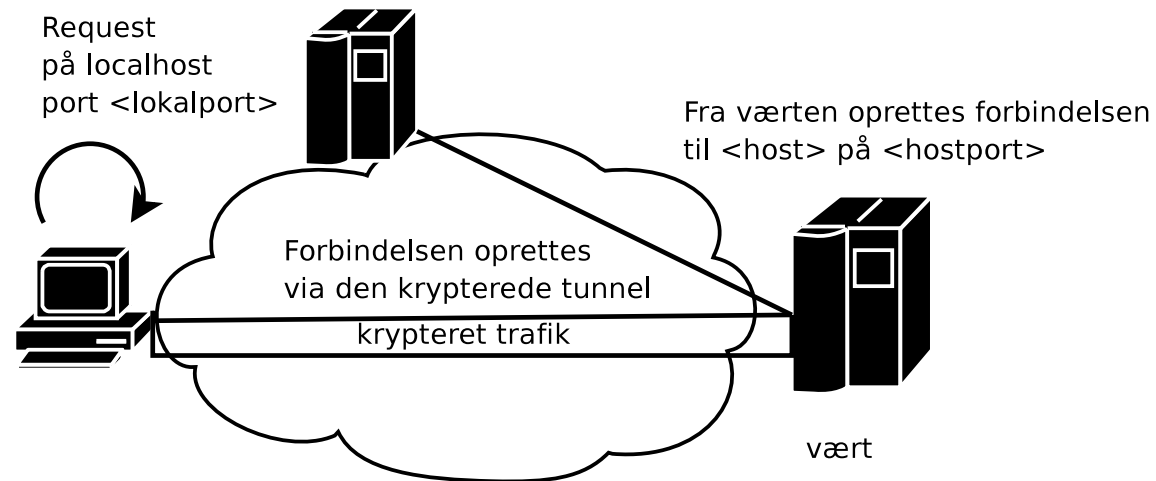
Tunneling

Hvad er tunneling?



Der etableres en krypteret tunnel til en vært, hvorefter værten opretter forbindelse til andre host, som nødvendigvis ikke er krypteret. Dette kalder vi en lokal tunnel. Vice versa, dvs. en vært modtager en request og denne sendes via den krypterede tunnel til klienten, kalder vi en remote tunnel.

Tunneling - Lokal



En lokal tunnel oprettes v.h.a.

```
ssh -L <lokalport>:<host>:<hostport> <værtsnavn>
```

Det der sker her er at:

localhost:lokalport → værtnavn → host:hostport

Hvis man ønsker at opsætte en lokal tunnel for en host hvergang man ssh'er kan man med fordel benytte **ssh/config**

```
Host <some host>
```

...

```
LocalForward <lokalport>:<host>:<hostport>
```

Anvendelser og Eksempel på brug

Anvendelsesmuligheder:

- Krypteret trafik der normalt sendes ukrypteret (pga. manglende kryptering i protocolen).
- Tilgå afsikrede zoner (fx. remote administration af en nat router, hvis der forefindes en server bag nettet der kan sshes til).

Eksempler:

- Eksempel 1:
Hvis man ønsker at benytte IMAP på en server, man har ssh adgang til, men serveren ikke understøtter IMAP-SSL kan man lave en local tunnel, der gør at trafikken bliver krypteret.

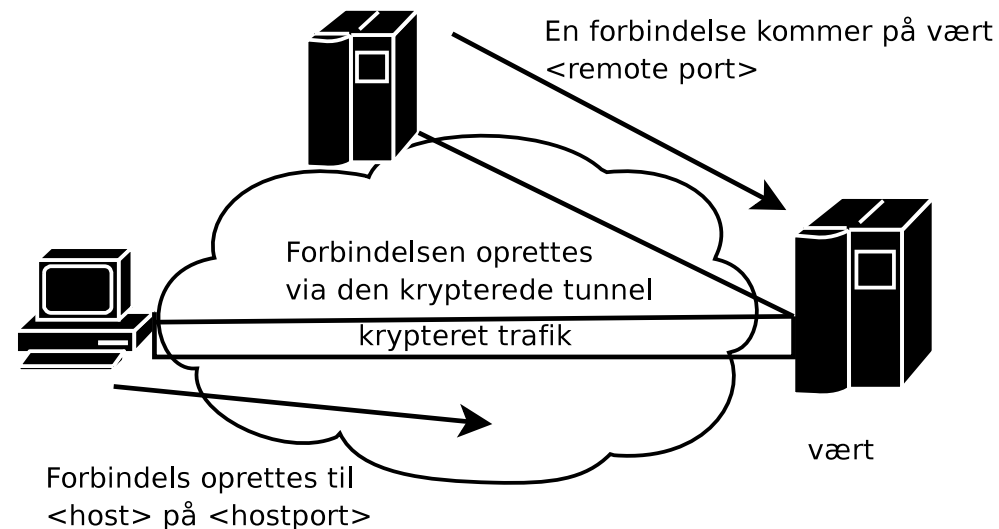
```
ssh -L 10143:localhost:143 <vært>
```

