Debian/GNU Linux Remote Services

Remote Login Services, Remote Desktops

Károly Erdei

December 6, 2009

Agenda

Agenda

Remote Login Services

Application services to use remote hosts interactively

- Scenario: remote host offers interesting services:
  - Programs installed on remote host
  - Files located on remote host
  - Resources (CPU, memory, disk) provided by remote host
- Goal: use these remote services from local host
  - Use local host as a terminal to login to remote host
  - Run programs/commands on remote host
  - See output on local host (either Ascii terminal output or graphical output by X clients or some other way (vnc,rdp))
- Powerful server computer may be used from many clients

Remote Login Services

General application structure:

- Remote host runs a login service, a server
- client runs terminal program that contacts remote login service
- Login service creates process on remote host to execute commands on behalf of the user
- User enters commands in terminal program; commands are transferred to remote process and executed on remote host output transferred back to local host

Relevant protocols/systems:

- TELNET (TCP/IP); rsh/rlogin (Unix/Linux), outdated
- SSH suite: ssh/slogin (secure shell, secure login)
- X-Windows X11 (network-transparent GUI)
- Real VNC (virtual network computing/console)
- MS Windows Terminal Server (remote desktop)

The Remote Login Server - an application program

TELNET or SSH Server

Process

- Master server waits for new connection requests SSH: port 22
- For each connection, it spawns a slave server to handle the connection
- Multiple sessions (from the same or different cliens) may be active at the same time
- Slave server handles the connection
  - transfers data from local keyboard to remote host and outputs data from remote host on the local display
- OS must provide pseudo terminal for the slave server
  - Entry point to transfer characters to OS as if they came from a keyboard
  - Used by application programs like the TELNET server, SSHD, etc.

Using TELNET for testing services

Testing services with TELNET

- At RISC used only for testing services on remote computer
- Giving port number as parameter for Telnet: 25, 80, 110,etc.
- Checking if a mail server is running

```
uhu:~> telnet bullfinch.risc.uni-linz.ac.at 25
Trying 193.170.37.222
Connected to bullfinch.risc.uni-linz.ac.at.
Escape character is '^]'.
220 bullfinch.risc.uni-linz.ac.at ESMTP Sendmail 8.13.8/8.13.8/Debi
Sun, 19 Oct 2008 13:11:27 +0200; (No UCE/UBE) logging access from:
cm64-139.liwest.at(OK)-cm64-139.liwest.at [212.241.64.139]
quit
221 2.0.0 bullfinch.risc.uni-linz.ac.at closing connection
Connection closed by foreign host.
uhu:~>
```

Telnet is outdates

SSH is the successor

TELNET and Rsh/Rlogin outdated

- because of security problems
- All data are transferred in clear text
- Any listener between client and remote server can read everything
- True for any unencrypted connection, think on http !
- Telnet/rlogin not available at RISC anymore (remark: telnet-ssl)

Replacement: Secure Shell (ssh, slogin)

- SSH suite is the modern replacement of TELNET and rlogin
- standard protocols for secure remote access over IP networks (RFCs: 4251-5254)
- All data are encrypted before they are transferred via IP
- Commercial implementations: `www.ssh.com`
- Free implementations: `www.openssh.org`, `www.putty.org`, etc.

Agenda

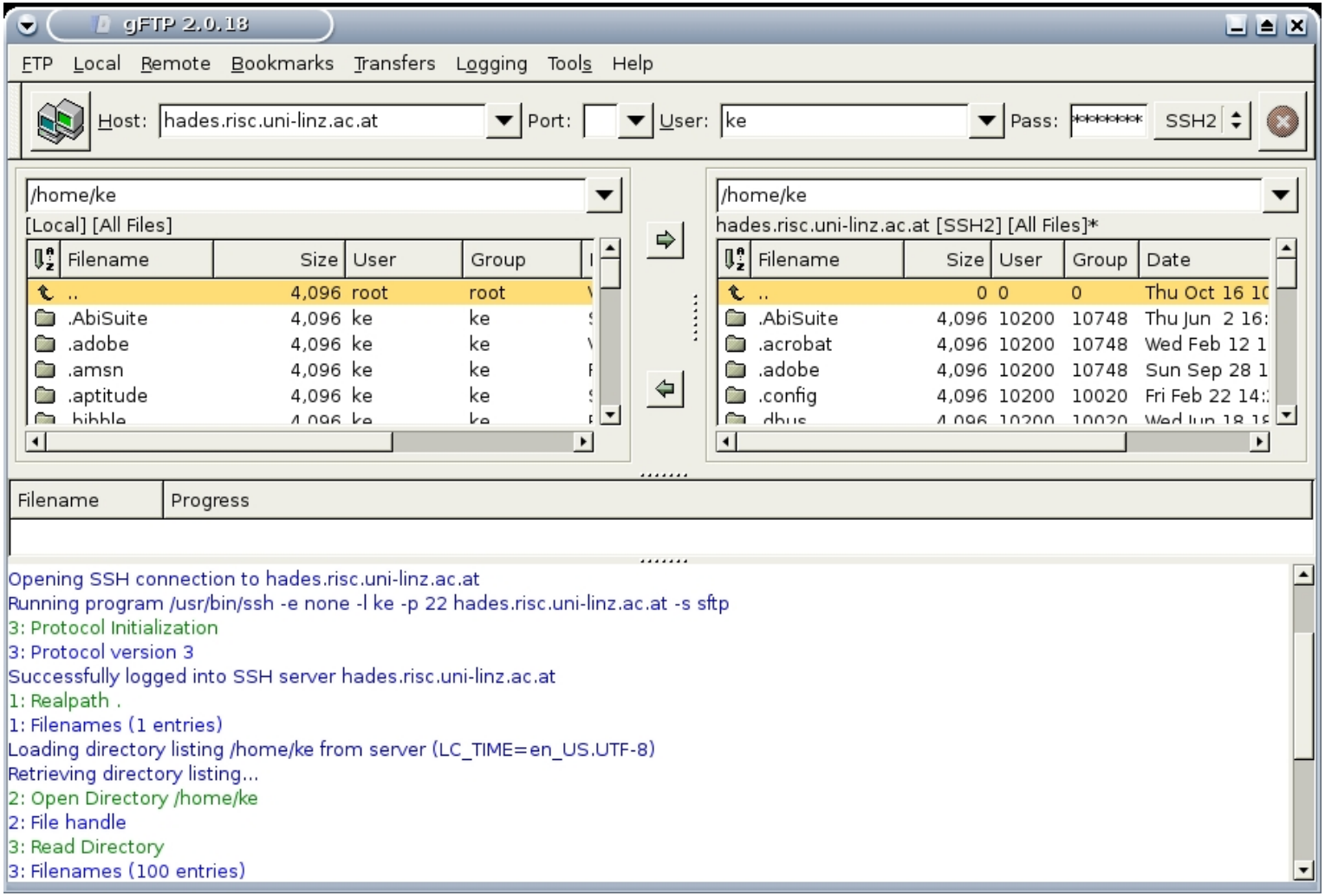
File Services

File transfer, File sharing

Application services to access files on remote hosts

- File transfer
  - Files are copied from one host to another
  - sftp (secure ftp), scp (secure remote copy), FTP is outdated !
  - Graphical tools: gftp, (kasablanca, etc.)
- File sharing
  - Files are accessed from a central server
  - Files are stored and backuped on central file server
  - Client applications operate on remote files like on local files
  - Transparent file access is provided by network file systems
  - NFS (Network File System), SMB (Server Message Blocks)

FTP with gftp



NFS (Network File System)

NFS: access to remote files

- Developed by Sun Microsystems
- Used in many Intranets to interconnect file systems
- Mainly for Unix/Linux computers
- Remote file system can be accessed like local files
  - A remote file system is mounted to an empty local directory
  - Files below this directory can be used like local files
  - No special file transfer commands needed, no file duplication arises
- Implemented on top of UDP

For security reasons, only used within an administrative domain

Agenda

SSH features

The SSH suite

SSH - a client-server solution for network security

- client-server solution for network security
  - encryption: all data will encrypted before sending from localhost to remote computer and vice verse
  - transparent for the user (does not notice background activities)
  - client side: login, authentication, data transfer, command execution

SSH features

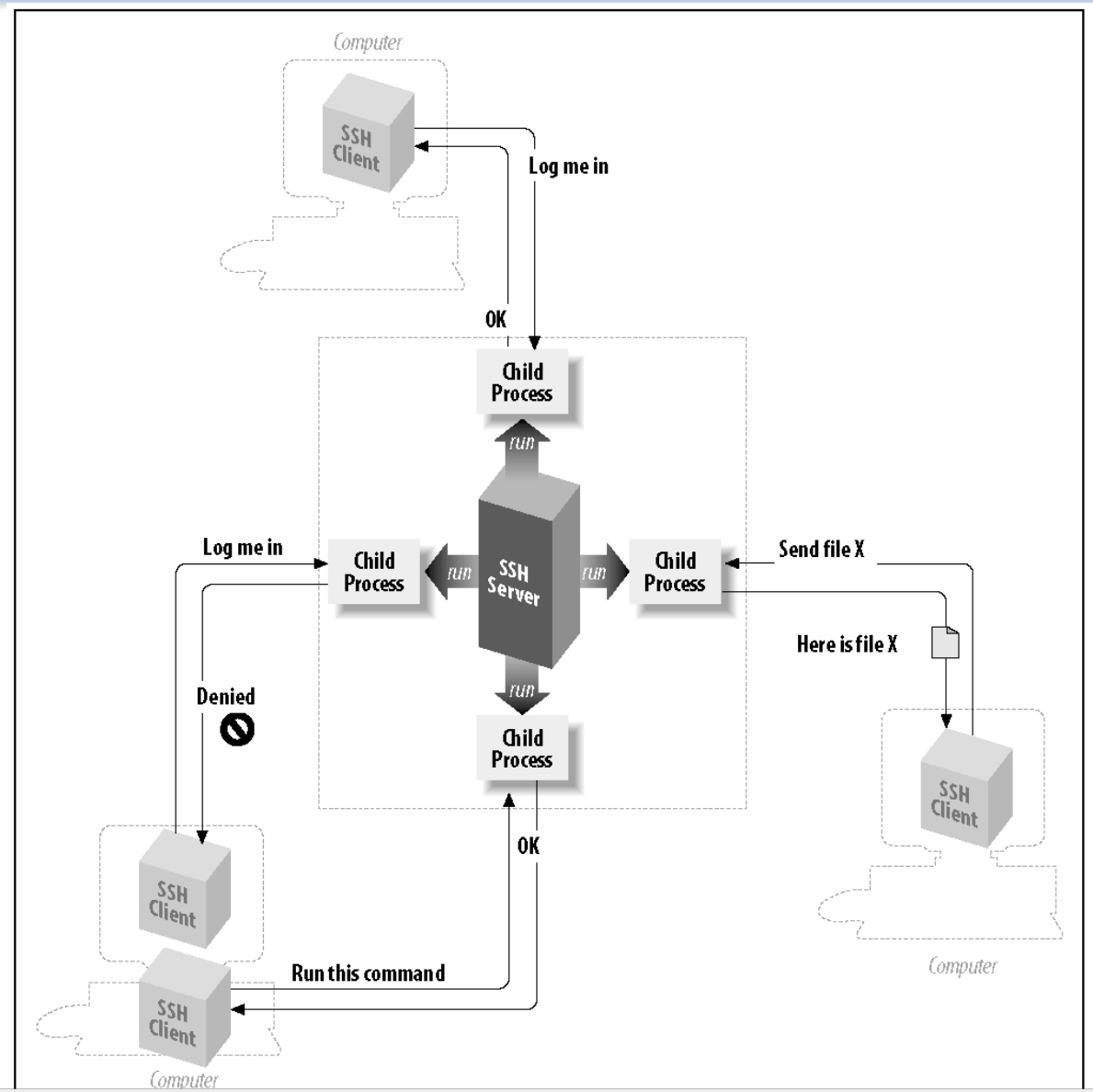
- it is a protocol: describes how to conduct secure communication over a network
- full, secure replacement for FTP and Telnet and the UNIX r-series of commands: rlogin, rsh, rcp, rexec
  - creates a secure channel for running a shell on the remote computer
  - sftp, scp is integrated in the protocol
- supports more authentication methods: password, public key, certificate, smart card, PAM and SecurID

