Tools SS 2022

Homeoffice Applications

Florian Beenen

15.06.2022

Contents I

Contents II

Introduction

Objectives

We want to learn...

- how to work on remote infrastructure.
  - Remote connection techniques (VPN, SSH)
  - Applications for working remotely (mstsc, VcXsrv, WSL)
- how can we use URZ services to our advantage.
  - heiBOX, GigaMove
  - Terminal Server
- how to use a selected number of free tools that are extremely useful for everyday work.
  - Use smartphone as webcam.
  - Use smartphone as document scanner.
  - Backup data in distributed fashion on own infrastructure.
  - Split and merge PDFs.

Connecting to Remote Infrastructure

- Scenario: We sit at home and want to use remote services
  - from university
  - from work place
  - from parents' house
- Publicly available services (websites, mail, ...) vs. restricted resources (fileserver, sensitive web pages, unencrypted services, ...)
- Restriction enforced by...
  - Firewall appliance: Blocks traffic based on origin / destination.
  - Application: Server only answers to authenticated clients / request from certain origin addresses.
- When connecting to restricted resources we need to change/move our origin address (IPv4/IPv6) to a trusted zone.
  - Overlay networks with "real" routing (VPNs).
  - Proxies / Jumphosts: Connection over a trusted intermediate host (SSH).

Virtual Private Networks (VPN)

Concept of VPNs

- Use public network (e. g. Internet) as transport network for a “virtual” overlay network.
- Connection is often protected by authentication (e. g. with certificates or username/password) and encrypted.
- VPNs have nothing to do with illegal activity (even though the term is often used in the same sentence with “copyright infringement”).

Configuration Types

Split Tunnel:

- Only certain addresses or networks are redirected through the tunnel (e. g. the university's network).
- Other traffic bypasses the VPN.
- Common scheme to access restricted resources without affecting other traffic.
- Name resolution (DNS) may or may not be redirected.

Time	Type	Domain	Client	Status	Reply
2020-11-30 10:01:51	AAAA	youtube.com	10.17.9.1	OK (forwarded)	IP (5.1ms)
2020-11-30 10:01:51	A	youtube.com	10.17.9.1	OK (forwarded)	IP (5.5ms)

Figure: Admin observing non-work-related DNS queries may not be desired.

Configuration Types (2)

Closed Tunnel (Full redirection):

- All traffic is redirected through the VPN.
- Target IP of the VPN gateway is (obviously) excluded from redirection.
- Benefit from encryption of VPN: Additional security only if local network is untrusted (e. g. unencrypted airport/hotel WiFi) **and** VPN host is trusted!
- VPN gateway sees all traffic that passes through it (may be undesirable or even prohibited in workplace environments).

	Application	L4 Proto	Client		Server
Info	TLS.Facebook ☺	TCP	[redacted]:42285		static.xx.fbcdn.net:https

Figure: Casually surfing Facebook on fully redirected VPN. Your Admin knows...

Bad VPNs (avoid those)

- Numerous commercial VPN providers exist.
- Advertising with increased “security”.
- Create illusion that you are “unprotected” without VPN and that everyone can see what you are doing on the Internet.
- With these VPNs the VPN provider can see everything (... at least to which site you connect). ➥ Shady provider is trusted with sensitive data.

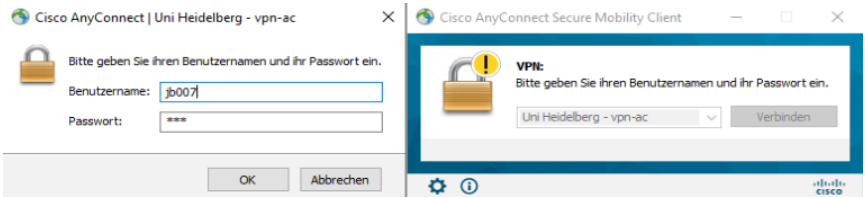
➥ I’d rather give my data to DTAG/Vodafone instead of some VPN company with HQ in Panama...