- Eksempel 2:
Administration af router med ip 192.168.0.1 på port 80

```
ssh -L 10080:192.168.0.1:80 <vært>
```

Tunneling - Remote

Remote tunneling gør det muligt at få værten til at lytte på en port, og forespørgseler på denne port vil blive sendt via tunnelen og en forbindelse vil blive oprettet på clienten.



En remote tunnel laves ved:

```
ssh -R <remote port>:<host>:<hostport> <vært>
```

Tilsvarende for `.ssh/config`

```
Host <somehost>
```

...

```
RemoteForward <remote port>:<host>:<hostport> <vært>
```

Default lytter værten på localhost men den kan sættes til at lytte på alle interfaces.

Anvendelses muligheder og eksempler

Anvendelses muligheder:

- Tilgå maskiner bag ved NAT, uden brug af port forwarding

Eksempler:

- På kollegiet er jeg bag ved NAT, der benyttede jeg RemoteForward til at tilgå min maskine når jeg ikke var hjemme v.h.a.

```
ssh -R 10022:localhost:22 apollo
```

Når jeg så logger ind på apollo er jeg istand til at tilgå min maskine v.h.a.

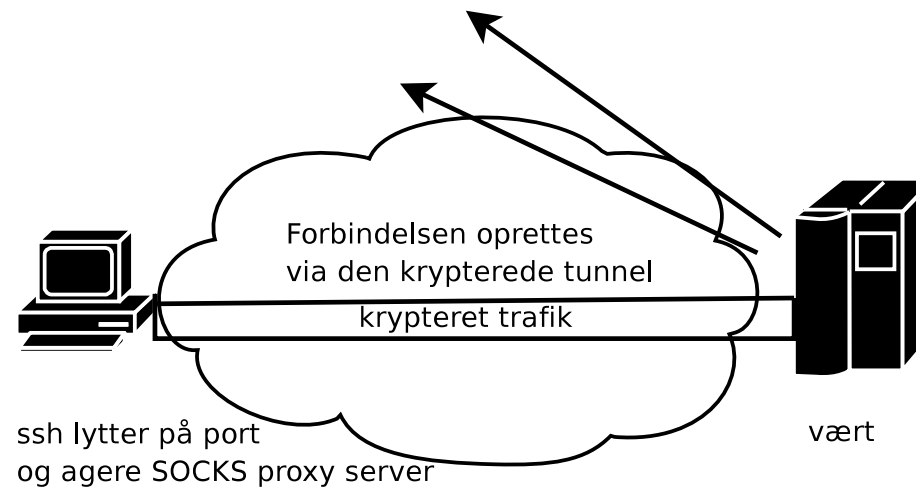
```
ssh -p 10022 localhost
```

Bemærk denne feature er smuk, men farlig! (som så meget andet i livet! ;-))

Tunneling - SOCKS proxy

SSH kan også agere SOCKS proxy server, dette benyttes således:

```
ssh -D <localport> <host>
```



Fra `.ssh/config` benyttes

```
Host <host>
```

...

```
DynamicForward <localport>
```

Alle forbindelser på `<localport>` vil blive oprettet fra `<host>`

Et par ord om tunneling

- Sikkehedsmæssigt farligt
- Problematisk - enten tillades de ellers gør de ikke
- Med `gatewayports` er de endnu mere farlige

SSH tilbyder 2 muligheder for filoverførsel.