SSH features

Security

- uses multiple high security algorithms and strong authentication methods
  - prevents such security threats as identity spoofing and man-in-the-middle attacks
  - man-in-the-middle attack: changing the IP in the packet you communicate with the remote computer, stating: I'm the remote computer
- Transparent and automatic tunneling of X11 connections
- Port forwarding or SSH tunneling: for arbitrary TCP/IP-based applications, such as e-mail
- Multiple channels that allow
  - to have multiple terminal windows and file transfers going through one secure and authenticated connection

The base services of SSH



Complete Structure of the SSH protocol

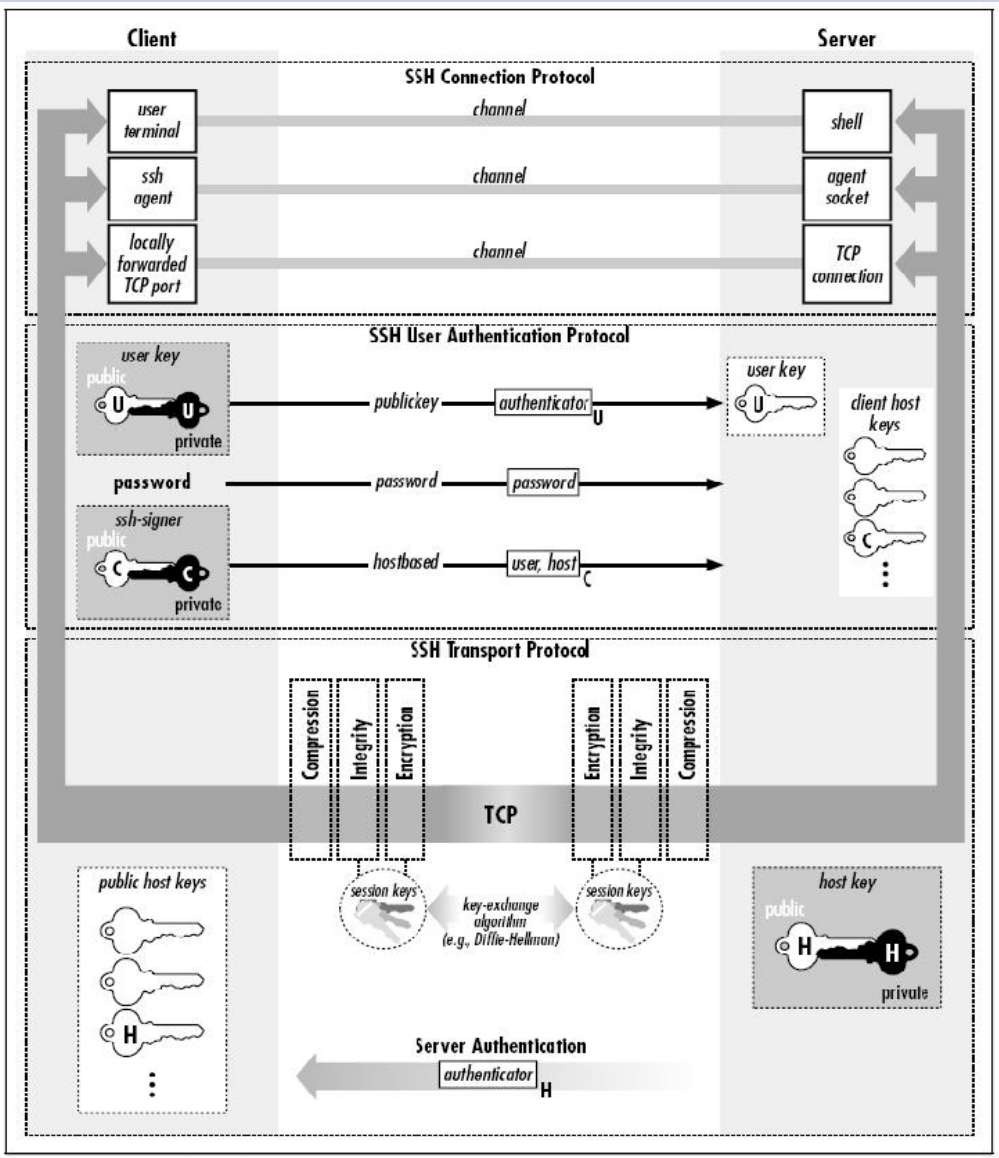


Figure 3-4. SSH-2 architecture

The structure of the SSH-2 Protocol

Very clean 3-layer internal architecture (RFC 4251)

- Transport Layer (RFC 4253)
  - initial key exchange, server authentication, data confidentiality, data integrity, compression, key re-exchange (algorithm negotiation, session-ID, privacy)
- User Authentication Layer (RFC 4252)
  - Client Authentication: provides various authentication methods (public key, host bases, password, etc.)
- Connection Layer (RFC 4254)
  - defines the logical channels and the requests to handle the services like: secure interactive shell session, X11 forwarding, TCP/IP forwarding (channel multiplexing, pseudo terminals, flow control, remote program execution, authentication agent forwarding, terminal handling, etc.)

The Components of the SSH suite

SSH binary programs, scripts

```
uhu:~> dpkg -L openssh-client | grep bin
/usr/bin
/usr/bin/ssh
/usr/bin/scp
/usr/bin/ssh-add
/usr/bin/ssh-agent
/usr/bin/ssh-keygen
/usr/bin/ssh-keyscan
/usr/bin/sftp
/usr/bin/ssh-vulnkey
/usr/bin/ssh-copy-id
/usr/bin/ssh-argv0
/usr/bin/slogin
uhu:~>
```

The Components of the SSH suite

SSH man page

```
uhu:~> ssh --help
usage: ssh [-1246AaCfgKkMNnqsTtVvXxY] [-b bind_address]
           [-c cipher_spec]
           [-D [bind_address:]port] [-e escape_char] [-F configfile]
           [-i identity_file] [-L [bind_address:]port:host:hostport]
           [-l login_name] [-m mac_spec] [-O ctl_cmd] [-o option]
           [-p port]
           [-R [bind_address:]port:host:hostport] [-S ctl_path]
           [-w tunnel:tunnel] [user@]hostname [command]
uhu:~>
```

The SSH suite

SSH parameters

Parameter of SSH

- If command is specified, it is executed on the remote host instead of a login shell.
- -F configfile
  - default configuration file: ~/.ssh/config
- -i identity_file (private key)
  - default for ssh2: ~/.ssh/id_rsa ~/.ssh/id_dsa
- -p port (to connect to on the remote host)
- -v Verbose mode
  - to debug problems and see the progress of connection
- -l username (ssh -l sysadmin atlantis)
- username@hostname (ssh sysadmin@atlantis)
- -X (X11 forwarding: ssh -X sysadmin@gorilla)

The SSH suite

SSH examples

```
hades:sysadmin!8> ssh ke@bullfinch
ke@bullfinch's password:
Linux bullfinch 2.6.24-etchnhalf.1-686 #1 SMP Thu Nov 5 02:25:56 UTC 20
......... deleted ........
No mail.
Last login: Sat Nov 21 17:45:11 2009 from hades.risc.uni-linz.ac.at
Sat Nov 21 17:45:12 CET 2009
bullfinch>

hades:sysadmin!12> ssh gonzales who
cschneid pts/2        Nov 18 12:11 (ozelot.risc.uni-linz.ac.at)
cschneid pts/3        Nov 19 15:33 (ozelot.risc.uni-linz.ac.at)
cschneid pts/4        Nov 18 13:52 (ozelot.risc.uni-linz.ac.at)
cdoench  pts/5        Nov 20 09:50 (dog.risc.uni-linz.ac.at)
mkauers  pts/6        Nov 21 12:01 (fennek.risc.uni-linz.ac.at)
hades:sysadmin!13>
```

The SSH suite

ssh with command

X11 forwarding wird activated

```
hades:sysadmin!13> ssh -X gonzales
Linux gonzales 2.6.26-2-amd64 #1 SMP Thu Nov 5 02:23:12 UTC 2009 x86_64
Last login: Fri Nov 20 15:24:10 2009 from tc14.risc.uni-linz.ac.at
gonzales:sysadmin!1>
gonzales:sysadmin!1> mathematica &
[1] 18455
gonzales:sysadmin!2> kill -TERM 18455
gonzales:sysadmin!3>
```

Agenda

SSH tunneling

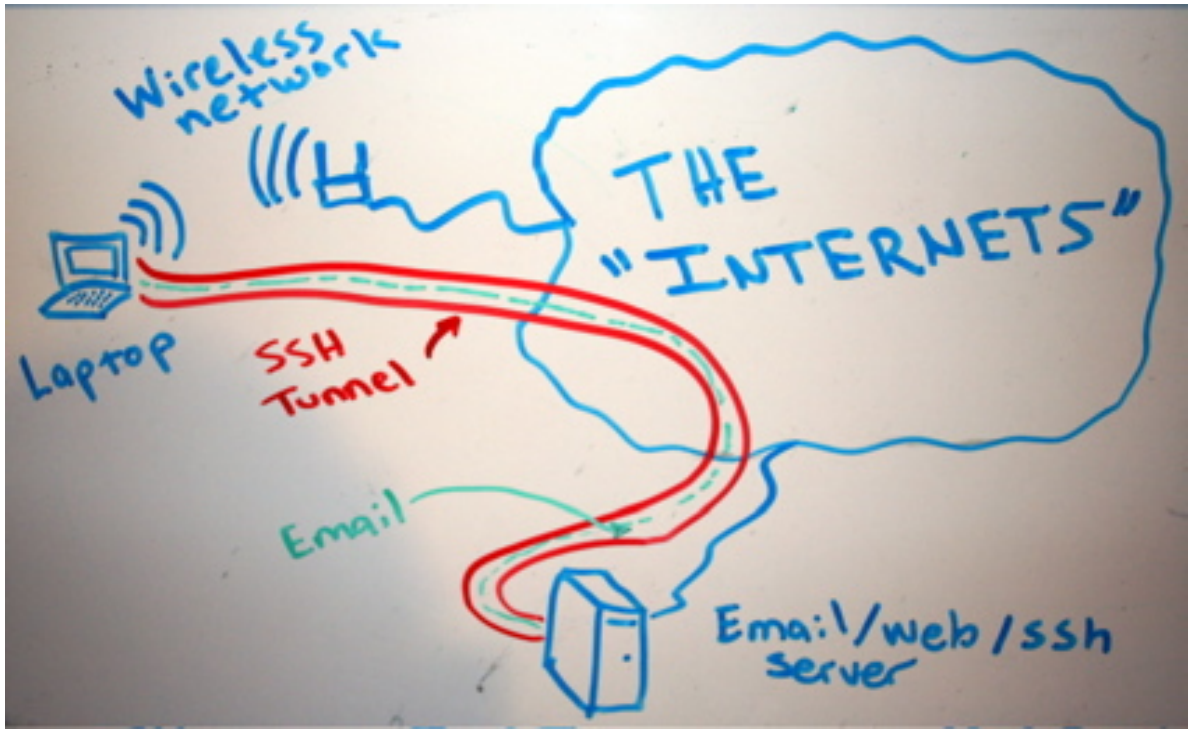
What is an SSH tunnel

- tunnel is a networking term, means a connection, usually encrypted
- connects two computers together across another usually untrusted network

Why do we need it - the Internet is very insecure !

- your laptop/home computer connects to another computer without encryption
- some protocols do have encryption built in, some do not
- your email client, your ftp program, VNC client, etc.
- Never use clear text connections !
- definitively not for login/password data!
- configure SSH tunnel for your connections!

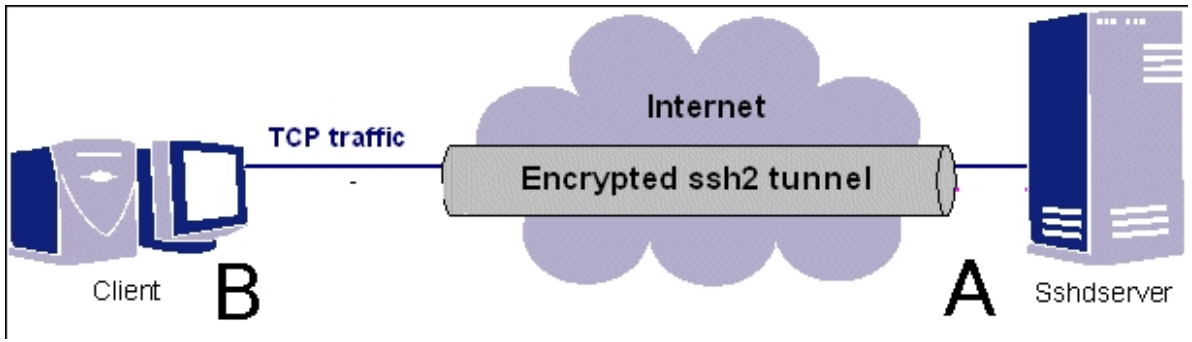
Secure WLAN connection through the Internet



SSH tunnel through the Internet

SSH Tunnel Topology

- Client B (laptop, PC at home, etc) connects using local ports
- Server A running the sshd server program
  - mail server: port 25 smtp; VNC server: port 5901
- through an SSH tunnel - encrypted connection !