Useful VPNs

- We want to use VPN to connect to university network.
- Restricted resources
  - Numerous external links from HEIDI (library)
    - IEEE subscription
    - Springer Link
  - KIP-Terminalservers (Linux)
  - SuS Applicationserver (at least for username/password authentication)
  - Fileserver access
- Heidelberg URZ uses Cisco AnyConnect VPN.
- Other universities may use OpenVPN instead.

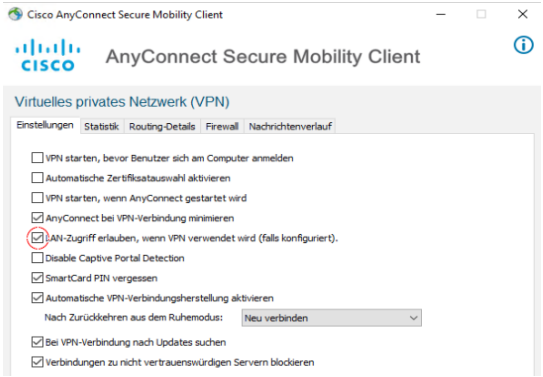
VPN Installation from URZ

- Installation guide here: `https://urz.uni-heidelberg.de/de/vpn`
- Download VPN Client here: `https://vpn-ac.urz.uni-heidelberg.de/`
- Connect to server `vpn-ac.uni-heideberg.de`.
- Log in with Uni-ID and password.
- VPN is set up as **fully redirected** tunnel by default!



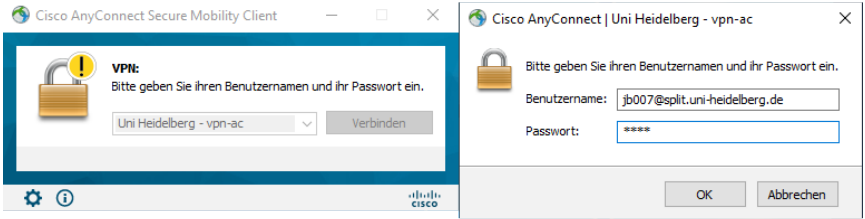
VPN Installation from URZ (2)

- LAN connections are blocked as well (no access to network printers, Spotify speakers, router, ...).
- Can allow LAN connections (only in the same subnet) with configuration option.



Split-Tunnel with URZ

- Split-Tunnel only redirects university network.
  - `129.206.0.0/16`
  - `147.142.0.0/16`
- Uses BelWü DNS (but does not enforce it).
- No access to IEEE and other subscriptions.
- Setup via changed **username**: `jb007@split.uni-heidelberg.de`
- Routing from home network to university network is forbidden (who would possibly do that...).



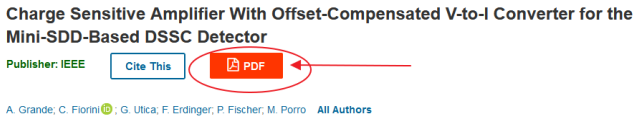
Try It!

Download an IEEE Document

Most IEEE documents are only accessible with a subscription (that you probably do not have – but the university has one (again...)).

Download a "random" paper here: `https://ieeexplore.ieee.org/document/8823042`.

Check that you cannot access the PDF directly with only your “normal” internet connection. Then connect to the VPN. Check again – you should be able to open the PDF now.



Summary VPNs

VPNs are cool because...

- applications do not need to be changed: Destination addresses of remote service remain the same.
- they can redirect your traffic and increase privacy in untrusted local networks.
- site-to-site connections are easy to set up (securely link your house with your parents' network).
- they are easy to use: client applications manipulate the routing tables transparently – the user only has to click on "connect".

VPNs are not so cool because...

- they can redirect your traffic: VPN gateway can see stuff that you may not want it to see.
- VPN client does unwanted stuff: Redirect *all* traffic instead of only the required networks. May filter or inspect DNS.
- from admin's point of view: Often need **very** sophisticated filtering rules in order to prevent bad stuff originating from VPN users in your network.

