ssh cheatsheet I

Max-Planck-Institut für Bildungsforschung
Max Planck Institute for Human Development

With the Secure Shell you can connect to a remote machine and transfer files in a truly secure way. It will connect the input of your terminal to the input of a remote site's terminal. An easy extension is the file transfer by reading a file locally and writing it through the secure channel to the remote site. Here is a list of useful commands associated to ssh

start a shell on a remote site:

```
ssh remote-user@remote-host-or-ip
```

run a single command on a remote site:

```
ssh remote-user@remote-host-or-ip "command"
```

setup a tunnel to access resources (i.e. a webserver) originating from the remote site:

```
ssh -fN user@remote -L \
 local_port:resource_hostname:resouce_port
```

Now point your client(browser) to "localhost:local_port"

login and forward the X-Desktop environment to start gui programs from the remote site (this requires an X-Server on your local machine):

```
ssh -f remote-user@remote-host-or-ip \
  "xterm #orany other gui-program"
```

scp (secure copy) can copy from or to a target that may be local or remote:

```
scp <origin> <destination>
```

A target is either a local file/directory (i.e. "." for current directory) or a remote target of the form:

```
user@machine:path/to/folder/
```

copy a single file to a remote site:

```
scp local-file user@remote-site:dest/path/
```

copy a single file to your local site:

```
scp user@remote-site:path/file ./path/newname
```

copy a directory and all its content (recursively) including the target of symbolic links inside of it:

```
scp -r ./dir/ user@remote-site:path/directory/
```

(fast) secure copy when accessing *a lot* of small files:

```
tar cf - ./dir/ | ssh user@remote "tar xf -"
```

ssh cheatsheet II

Max-Planck-Institut für Bildungsforschung
Max Planck Institute for Human Development

In order to use ssh/scp without a password you have to use an ssh-agent, which will manage/cache your ssh identities and therefore needs to decrypt your private key(s) once.

ssh-agent ⟸ *ssh-add* ⟸ *passwordless ssh*

This agent should be running already. You can check with
`echo $SSH_AGENT_PID`

Whenever you use ssh it will test for an agent with this PID and communicate with it (i.e. query the corresponding private key).

You can manually start an agent by issuing `eval `ssh-agent``
This will run something like this in the background and export the agent to all child processes:

```
SSH_AUTH_SOCK=/tmp/ssh-
VAlmm17490/agent.17490; export
SSH_AUTH_SOCK;
SSH_AGENT_PID=17491; export
SSH_AGENT_PID;
echo Agent pid 17491;
```

Now that the agent is running you can add identities (equals a ssh private key) to it. Usually you will only have one identity only, so you have to run this once at the beginning of your work/at the beginning of a new shell. When you encrypted your private key this command will of course ask for a password to decrypt it. In a default environment the usage is simply: `ssh-add`

To connect to a ressource without any interaction you will have to copy your public key to a remote site and add it to the list of authorized identities.

Your public key is located in
`~/.ssh/id_rsa.pub` or
`~/.ssh/id_dsa.pub`

Now append the content of this file to the remote site's identity stash located in:
`~/.ssh/authorized_keys`

i.e.

```
ssh user@remote "cat >> .ssh/
authorized_keys" < .ssh/id_rsa.pub
```

Or on debian systems(grid):

```
ssh-copy-id user@remote
```

Do you have any questions?
Please contact us at:
grid-admin@mpib-berlin.mpg.de
phone: 506