How to make SSH tunnel in Linux

basic version:

- ssh -L localport:hostname:hostport hostname
  - Specifies that the given port (localport) on the local, the client host is to be forwarded to the given host (hostname) and port (hostport) on the remote side (hostname).
  - ssh -L 22000:bullfinch.risc.uni-linz.ac.at:143 bullfinch.risc.uni-linz.ac.at
- ssh -L localport:hostname:hostport remotehost
  - Specifies that the given port (localport) on the local, the client host is to be forwarded to the given host (hostname) and port (hostport) on the remote side (remotehost).
  - ssh -L 20000:kernel.risc.uni-linz.ac.at:143 bullfinch.risc.uni-linz.ac.at

How to make SSH tunnel in Linux

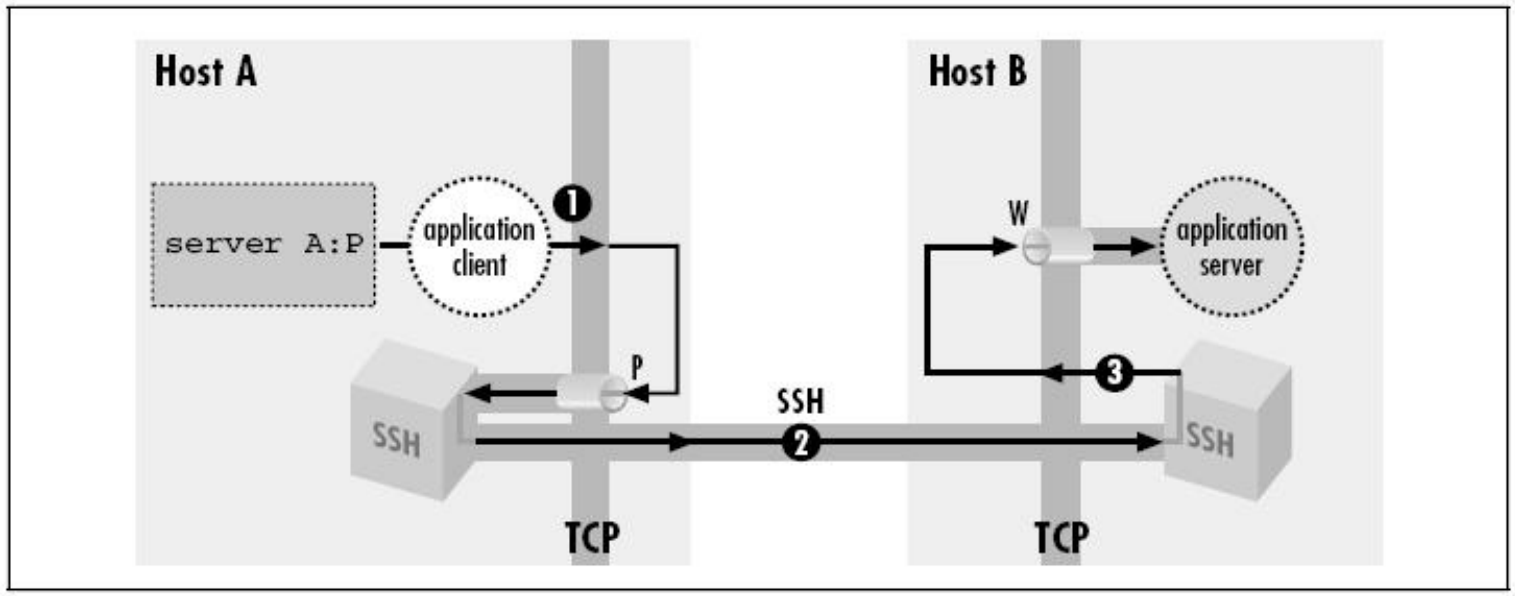
full version

- ssh -f -N -L localport:host:hostport sshd-server-computer
  - B: local computer, C: host, A: sshd-server-computer
- -N do not execure command (-N is for portforwarding)
- -f go into background
- you can use more -L option in one command, (create more tunnels!)

SSH Tunnel - Port forwarding

Window SSH client from www.ssh.com

Port forwarding



SSH Tunnel - Port forwarding

examples for more tunnels

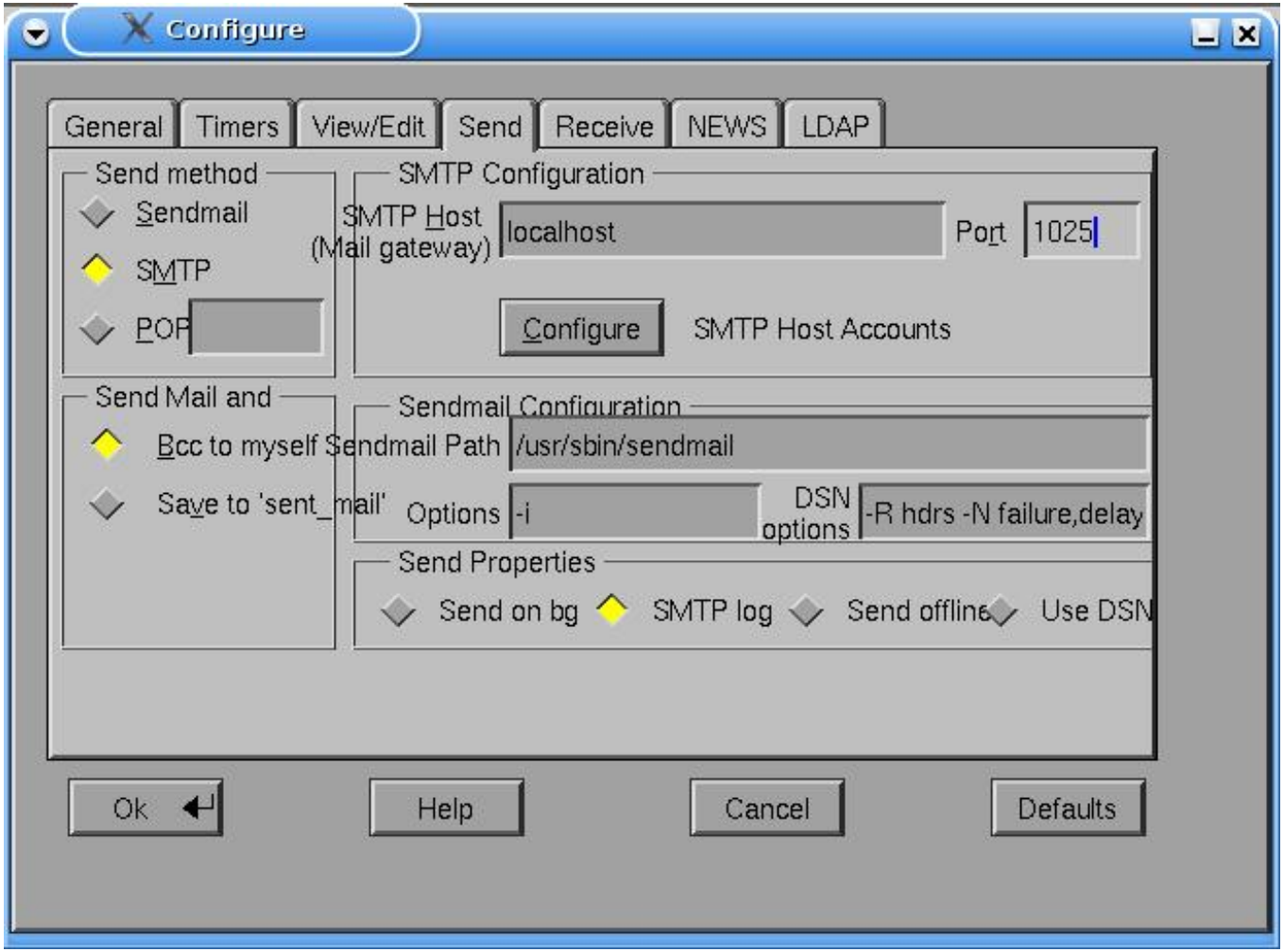
Tunnels

- shell alias: homer

```
uhu:~> which homer
homer:    aliased to
        ssh -f -N
          -L 1025:homer.risc.uni-linz.ac.at:25
          -L 3389:crutch.risc.uni-linz.ac.at:3389
        homer.risc.uni-linz.ac.at
uhu:~>
```