Secure Shell (SSH)

Introduction to SSH

- SSH is commonly known for “getting a shell on a remote computer”.
- Can do **much** more than just providing a command prompt.
- Features
  - Providing a remote command prompt (obviously...).
  - Transferring files (with sftp subsystem).
  - Displaying remote windows (with local X-Server).
  - Relaying/forwarding TCP ports.
  - Running SOCKS proxy.
- Traffic is encrypted and uses only one single port (unlike FTP).
- SSH is extremely useful for automation / writing scripts ☺.

```
1 florian@Florian-Laptop:~$ ssh door.ziti.uni-heidelberg.de "ip ad | grep glo"
2     inet 147.142.39.240/24 brd 147.142.39.255 scope global dynamic ens192
```

Listing 1: Executing command on remote server

Introduction to SSH (2)

- SSH connections use port 22 by default.
  - May be blocked in a lot of firewall configurations (for “security” reasons...).
  - Often useful to run SSH servers on other ports (2222 for example).
  - Insane amount of unauthorized login attempts when using default port (yes, we got one of our hosts basically DDOS’ed...).
- Different programs
  - Console access: PuTTY / `ssh` command
  - Data transfer: FileZilla / WinSCP / `scp` / `rsync`
- Authentication methods
  - Username/password (okay for occasional usage, bad for scripts)
  - Cryptographic key (increased productivity, enables automated logins)

Secure Shell (SSH)

—

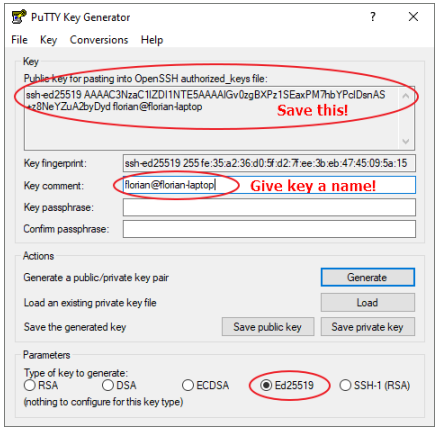
Public Key Authentication

Public Key Authentication

- Generate **pair** of cryptographic keys
  - Private Key: Belongs to you – **never** share this key with anyone. Do not upload to OneDrive, etc...
  - Public Key: Also belongs to you ☺ – this key is shared with anyone that wants to communicate with you using encryption.
  - Both keys are "inverse" to each other: Data encrypted with public key can **only** be decrypted with private key (and vice versa).
  - Try to use ED25519 key (elliptic curve crypto) or RSA with large key size (at least 2048 bit).
- Put public key on SSH server in special directory.
- When authenticating with server your client tries to do crypto with the server using your keys. If the server knows your public key, it will let you in.

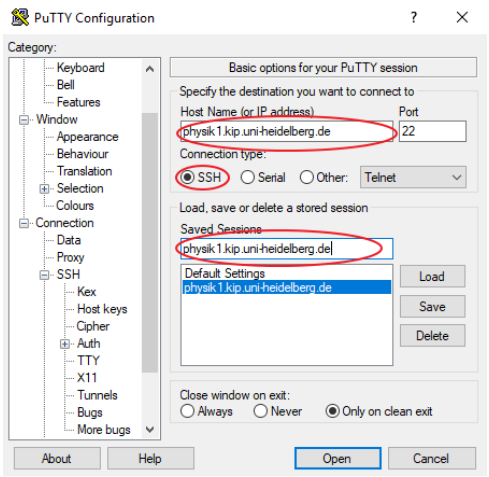
Generating Keypair (Windows)

- For Windows we use PuTTY: `https://the.earth.li/~sgtatham/putty/latest/w64/putty.exe`.
- Small nice program without ads.
- Need PuTTYGen (once) for key generation: `https://the.earth.li/~sgtatham/putty/latest/w64/puttygen.exe`
- Generate the keys
- Save the keys (**NOT** on your OneDrive...)