- SCP - kopiering af enkelte filer
- SFTP - FTP klon

Filoverførsel med SFTP

SFTP (Secure File Transfer Program) benyttes således

```
bo@independence:~$ sftp apollo
Connecting to apollo...
sftp> mkdir tmp
sftp> cd tmp
sftp> ls
```

SFTP benytter STDIO, ergo åbnes der blot en SSH session. Derfor gives options v.h.a. SSH fx.

```
ssh -o ``Port <alternativ port>`` vært
```

Filoverførsel med SCP

SCP (Secure CoPy) er en rcp klon, som virker ligesom cp, blot med copiering mellem 2 hosts, eller den lokale maskine og en remote host.

Syntaxen er følgende:

```
scp -o <ssh options> bruger@src-host:fil1 fil2 .. filn bruger@d
```

Fx.

```
scp backup.tgz apollo
```

Herved overføres backup.tgz til apollo (bemærk denne er defineret i .ssh_config)

Escape Character

Escape charecteren benyttes under en kørende ssh session. På samme på som med fx. telnet der er escape charecteren ctrl-]. Følgende kommandoer er tilgængelige

- ~. - Afbryder forbindelsen - praktisk når den hænger
- ~^ Z - Sætter ssh i baggrunden.
- ~# - Lister forwards
- ~C - Kommando linie - giver mulighed for forwards
- ~& - Når der logges ud sættes ssh i baggrunden så forwards stadigt virker.
- ~? - Lister alle options
- ~~ - Escaper således at det giver ~.

Escape charecteren er default som ses ovenstående, denne kan overrides til at være noget andet

Dette gøres i ssh_config således:

```
EscapeChar <escape charecter>
```

Fejl meddelelser

```
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@ 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 the RSA host key has just
been changed.
```

...

Hvad gør man?

- Sikre dig at maskinen ikke er blevet geninstalleret
- Hvis det er din egen maskine - geninstaller e.l.
- Skift nøgler hvis intet af ovenstående er sket.

Generelt debug brug

```
ssh -V <host>
```

Viser alt hvad der sker under handshake.

SSH serveren

Med SSH laves langt den meste konfiguration i klienten. SSH dæmonens konfigurationsfil er placeret i følgende katalog `/etc/ssh/sshd_config`.

Alternativ port

SSH kan opsættes til at lytte på en alternativ port end 22, ved at benytte

Port 443

Her sættes sshd til at lytte på port 443, som benyttes til secure http, hvis man er placeret på et restriktivt netværk er der somregl altid åbent for port 443.

Listen address

Hvis en maskine har flere interfaces, og der af fx. sikkerhedsmæssige årsager ønskes, at der blot lyttes på et interface, kan det gøres således.

```
ListenAddress 217.210.94.200
```

En anvendelsesmulighed kunne være at man kun ønsker ssh adgang fra det interne interface på en firewall (SSH er et ret sikkert stykke software, så hvorfor?! ;-))

X forwarding

I sshd kan der tillades om klienterne skal have mulighed for at lave X11Forwarding, hvilket kan være meget nyttigt, hvis en maskine står placeret på en lille båndbredde / trafik afregnet båndbredde, da X trækker ret meget.