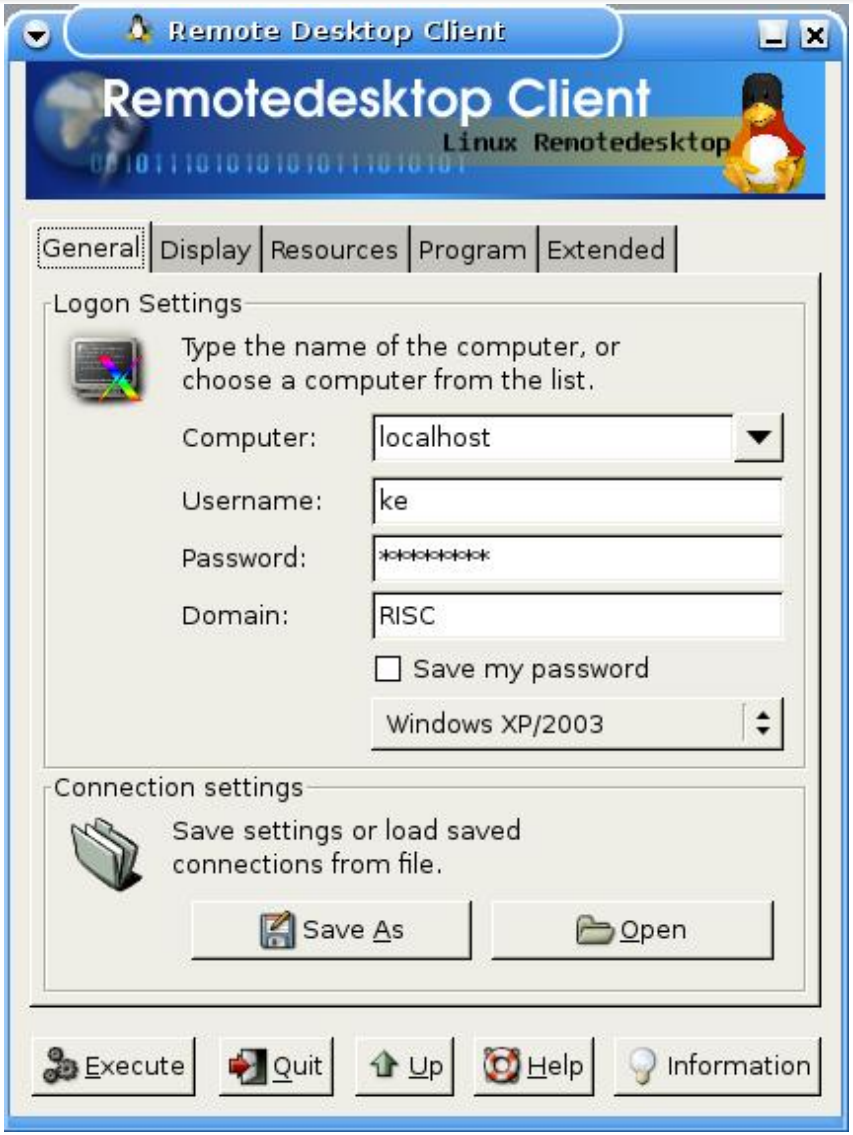
SSH Tunnel - Port forwarding

examples for more tunnels



SSH Tunnel - Port forwarding

examples for more tunnels



SSH Tunnel - Port forwarding

examples for more tunnels

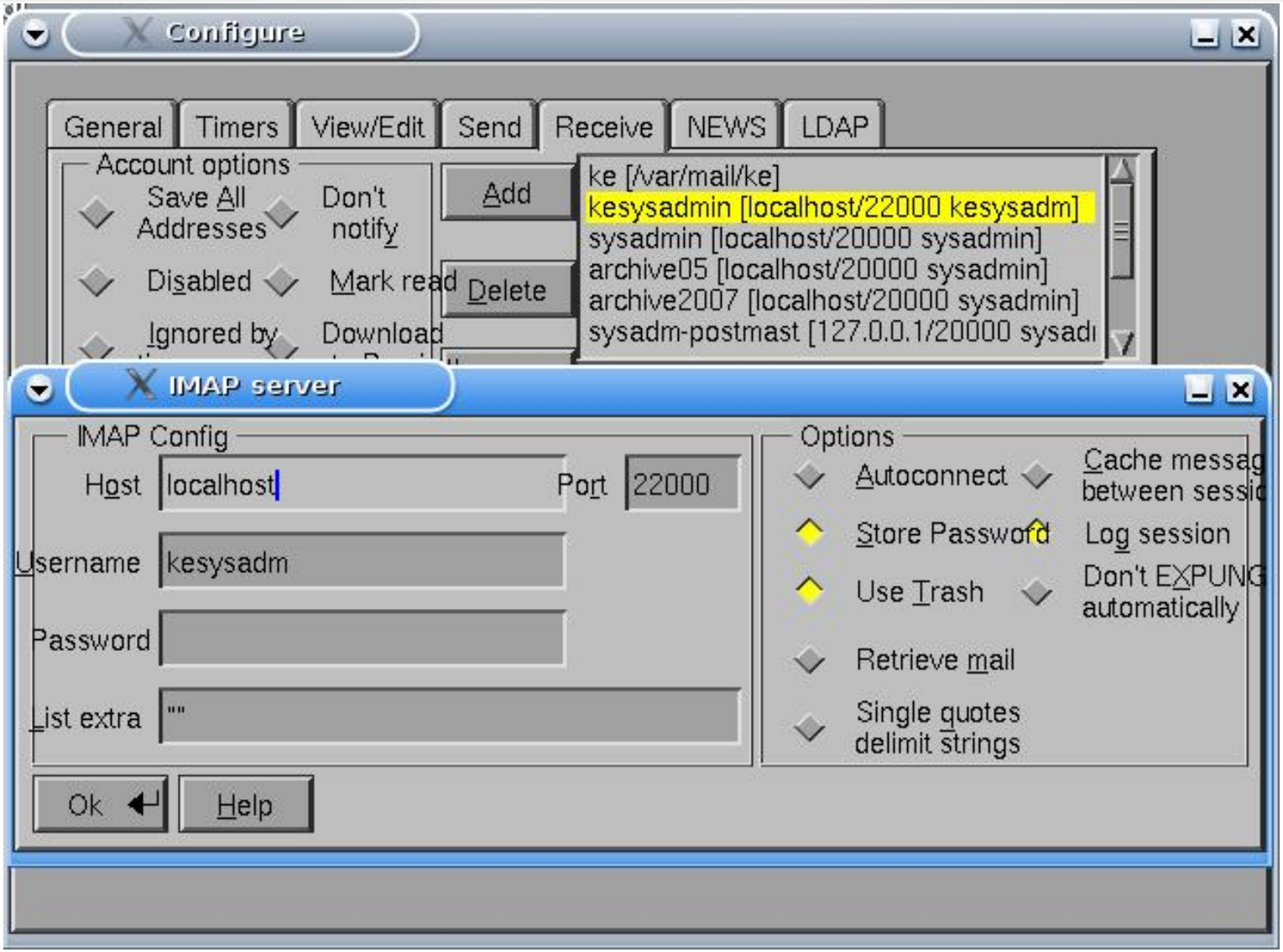
Tunnels

- shell alias: bt (bullfinch tunnel)

```
uhu:~> which bt
bt:       aliased to
     ssh -f -N
         -L 20000:kernel.risc.uni-linz.ac.at:143
         -L 22000:bullfinch.risc.uni-linz.ac.at:143
         -L 2222:bullfinch.risc.uni-linz.ac.at:995
         -L 4442:localhost:442
     bullfinch.risc.uni-linz.ac.at
```

SSH Tunnel - Port forwarding

examples for more tunnels

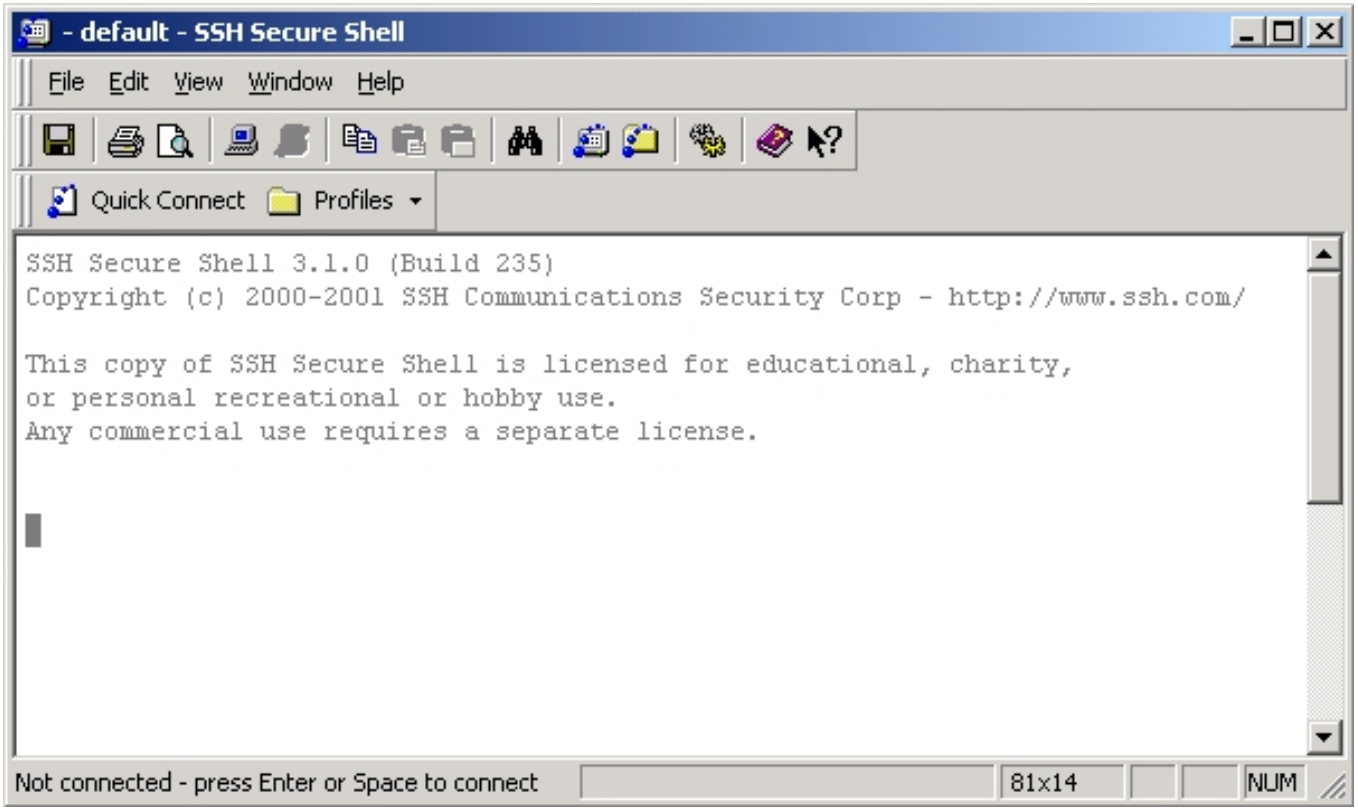


SSH Tunnel - MS Windows

SSH Shell from ssh.com

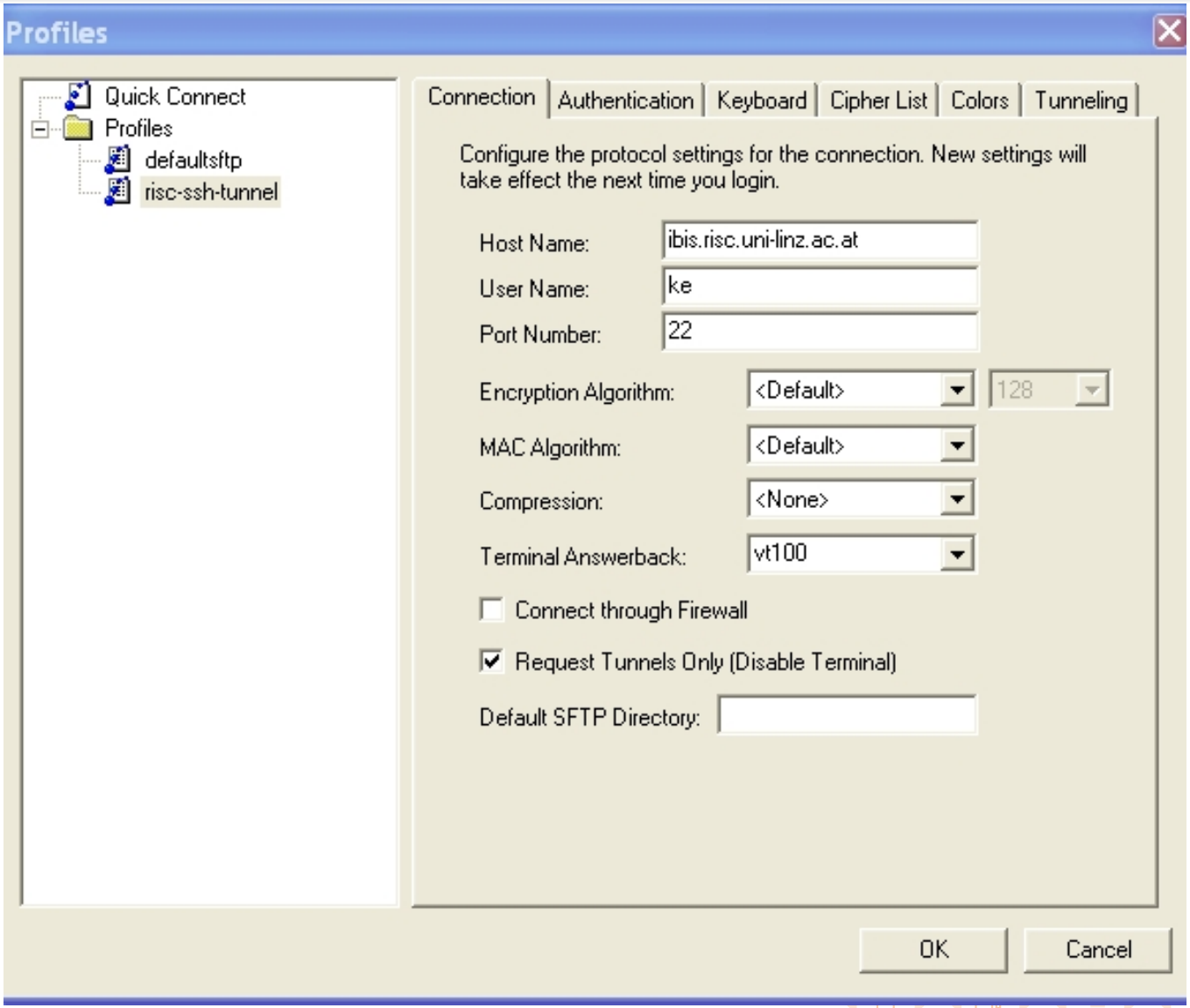
Windows SSH-Client (ssh.com) from TU-Wien:

- ftp://gd.tuwien.ac.at/utils/shells/ssh/SSHSecureShellClient-3.2.9.exe



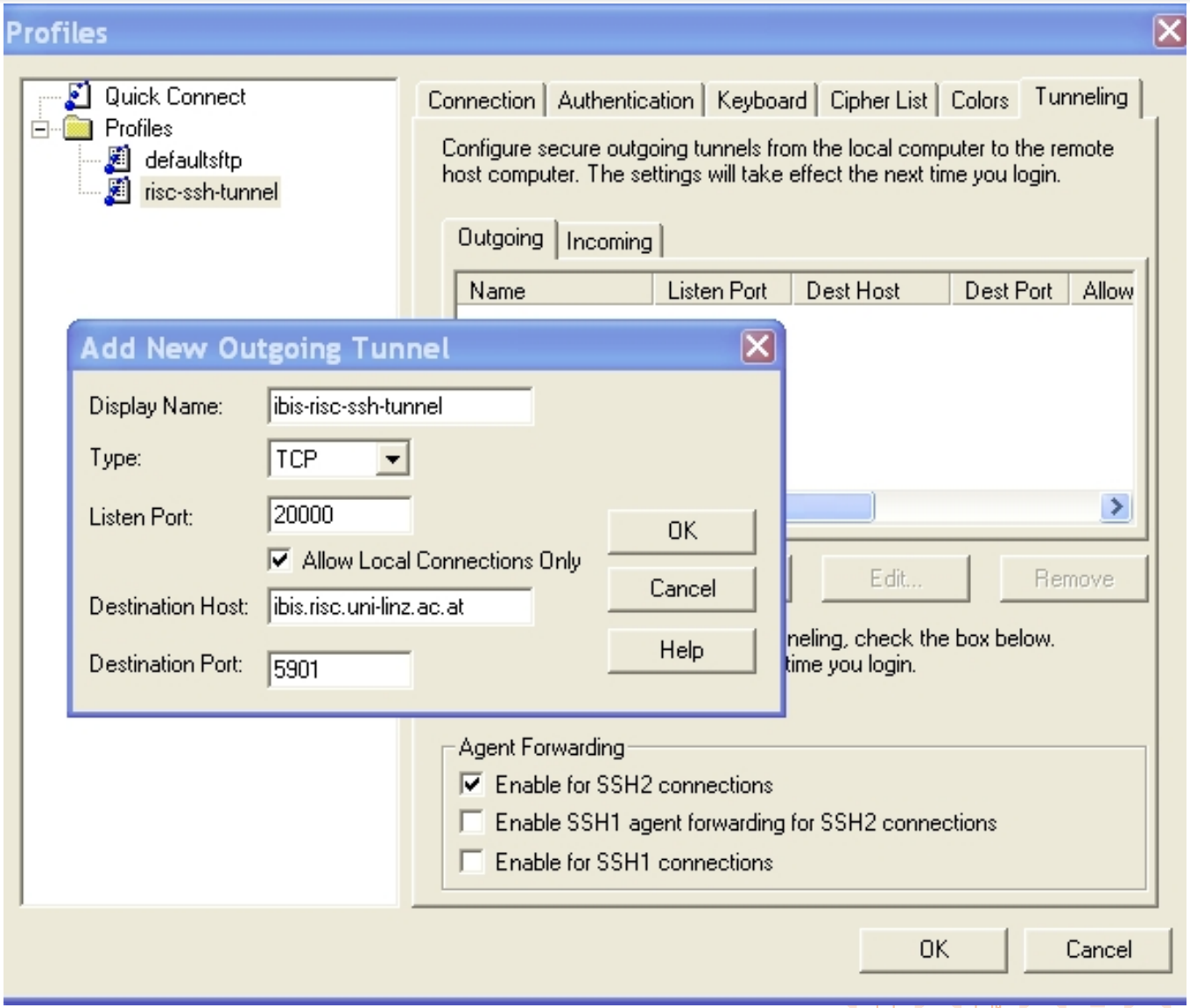
SSH Tunnel - MS Windows

Connection configuration



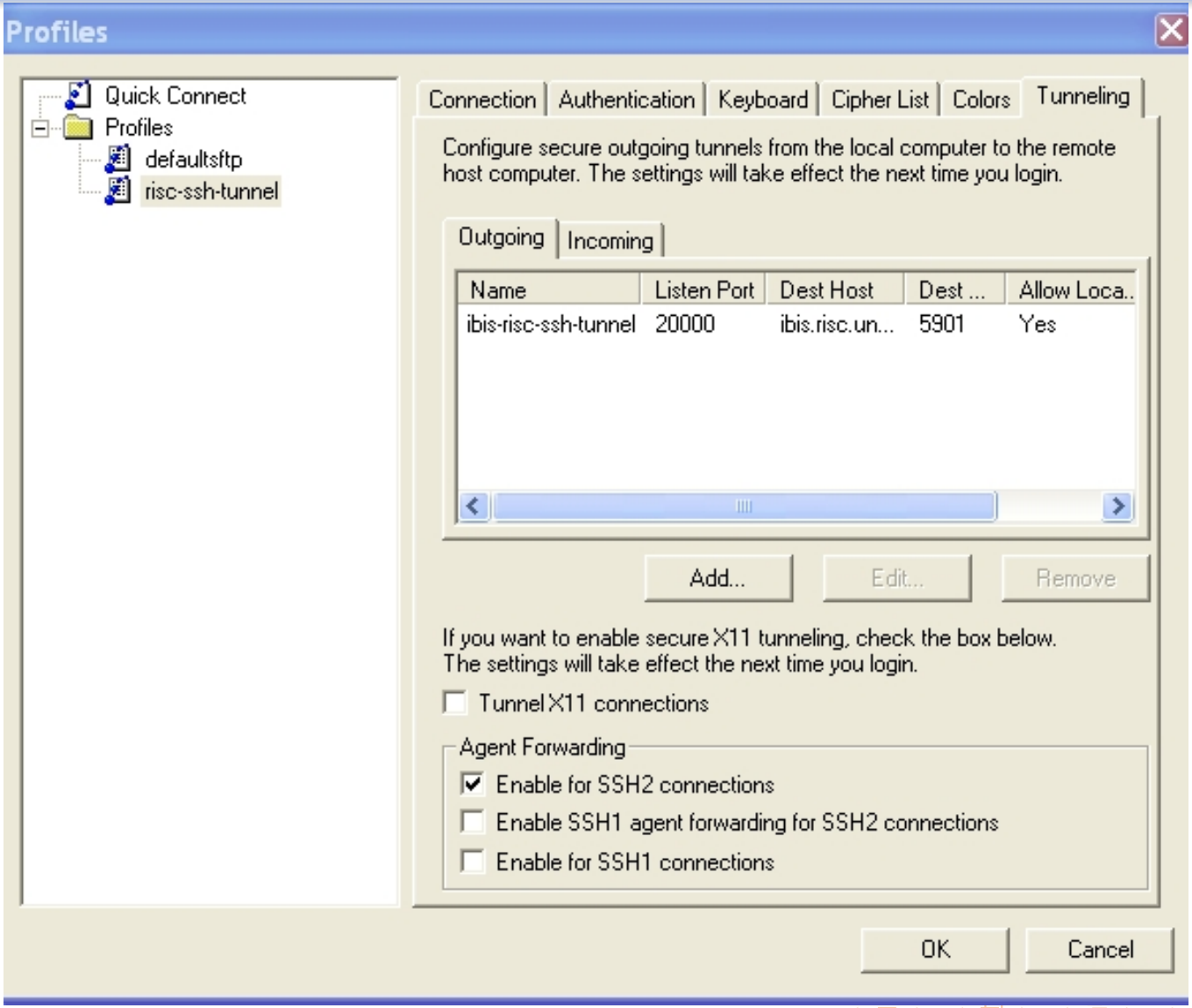
SSH Tunnel - MS Windows

Configuring ports



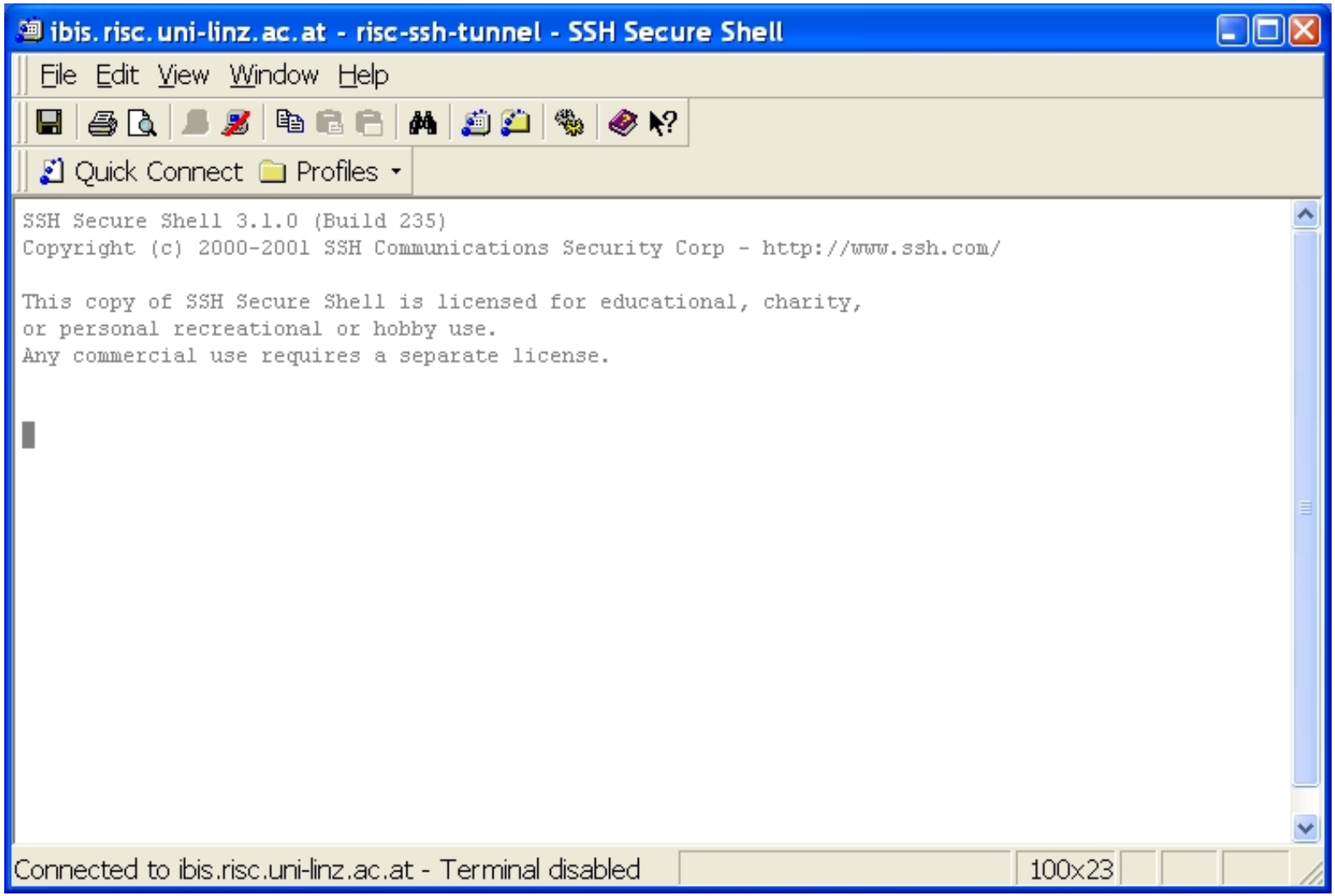
SSH Tunnel - MS Windows

Established tunnel



SSH Tunnel - MS Windows

Established tunnel



Agenda

Remote login without passwd by SSH

How to set up

Basics of the authentication

- SSH authentication methods
  - password authentication; private key authentication
- private key authentication
  - Create a private key - public key pair with ssh; set the passphrase for the private key !
  - Copy the public key to the remote computer
  - Configure the authentication agent: ssh-agent
  - use ssh-add command to add your identity to the `ssh-agent`
- Customizing the authentication
  - installing ssh-askpass
  - Starting `ssh-add` by an icon

Remote login with SSH

Create public key

- Create a public key: ssh-keygen -t dsa
  - always USE a passphrase
  - without passphrase: if your private key is stolen your identity is stolen
  - choose it different from your password, choose a long one
  - it must as save as your password, it can be more save (less restriction)

```
bienenfresser:~> ssh-keygen -t dsa
Generating public/private dsa key pair.
Enter file in which to save the key (/home/ke/.ssh/id_dsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/ke/.ssh/id_dsa.
Your public key has been saved in /home/ke/.ssh/id_dsa.pub.
The key fingerprint is:
a8:00:0e:39:b9:5e:30:a0:c7:70:cd ke@bienenfresser
bienenfresser:~>
```

Remote login with SSH

Copy public key

- copy the public key to the RISC computer
- add to .ssh/authorized_keys file

```
bienenfresser:~> cat .ssh/id_dsa.pub |
   ssh goose.risc.uni-linz.ac.at 'cat - >>.ssh/authorized_keys'

   ke@goose.risc.uni-linz.ac.at's password:
bienenfresser:~>
```