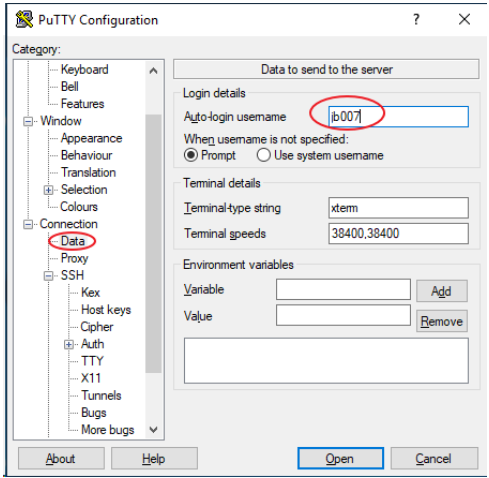
Create PuTTY Profile

- Start PuTTY
- Insert hostname of test server: `physik1.kip.uni-heidelberg.de`, Port 22



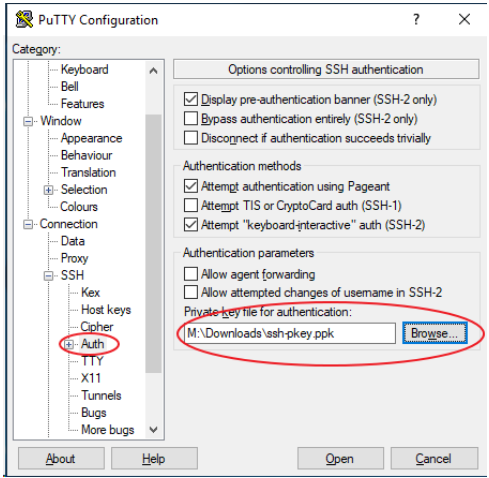
Create PuTTY Profile (2)

- Move to *Connection* → *Data*
- Insert your Uni-ID as *auto-login username*



Create PuTTY Profile (3)

- Move to *Connection* → *SSH* → *Auth*
- Select path to your generated SSH private key.
- Move back to *Session*.
- Save the profile under a name!
- Open the connection.



Key Management with Linux

This is meant for Unix users that do not want to use PuTTY:

- Generate key with `ssh-keygen -t ed25519`.
- Can create key without passphrase.
- Keys will be located at `~/.ssh/id_ed25519` and `~/.ssh/id_ed25519.pub`.
- Get the public key (`cat ~/.ssh/id_ed25519.pub`) – we need it for the next step.

Connect to SSH servers:

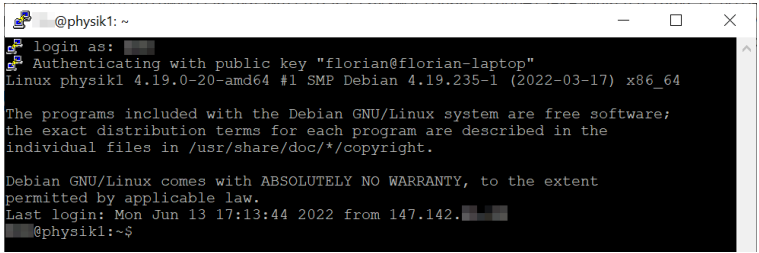
- Use the `ssh` command!
- Connect to testserver via `ssh jb007@physik1.kip.uni-heidelberg.de`.

Store the Public Key on the Server

- When connecting you get warnings.
  - That the server key is unknown: Your system does not know the server because you are connecting for the first time. You can click *yes*. Do **not** ignore this warning under normal circumstances!
  - The server refused our key. This is normal since the server does not known about our key yet.
- Authenticate with your password.
- Get the string with the public key that you saved in PuTTYGen on Slide 22.
- Go into the `~/.ssh` directory – create it if it does not exist (`cd ~/.ssh` or `mkdir ~/.ssh`).
- Open the file `authorized_keys` with an editor (e. g. with `nano`) (`nano ~/.ssh/authorized_keys`).
- Paste your public key from the saved string from Slide 22 into the file. If there is already stuff in the file, append your key in a new line at the end.
- Save the file (`Ctrl` + `X` → `Y` → `Enter`).

Store the Public Key on the Server (2)

- Ensure that the permissions on the `~/.ssh` and all the subordinate files are okay. SSH server does not like it if the permissions are "too open" (should be done already – please don't break things ☺).
  - `chmod 700 ~/.ssh`
  - `chmod 600 ~/.ssh/authorized_keys`
- Terminate the session by typing `exit`.
- Try to reconnect with your PuTTY profile. This time the server should accept your key.