X11Forwarding slås til og fra v.h.a.

```
X11Forwarding [yes|no]
```

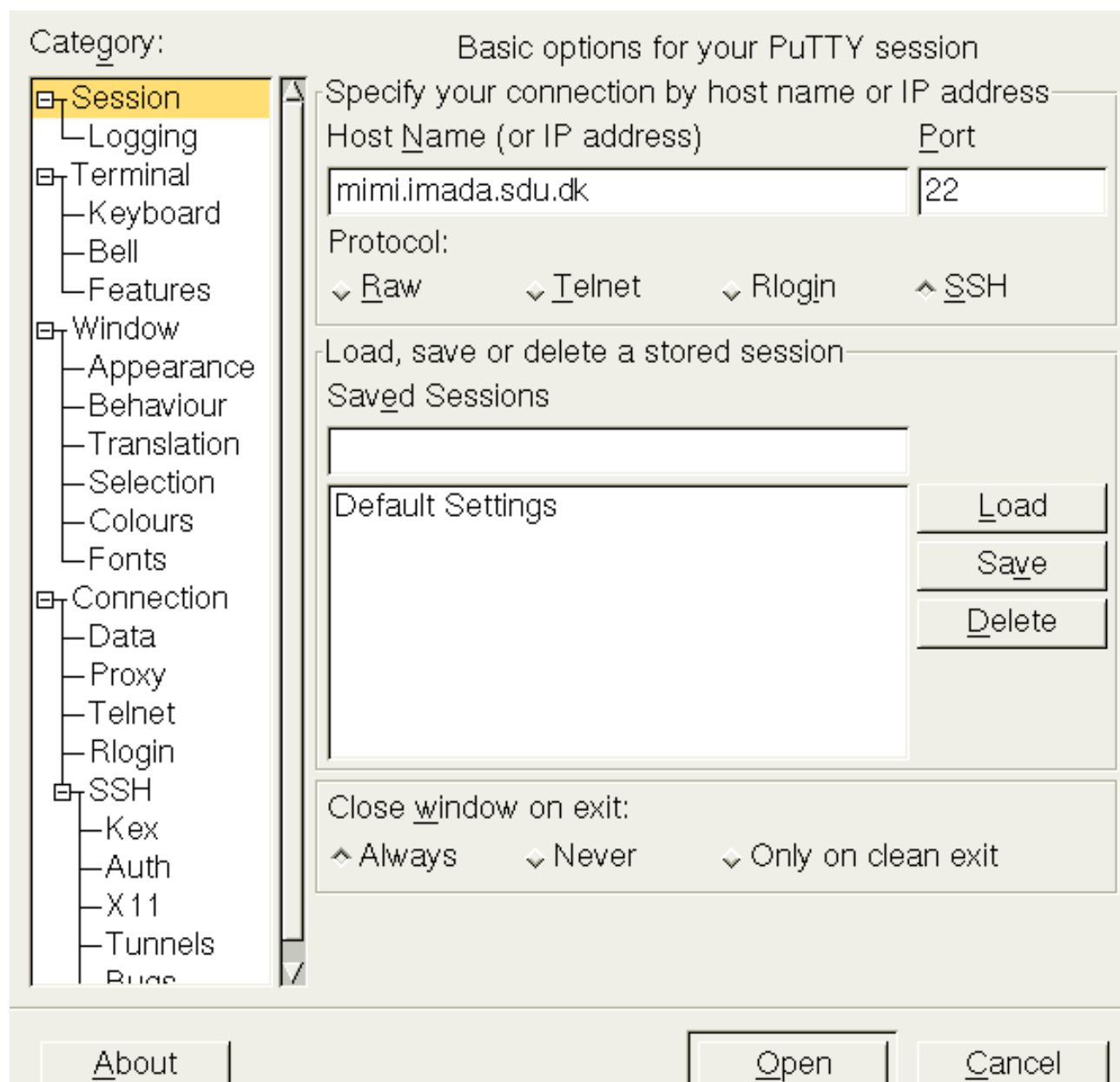
Sikkerheds options

Følgende options kan "hardne" sshd

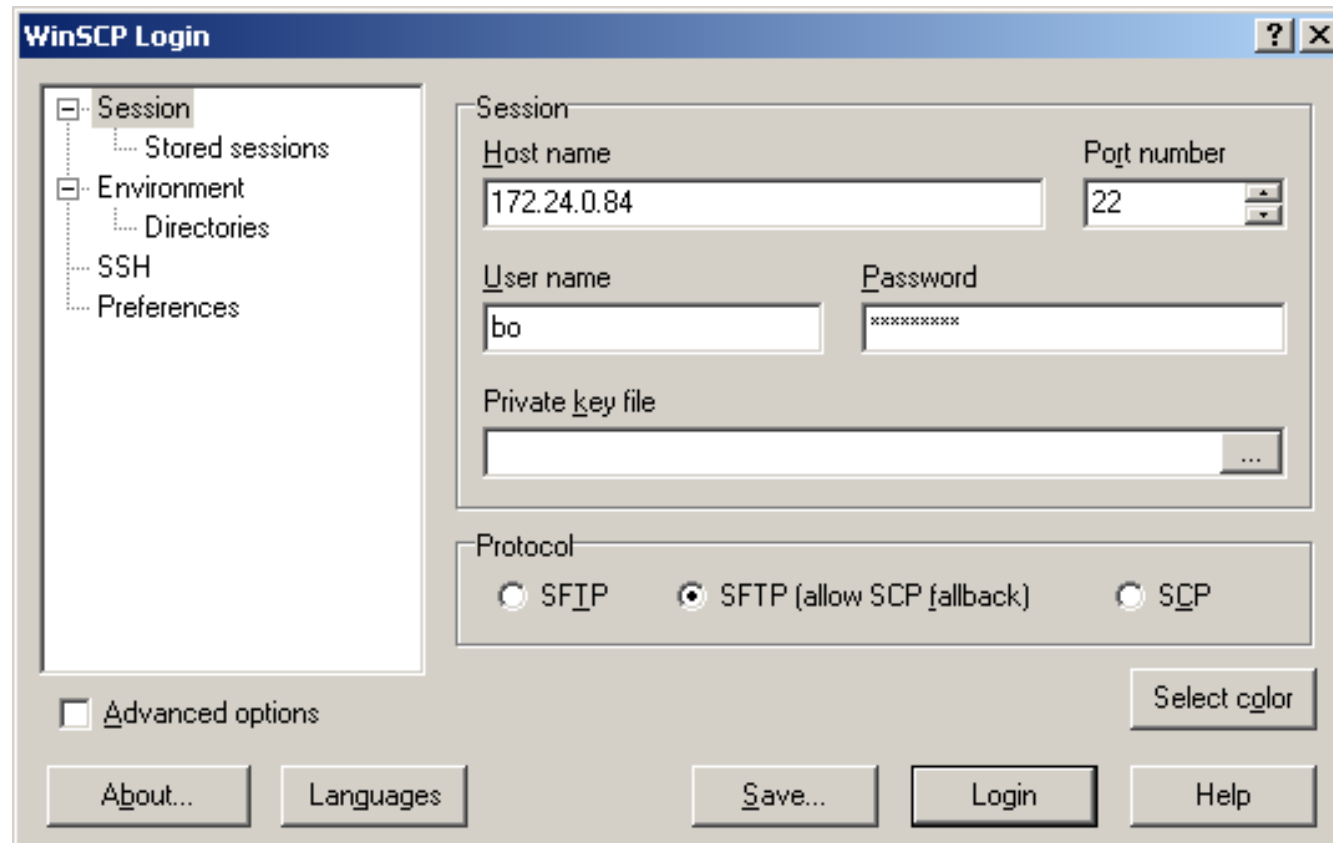
- `PermitEmptyPassword [no|yes]` - anbefales absolut ikke, hvis maskinen er placeret på nettet. Default er no
- `PermitRootLogin [no|yes]` - anbefales ikke, da det giver en ekstra sikkerhed man skal logge ind som user og herefter benytte `su / sudo` (Helst!) til at opnå root adgang.
- `GatewayPorts [no|yes]` - anbefales denne sættes til no, da denne option gør at `RemoteForwards` lytter på `ListenAddress`, hvilket er ret farligt hvis man ikke stoler 100% på sine brugere (hvilket ssh ikke er god til hvis man ikke gør)!
- `Protocol 2` - Uden protocol option tillader ssh både protocol 1 og 2, der er visse fejl i protocol 1, hvilket gør at det ikke anbefales at tillade protocol 1! (2,1 angiver helst protocol 2 men protocol 1 tillades også)
- `ChallengeResponseAuthentication [no|yes]` - Hvis denne sættes til no tillades password login ikke, kun login med RSA/DSA nøgler. Anbefales, da det giver væsentligt bedre sikkerhed!