- you will be asked for your password on the remote computer
- check that it works:
  - ssh -X goose.risc.uni-linz.ac.at
  - passphrase will be asked for

Remote login with SSH

ssh-agent

- Authentication agent, ssh-agent
  - saves the identity value (private key) in the memory
  - supports authentication requests from SSH
  - started by login in KDE, GNOME

ssh-add

- transfers the identification (.ssh/id_dsa) to ssh-agent
- asks for the passphrase, to decrypt the private key

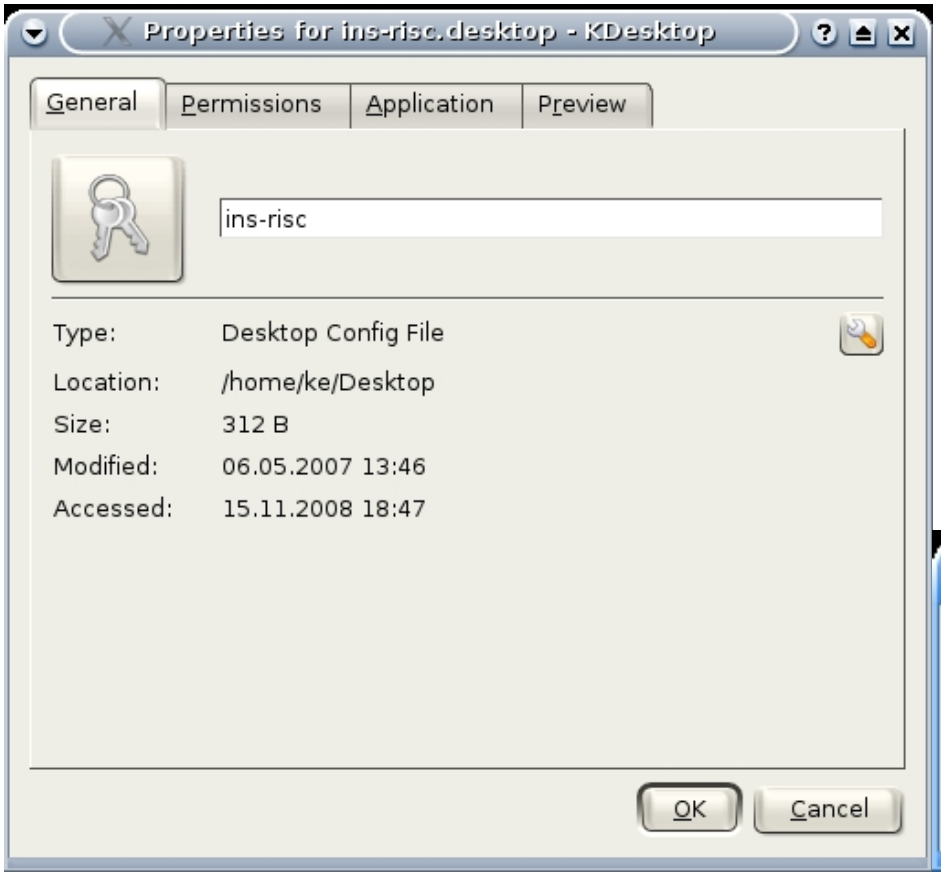
```
bienenfresser:~> ssh-add .ssh/id_dsa
Enter passphrase for .ssh/id_dsa:
Identity added: .ssh/id_dsa (.ssh/id_dsa)
bienenfresser:~>
```

- will invoke ssh-askpass, if get a zero in standard input

Customizing ssh-add

Create a small script in i.e. /usr/local/bin/ or ~/bin

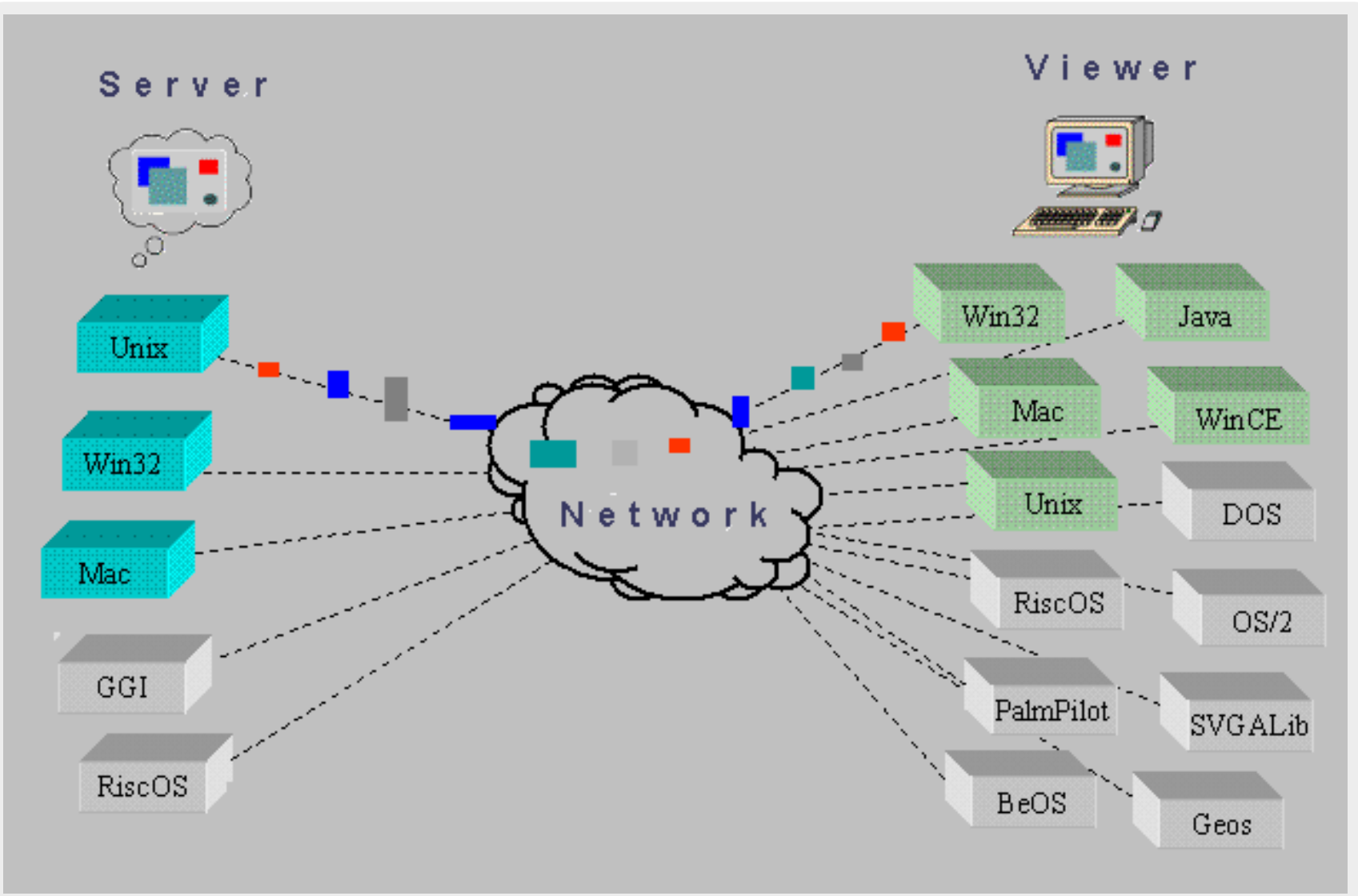
```
#!/bin/csh
cat /dev/null | ssh-add .ssh/id_dsa
```



Agenda

VNC (Virtual Network Computing)

referred as Virtual Network Console, too



VNC - Virtual Network Computing

Basic Features

VNC is a free platform-independent application

- is a Client-Server architecture based on the RFB protocol
- is a graphical desktop sharing system
  - without the need of X on the client side
- transmits the keyboard and mouse events from one computer to another
- relays the graphical screen updates back in the other direction
- is not a secure protocol
  - passwords are not sent in plain-text
  - crack could be successful if both the encryption key and encoded password are sniffed from a network
- always use VNC through an SSH tunnel !
- Open source tool: `http://www.realvnc.com`

VNC - Virtual Network Computing

Basic terminology

Framebuffer (FB)

- is a video output device that drives a video display from a memory buffer containing a complete frame of data
- the information in the buffer consists of color values for every pixel on the screen
- total memory required for the FB depends on the resolution, and on the color depth
- a FB device driver was created for X11: XF86 FBDev as standard part of XFree86
- FBDev is basic driver in X, without using the features of the GPU

VNC - Virtual Network Computing

RFB Protocol

Remote Framebuffer (RFB) protocol

- is a simple protocol for remote access to graphical user interfaces
- it works at the framebuffer level, it is applicable to all windowing systems and applications, including X11, Windows and Macintosh.
- to the basic features a lot of extensions added
  - file transfers
  - more sophisticated compression
  - security techniques
- seamless cross-compatibility
  - between the many different VNC client and server implementations
- clients and servers negotiate using
  - the best RFB version
  - most appropriate compression and security options

RealVNC, Ltd.

- continues development of VNC and to maintain the RFB protocol

VNC - Virtual Network Computing

VNC Server

VNC Server features

- does not have a physical display! (does not bind to a display)
- consists of two servers on Linux/Unix OS
  - Framebuffer Server: to communicate remotely with the VNC client
  - X Server: to communicate locally with the X-clients, presenting itself as a real X-Server
  - the X-server part fills up the framebuffer with the output from the X-clients
  - the FB-server part transfers the content of the FB to VNC-client(s)
- the session information will be kept in the server side
  - if you disconnect from the VNC server it will not close the session
  - Disconnecting from VNC server behaves like locking the session and switching off the monitor
- you have explicitly kill the VNC server after your work !

VNC - Virtual Network Computing

VNC Server II

VNC Server features

- by default uses TCP ports 5900 through 5906
  - each port corresponds to a separate screen (:0 to :6)
- uses ports 5800 through 5806 for java connections
  - allowing clients to interact through a Java-enabled web browser
- Xvnc is the Unix VNC server, it is based on standard X server
- any number of Xvnc server can be started (resources!)
- more clients can connect to the same server
- VNC need more/high bandwidth because of tranferring screenshots
  - the session and switching off the monitor

VNC - Virtual Network Computing

Starting the VNC Server

Starting the VNC server

- log in by ssh to a RISC computer, e.g. gepard:
  - ssh -l username gepard.risc.uni-linz.ac.at
  - uhu> ssh -l guestuser gepard.risc.uni-linz.ac.at
- start the VNC server by the command:
  - gepard:1> vncserver -geometry 1024x768 -depth 24
- You will see something similar in the screen (it just ask a session password at the first run):

```
You will require a password to access your desktops.

Password:
Verify:
New 'X' desktop is gepard:1

Starting applications specified in /etc/X11/Xsession
Log file is /home/yourusername/.vnc/gepard:1.log
```

VNC - Virtual Network Computing

Starting the VNC server

Starting Server

- You have to memorize the server name and the screen number - after the computer name (in this case it is ":1")
  - The port number will be 5901 (5900+screen number)
- You have to shutdown the VNC server, after you do not need it:
  - gepard:3> vncserver -kill :1

    ```
    Killing Xvnc4 process ID 2693
    gepard:4>
    ```
- The configuration and log data for the VNC server are stored in the
  - /home/<username>/.vnc/ directory
- The VNC server asks for the password at the first time only
- If you forgot the password for the VNC server, remove or change it:
  - rm /home/<username>/.vnc/passwd
  - vncpasswd /home/<username>/.vnc/passwd

VNC - Virtual Network Computing

Starting the VNC client

Starting the VNC Client

- create an ssh tunnel on your local computer to the vnc server:

```
     ssh -f -l username -N -L 5901:localhost:5901 srvname
uhu> ssh -f -l guestuser -f -N -L 5901:localhost:5901 gepard.risc.u
```

- start the VNC client on your local computer

```
uhu> xvncviewer localhost:1
```

- best solution is to use a shell alias, e.g. for the tcsh in /home/username/.cshrc :
  - gepardtunnel alias "ssh -f -l username -N -L 5901:localhost:5901 gepard.risc.uni-linz.ac.at "
  - source /home/username/.cshrc
  - activate the tunnel in the command line by gepardtunnel
- Security Risk
  - your password can be stolen using xvncclient without ssh tunnel !
  - hacker get full access to your home directory