Exercise: Login with Key

Login to SSH System

- Generate a key and store it to the `authorized_keys` file as explained before.
- Try to log in to another KIP-Server (`physik[12345].kip.uni-heidelberg.de`)
- You should be able to log in without being asked for a password.

Note: If you have an "old" Uni-ID this may not work. To find out if your Uni-ID is "old" execute `df | grep jb007` (with your Uni-ID instead of jb007). If something with `//heivol-u.ad.uni-heidelberg.de/` shows up, your Uni-ID is "old". If this is the case, you need to open a SSH session with password authentication first; then you can open another SSH session which uses the SSH key. This is a Kerberos matter and not a bug.

Secure Shell (SSH)

—

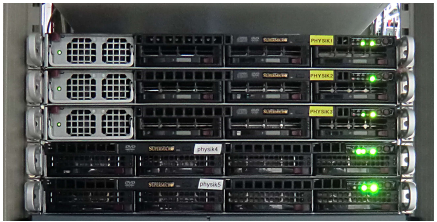
Getting a Remote Shell

Getting a Remote Shell

- Most basic functionality of SSH: Get a command prompt of a remote server.
- Most prominent for Unix systems – SSH shell is also available for Windows (accessing `cmd.exe` remotely).
- Why should you want to run things remotely:
  - Your computer is not powerful enough: We have big machines at university to run compute-intensive calculations.
  - You do not have the software: License agreements prohibit you from installing e. g. Virtuoso on your private machine.
  - You do not have the data: Your work files are located on remote infrastructure and you do not want to load/store your data over the internet all the time.
- Shell allows you to execute any command on the remote server.
- It is also possible to start graphical applications and "redirect" window output to your local machine.

Logging into KIP Server

- The KIP hosts five different servers for working remotely (`physik1.kip.uni-heidelberg.de`, ..., `physik5.kip.uni-heidelberg.de`).
- Can only log in from university network (either be there locally over Eduroam or use VPN).
- A lot of useful software is installed on the system – we will try some of it.
- Machines are not the *most* powerful (AMD Opteron 8350) – your notebook probably has greater performance.
- I use KIP machines to run Mathematica and to access my files.



Secure Shell (SSH)
—
Port Redirection

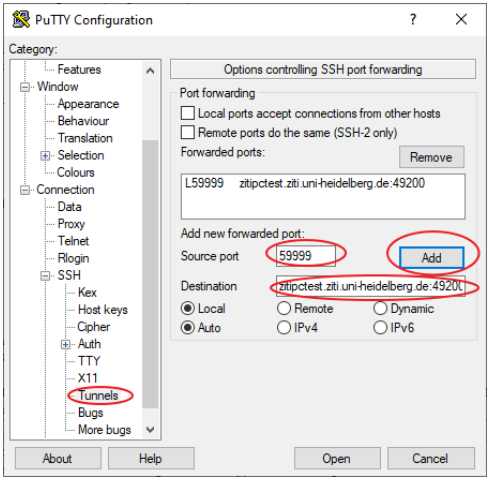
Redirecting TCP Ports

- You can use SSH to redirect ports of remote hosts to you via the SSH gateway.
- SSH gateway acts as proxy: Remote host thinks that the SSH gateway connects.
- Extremely useful for accessing restricted resources!
- Local port is virtually bound to remote port.
- We connect to remote resources by accessing `localhost` with the bound port number.
- Connection is held as long as SSH connection is open (➥ do not close PuTTY ☺).

➥ Firewalls cannot see tunneled traffic and therefore cannot block it. Let the traffic obfuscation begin...

Configuring Port Redirection in PuTTY

- Move to *Connection* → *SSH* → *Tunnels*
- Add an unused random local port (e. g. 59999).
- As destination enter `zitipctest.ziti.uni-heidelberg.de:49200`
  - This is a special host where only a webserver runs.
  - It tells you whether you connect to it in the "right" way ☺.
- Don't forget to **save** the session!