SSH i Windows

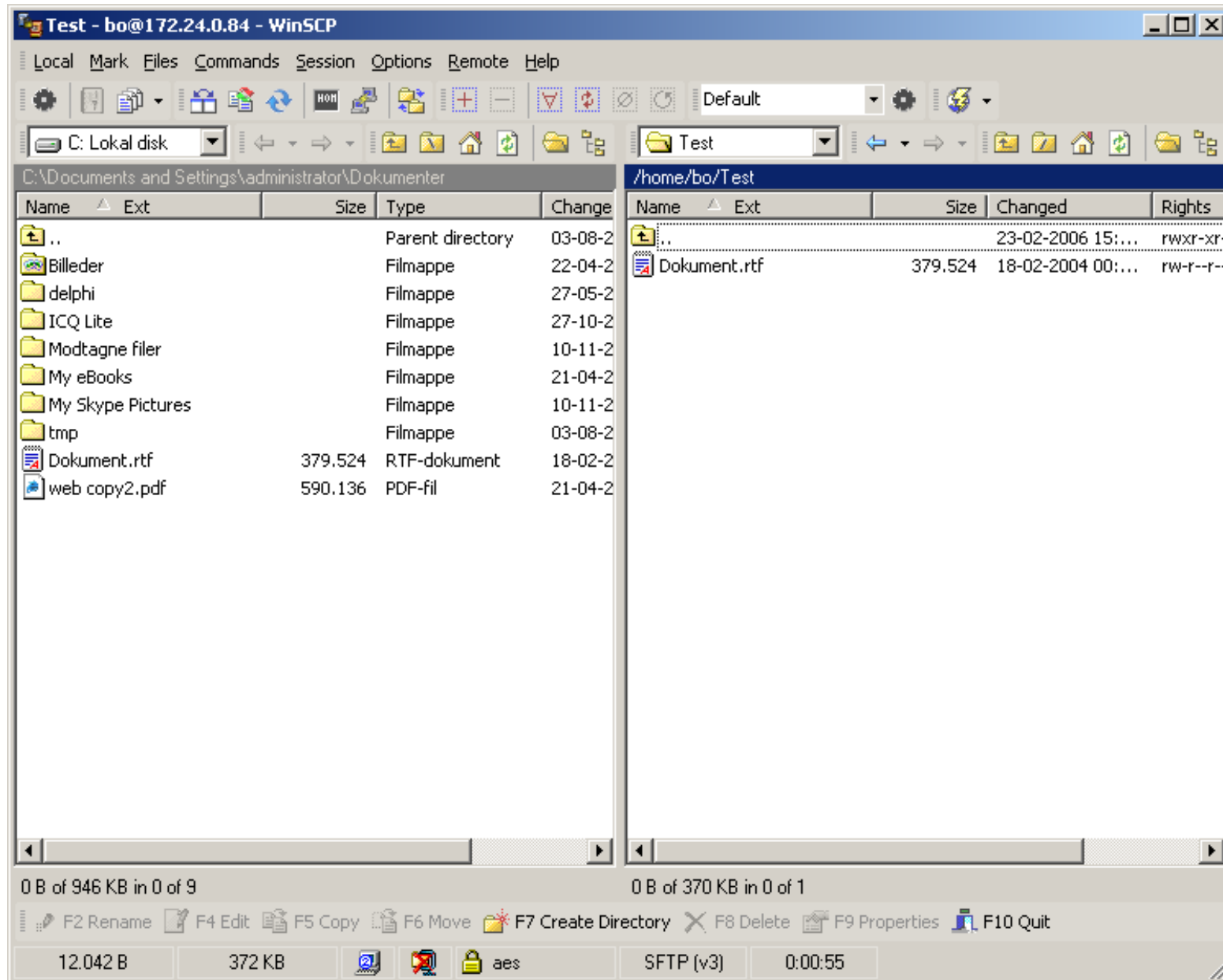
Putty fås også til Linux, men den er absolut mest brugt af Windows folk til at tilgå UNIX maskiner vh.a. ssh



WinSCP (<http://www.winscp.net>) benyttes til SCP/SFTP filoverførsler



WinSCP - fortsat



Afslutning

Yderligere læsning

- man 1 ssh
- man 5 ssh_config
- man 5 sshd_config
- Diffie-Hellman -
<http://www.rsasecurity.com/rsalabs/node.asp?id=2248>
- RSA kryptering -
http://www.econ.au.dk/fag/8020/f03/kryptering_RSA.htm

Kommentare?

- Kommentare på foredraget ønskes. Enten nu eller via mail.
- Ubesvarede spørgsmål?
- Tak fordi i kom - håber det har gjort jer lidt klogere. :-)