VNC - Virtual Network Computing

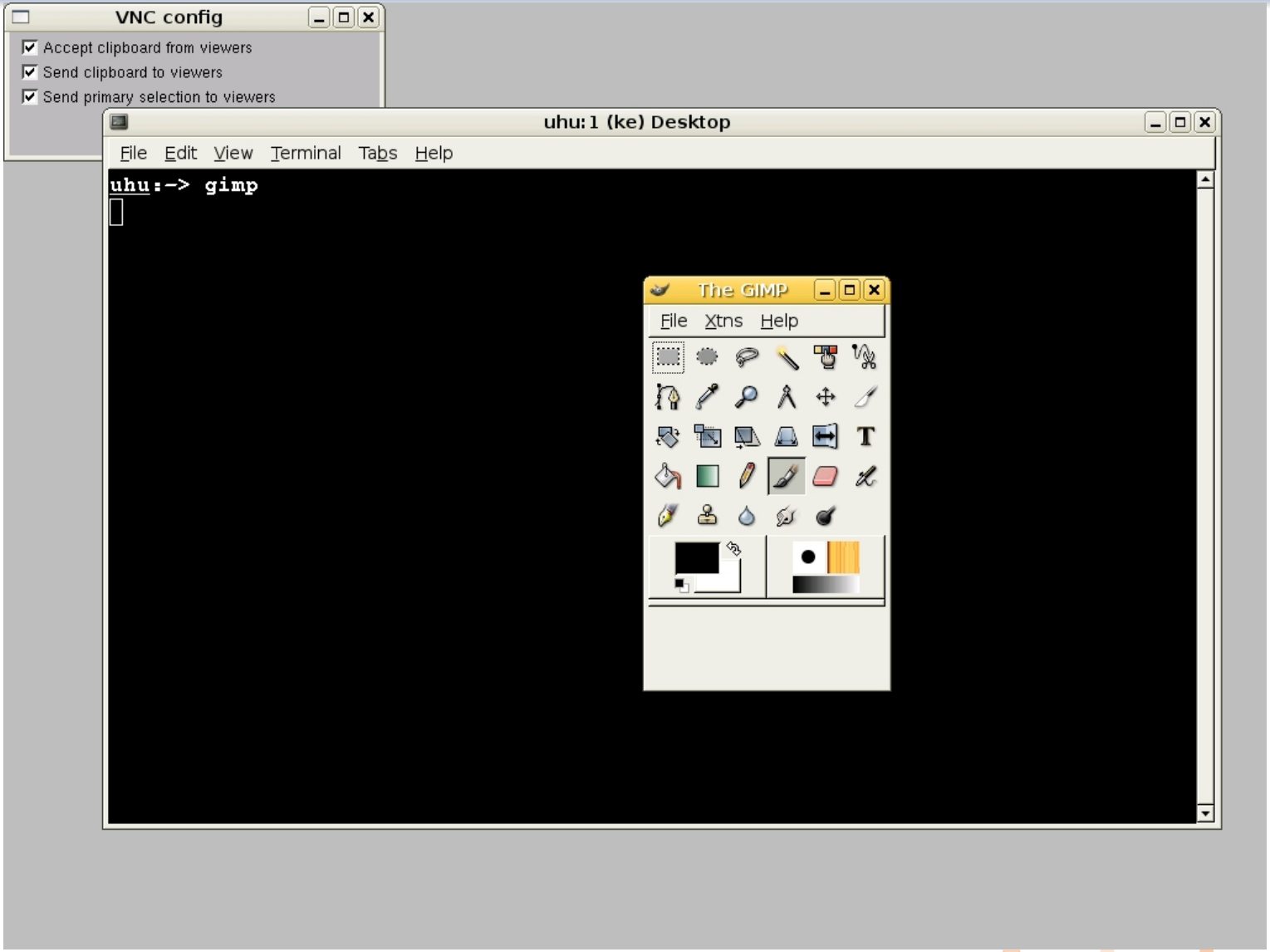
VNC server and client starting

Configuration of the `vncserver` at RISC

- configuration of the vncservers at RISC with option -localhost
  - this means, that the vncserver accepts connections only from localhost (127.0.0.1)
  - with other words: you MUST use ssh tunnel to the host where the vncserver is running (otherwise you'll get error: connection refused).
- example: assumed, you started the vncserver on the computer speedy.risc.uni-linz.ac.at, you need the follwoing ssh-tunnel:
  - `ssh -f -l username -N -L 5901:localhost:5901 speedy.risc.uni-linz.ac.at`
  - localhost will be replaced by 127.0.0.1, and this is the IP from which the vncserver accepts connections.

VNC - Virtual Network Computing

VNC Client - xnvcviewer - standard xterm



VNC - Virtual Network Computing

VNC Server - default xstartup file

xstartup file - simple window manager

```
#!/bin/sh
# Uncomment the following two lines for normal desktop:
# unset SESSION_MANAGER
# exec /etc/X11/xinit/xinitrc

[ -x /etc/vnc/xstartup ] && exec /etc/vnc/xstartup
[ -r $HOME/.Xresources ] && xrdb $HOME/.Xresources
xsetroot -solid grey
vncconfig -iconic &
x-terminal-emulator -geometry 80x24+10+10 -ls -title "$VNCDESKTOP Deskt
x-window-manager &
```

- x-terminal-emulator is a link to /usr/bin/x-terminal-emulator
  - which is a link to /usr/bin/gnome-terminal.wrapper
- x-window-manager
  - is the /usr/bin/metacity, a very simple WM

VNC - Virtual Network Computing

Xvncserver configuration

How to start KDE session

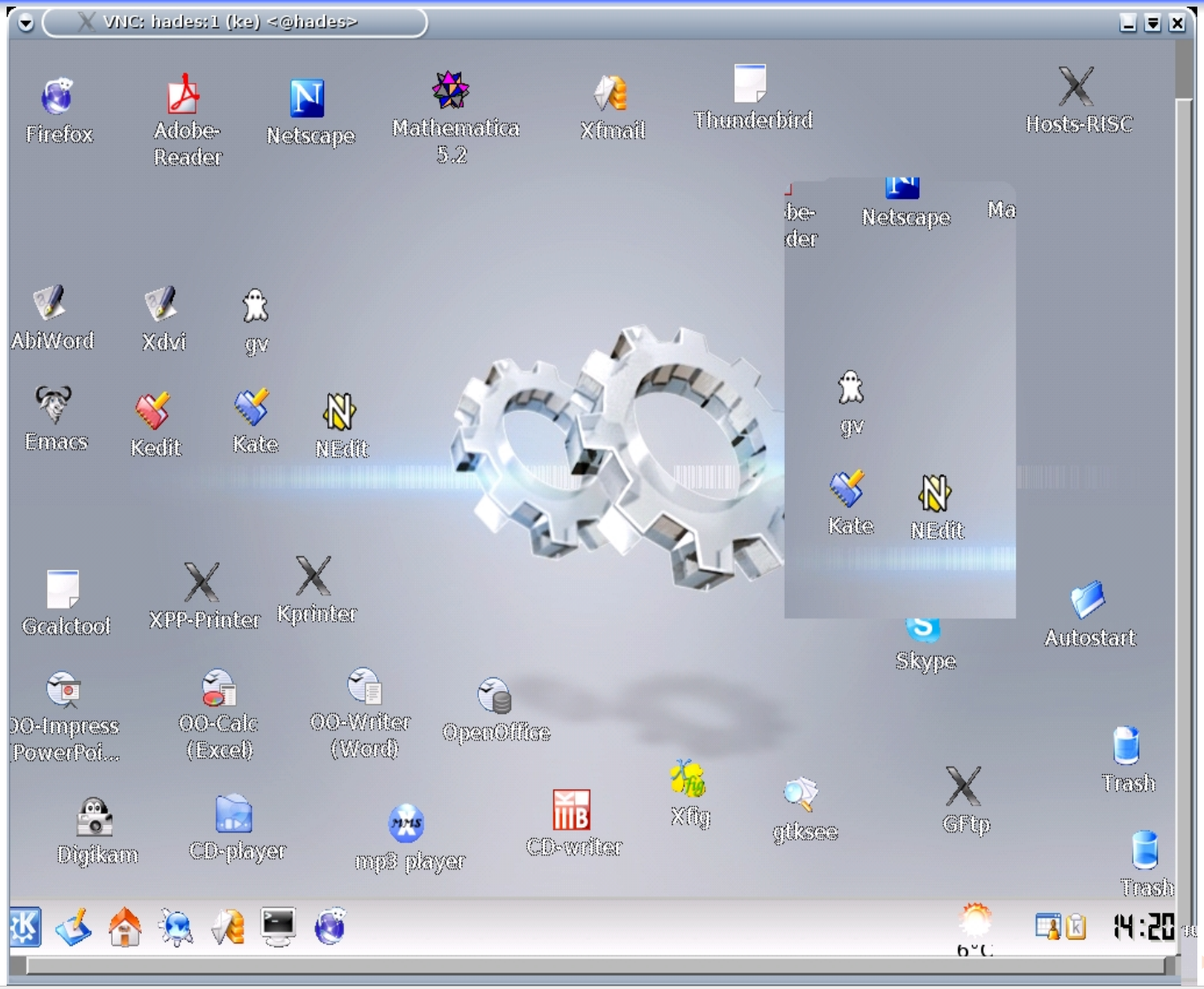
- configuration directory: ~/.vnc
  - password: passwd; session startup: xstartup

```
#!/bin/sh
# Uncomment the following two lines for normal desktop:
# unset SESSION_MANAGER
# exec /etc/X11/xinit/xinitrc
[ -x /etc/vnc/xstartup ] && exec /etc/vnc/xstartup
[ -r $HOME/.Xresources ] && xrdb $HOME/.Xresources

# xsetroot -solid grey
# vncconfig -iconic &
# x-terminal-emulator -geometry 80x24+10+10 -ls -title "$VNCDES
## metacity window manager will be started:
# x-window-manager &
## to start a kde session uncomment the line below and add star
#  x-session-manager &
startkde &
```

VNC - Virtual Network Computing

VNC Client - xnvcviewer - with KDE



VNC - Virtual Network Computing

VNC Server - xstartup file

simplest form of xstartup to start the metacity WM

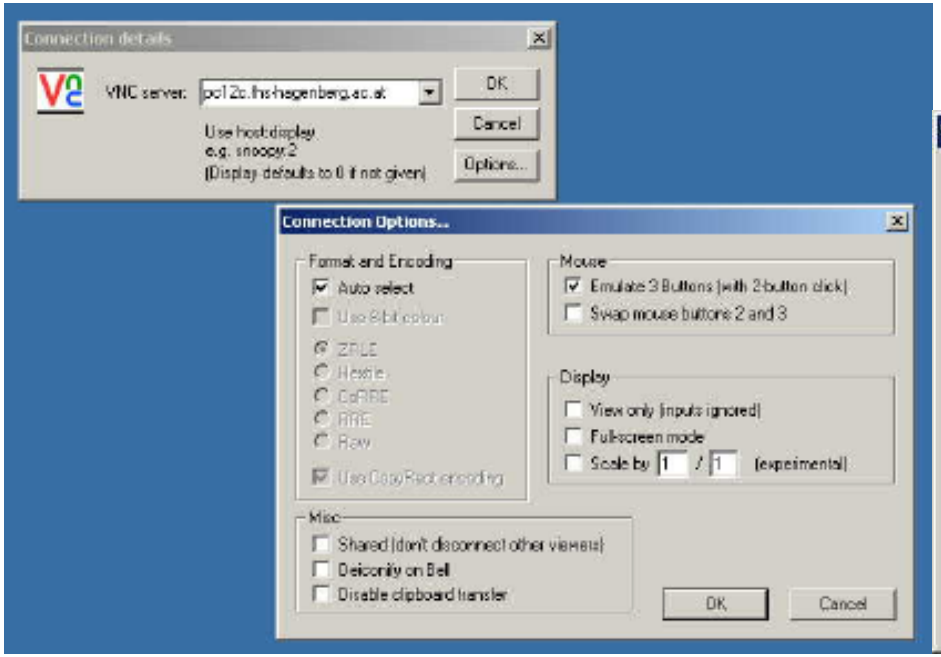
```
#!/bin/sh
xsetroot -solid grey
vncconfig -iconic &
x-window-manager &
xterm &
```

simplest form of xstartup to start KDE

```
#!/bin/sh
exec startkde &
```

Real VNC

Using VNC under MS Windows



Demonstration VNC client

Now make a short demonstration

how VNC works through a tunnel

Agenda

Remote Desktop Protocol

Microsoft Windows

Windows NT/2000: Terminal Services extension

- Remote Desktop Protocol (RDP) developed in the mid 1990's by Microsoft
  - RDP client computer (Windows/Unix) opens a remote desktop session on a Windows NT/2000 server with terminal services extension
  - In client window, user sees another desktop running on the server
  - Introduced by Windows NT Terminal Server Edition
  - Installed at RISC in 1999 for MS Office Compatibility goals
  - The first MS Windows Multiuser OS !
- Windows XP:
  - Provides builtin RDP service functionality
- Windows 2003 Server: successor of NT/2000 Terminal Server Editon