Test your Connection

Login to SSH System

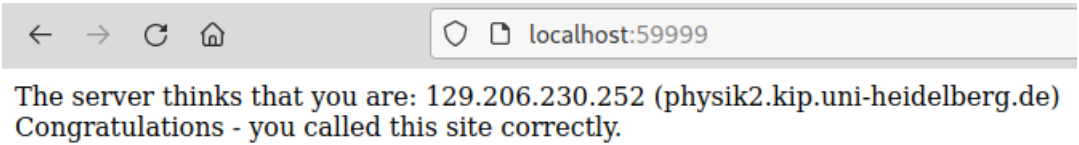
- Open your browser and access `http://zitipctest.ziti.uni-heidelberg.de:49200/`. Ensure that the website says that you are connecting in the "wrong" way.
- Open the SSH connection with the configured tunnel.
- Now try to access `http://localhost:59999` in your browser. Now the website thinks that you are connecting from a different host. This should be good.

Pitfalls:

- Ensure that SSH port redirection is not blocked by your local antivir or firewall!
- Do not close the SSH connection when you try to access tunneled ports.
- Did you save your SSH configuration in PuTTY ?!

Test your Connection (2)

The result should look something like this...



➥ If you get errors, triple-check your tunnel-configuration and investigate what software may block stuff on your computer (especially antivir).

Secure Shell (SSH)

—

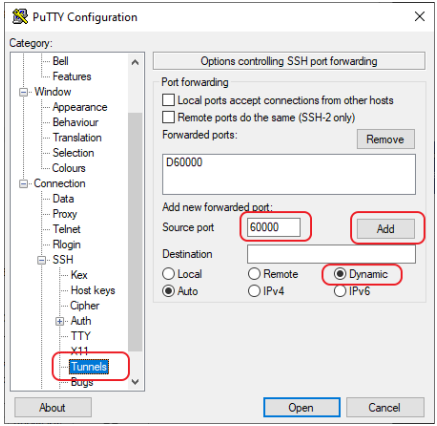
SOCKS Proxy

Running SOCKS Proxy over SSH

- SOCKS Proxy is a possibility to send (nearly) arbitrary data via a proxy server.
- Proxy server: Man-in-the-middle between you and your desired destination (e. g. IEEE webserver).
- SSH supports “dynamic port forwarding”
  - Port redirections that we have previously defined manually can be set up automatically “on-the-fly”.
  - Applications that support SOCKS proxy servers can redirect their traffic through SSH connections.
- Why is this cool?
  - We can redirect traffic easily without using VPN!
  - We only need an SSH client (such as PuTTY) and a proxy-aware application (e. g. Web browser).
  - Does not mess with routing table ➥ Does not need admin privileges ☺.

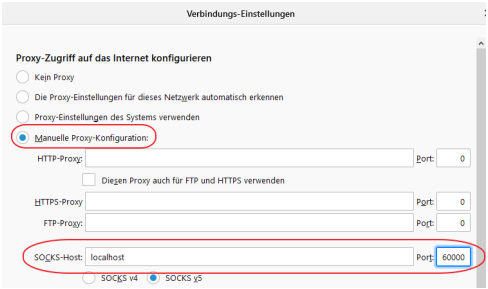
SOCKS Proxy in PuTTY

- Edit your connection profile in PuTTY.
- Move to *Connection* → *SSH* → *Tunnels*.
- Enter a new unused local port (e. g. 60000).
- Click on *Dynamic*.
- Click on *Add*.
- Save your profile !!



SOCKS Proxy in Firefox

- Open the connection in PuTTY and log in.
- Open your browser (e. g. Firefox).
- Configure your browser to use SOCKS Proxy:
  - Open configuration menu on Firefox and scroll down.
  - Enable manual proxy settings and fill in `localhost` and port 60000 under *SOCKS-Host*.



SOCKS Proxy in Firefox (2)

- Open random websites (e. g. `https://www.wieistmeineip.de/`).
- You will find that the source IP is not your own (but the one of the proxy).
- Pitfalls:
  - The proxy server is a man-in-the-middle: It can see much of what you are doing.
  - For some reason connections to IPv6 sites are not possible (I don't understand why – may have something to do with `sshd`).

