SSH

Sterrewacht

Most of the desktop machine at the STRW can be accessed through the ssh protocol. So when you know your machine name, use that (including the strw.leidenuniv.nl domain) to access that machine directly.

If you do not have a personal machine you can use the `ssh.strw.leidenuniv.nl` virtual machine to log into our systems and continue from there with an ssh to any of the science servers or cluster machines.

Special access

Some places we visit (e.g. China or Iran) or some hotels abroad limit the internet access to web browsing only. Because you want more in such cases the ssh server of the Sterrewacht now also serves the ssh protocol on web ports 80 and 443. So you can now get access to the Sterrewacht computer systems from those limiting environments using

```
ssh ssh.strw.leidenuniv.nl -p 80 -l <your STRW accountname>
```

With this type of connectivity you can add the tunnelling options (as indicated below) to gain connectivity to a windows remote desktop or your Linux VNC environment.

Instituut Lorentz

For Instituut Lorentz, the server is ssh.lorentz.leidenuniv.nl. Desktops and servers cannot be reached directly from outside, so you will always have to go through the ssh server first. But see also our list of details and tricks

LION

For LION, there is ssh3.physics.leidenuniv.nl.

SSH tunnels

For the Mac and Linux commandline ssh client, setting up a tunnel is usually a matter of using the option
`-L local_port:remote_machine:remote_port`, e.g. `-L 3389:windows machine:3389`

for forwarding a Windows remote desktop. More detail can be found in the vnc ssh tuning pages.

See putty for instructions about setting up a tunnel with `putty` (Windows, linux ssh client).

SSH client software

Linux and macOS come with a commandline client for ssh. For Windows, the recommended client is putty

SSH keys

Create a key pair

To create the most simple key, with the default encryption, open up a console, and enter the following command:

```
$ ssh-keygen -t dsa
Generating public/private dsa key pair.
Enter file in which to save the key (/home/xxxx/.ssh/id_dsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/xxxx/.ssh/id_dsa.
Your public key has been saved in /home/xxxx/.ssh/id_dsa.pub.
The key fingerprint is:
7b:ab:75:32:9e:b6:6c:4b:29:dc:2a:2b:8c:2f:4e:37 xxxx@yyyy
```

When asked for a “passphrase”, we won't enter one. Just press enter twice.

The ssh-keygen program will now generate both your public and your private key. For the sake of this first simple tutorial I will call these files by their default names “identity” and the public key “identity.pub”.

Your keys are stored in the .ssh/ directory in your home directory, but you can store them where ever you'd like.

The file identity contains your private key. YOU SHOULD GUARD THIS KEY WITH YOUR LIFE! This key is used to gain access on systems which have your private key listed in their authorized keys file. I cannot stress this enough, dont have your keys drifting around. Also, make sure your private key always is chmod 600, so other users on the system won't have access to it.

The file identity.pub contains your public key, which can be added to other system's authorized keys files.

Copy public key to server

To be able to log in to remote systems using your pair of keys, you will first have to add your public key on the remote server to the authorized_keys (for version 1) file, and the authorized_keys2 (for version2) file in the .ssh/ directory in your home directory on the remote machine.

In our example we will assume you don't have any keys in the authorized_keys files on the remote server. (Hint: If you do not have a remote shell, you can always use your own useraccount on your local machine as a remote shell (ssh localhost))

First we will upload the public keys to the remote server:

```
$ cd .ssh/
$ scp id_dsa.pub xxxx@zzzz:./id_dsa.pub
id_dsa.pub    100% |*****************************************************|
526       00:00
```

This will place your keys in your home directory on the remote server. After that we will login on the remote server using ssh or telnet the conventional way... with a password.

When you are logged in you should create a .ssh directory, and inside the .ssh/ directory create an authorized_keys and an authorized_keys2 file and add the keys to the files. Make sure the files are not readable for other users/groups. chmod 600 authorized_keys* does the trick.

Placing the key for version 2 works as follows:

```
$ cd .ssh
$ touch authorized_keys2
$ chmod 600 authorized_keys2
$ cat ../id_dsa.pub >> authorized_keys2
$ rm ../id_dsa.pub
```

From now on you can login from client yyyy to server zzzz without having to specify a password.