Remote Desktop Protocol

crutch - the RISC Windows 2003 server

crutch: Linux - Windows integration

- Supporting the RISC users for some MS Windows applications
  - for software available only on MS Windows
- Microsoft Software
  - OpenOffice and MS-Office are not fully compatible
  - MS Office is available in the (near) last version on crutch
- Adobe Software
  - Adobe Acrobat 9 Pro Extended (2 concurrent licenses)
  - Adobe Photoshop Lightroom 2.1 (1 concurrent license)
- Other Software
  - ACDSee 8 (image management and manipulation sw)
  - Canon DPP (Digital Photo Professional, for Canon DSLR RAW images)
- Configuration of crutch
  - the riscwide home directory is available (scratch,too)

Remote Desktop Protocol

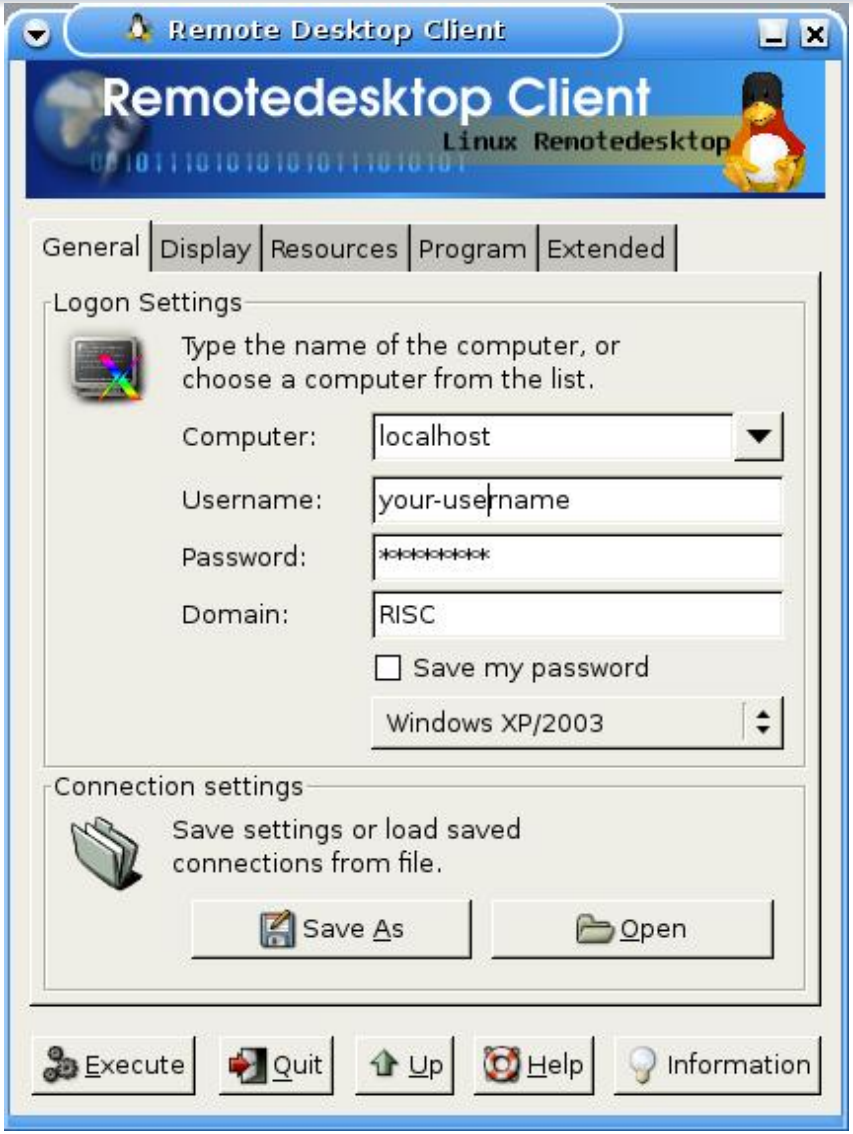
crutch - the RISC Windows 2003 server

RDP ports, connections

- How to connect through an SSH tunnel to crutch
  - RDP uses the port 3389
  - the Windows-2003 server has no SSH server implementation
  - you have to connect to a Linux computer at RISC with SSH and make the tunnel through this computer to crutch
- ssh -l username -f -N -L 3389:crutch.risc.uni-linz.ac.at:3389 gepard.risc.uni-linz.ac.at
  - this is an SSH connection from your computer to gepard
  - the tunnel runs from your computer through gepard to crutch
  - the tunnel section between gepard and crutch is not secure
- Configuration of grdesktop
  - define localhost in the General options for the field Computer

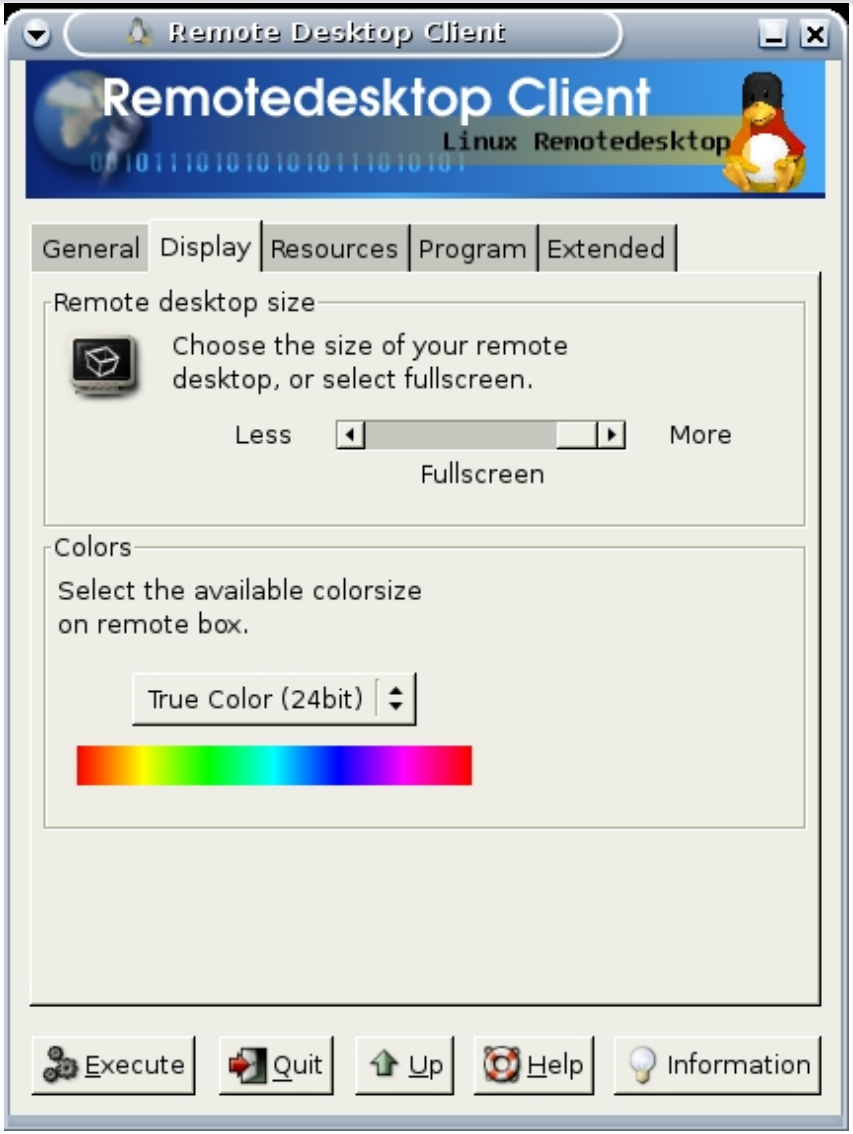
GRDesktop - Configuration

Gnu RDP Client



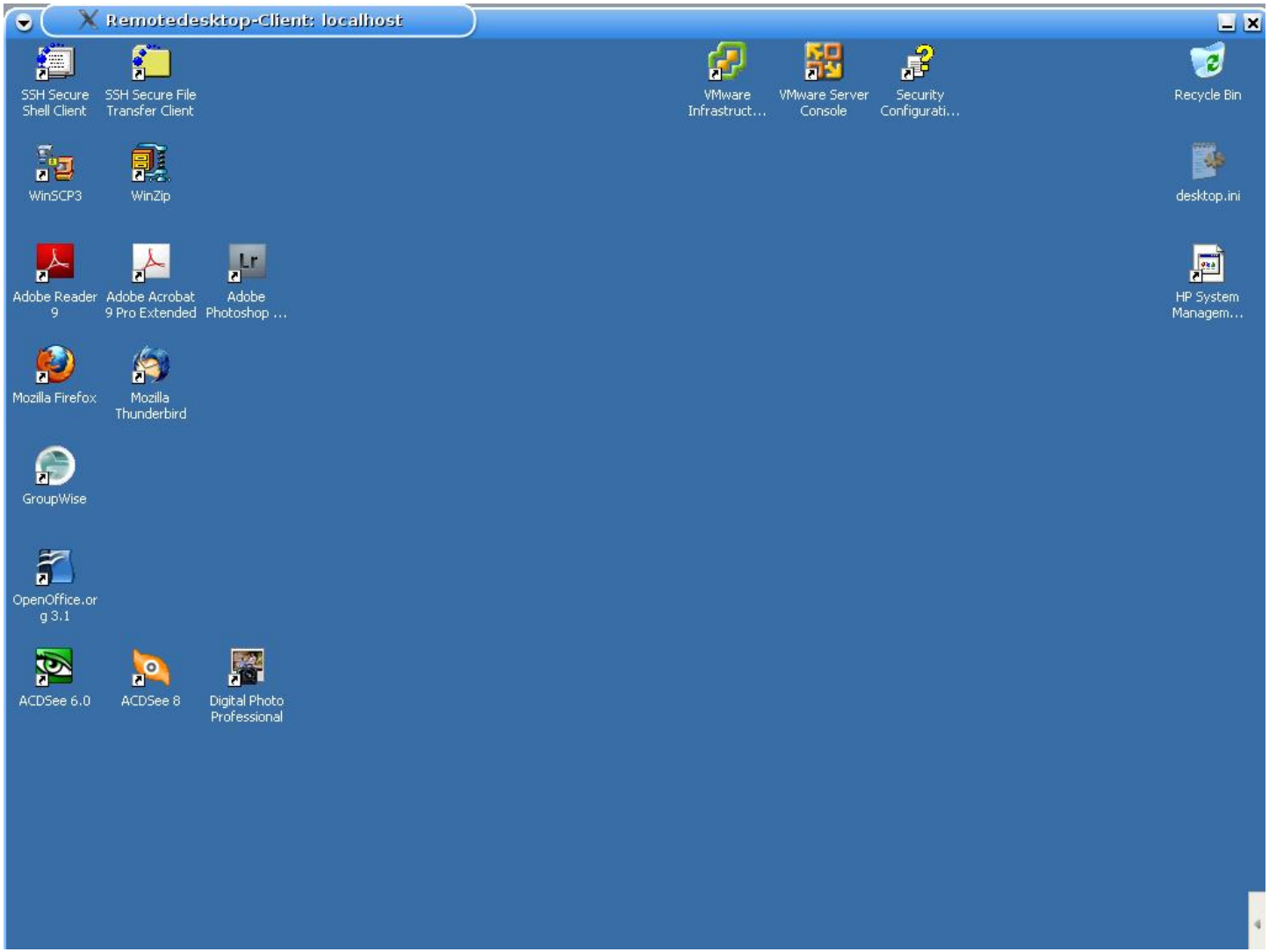
GRDesktop - Configuration

Gnu RDP Client



GRDesktop

Main screen



End of Remote Services, Deskttops

Thanks for your attention !