Secure Shell (SSH)
—
File Transfer

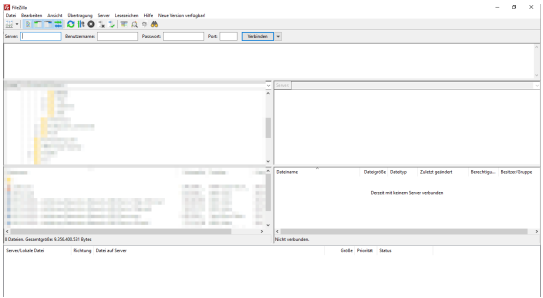
Copying Files over SSH

- SSH has optional subsystems. One of those is *sftp*.
- It behaves like normal FTP (File Transfer Protocol) but better.
  - Uses encryption provided via SSH.
  - Only uses one port (from the SSH connection) instead of the FTP 20/21 frustration from the past ☺.
- Can use programs like FileZilla or WinSCP for transferring files.

SFTP: Setup

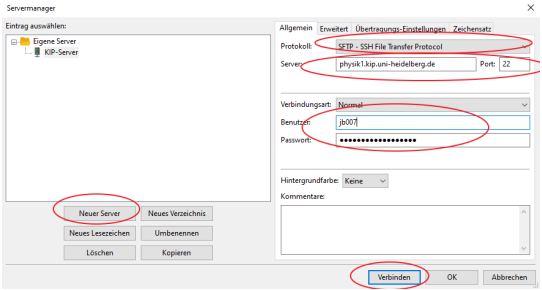
What we need:

- If working from home: Working and connected VPN tunnel (either split or with full redirection).
- Download and install FileZilla: `https://filezilla-project.org/download.php?show_all=1`.
  - Linux users can install via repo: `sudo apt install -y filezilla`.



SFTP: Setup (2)

- Open Server Manager (Leftmost icon directly under the navigation bar tag “File”/“Datei”).
- Create a new server.
- Fill in the following data.
  - Protocol: SFTP
  - Server: `physik1.kip.uni-heidelberg.de` (they have `physik1` ... `physik5`)
  - Port: 22
  - Username: your Uni-ID (e. g. jb007).
  - Password: the password for your Uni-ID.
- Click on “Connect”.
- Ensure that VPN is up and running – otherwise it will not work (when accessing the server from home).



FileZilla Usage

- Left side of the FileZilla window: local directory structure on this computer.
- Right side: Remote directory structure of the currently connected SFTP-server.
- Right-click on one file on either side allows download/upload of the file to the respective other side.
- Transfer works recursively for directories.

➥ When connecting to `physik1.kip.uni-heidelberg.de` you have full access to your private files stored in your account!

Login to SFTP Server

- Connect to `physik1.kip.uni-heidelberg.de` via SFTP (remember to open VPN first).
- Try to upload a small file to the server using FileZilla.
- (Example is a little bit academic, if we are trying this from the CIP-pool)

Rsync

- Transferring large directories via command line is often required.
- Command line tool `rsync` synchronizes directories *very* fast.
  - Only copies files that are different.
  - Can compress data during transfer (very useful when copying e. g. database files or other sparse data objects).
  - Can copy over SSH connection (no extra firewall rules required).
  - Can optionally compare checksums of files.
- Rsync needs additional program `rsync` installed. Is sometimes installed by default but not always.

Rsync – Basic Commands

Command	Purpose
`rsync file1.txt file2.txt`	Copy+Overwrite `file1.txt` to `file2.txt`
`rsync -ave ssh file1.txt user@server.example.org:dir`	Copy local `file1.txt` to the given remote server and put the file in the directory `dir` relative to the home directory of the given user.
`rsync -av dir targetDir`	Copy directory `dir` to directory `targetDir`. Will create a directory named `dir` under `targetDir`.
`rsync -av dir/ targetDir`	Copy contents of directory `dir` to directory `targetDir`. Does not create a subdirectory.
`rsync --delete -av dir targetDir`	Copy directory `dir` to directory `targetDir` and remove everything else in `targetDir` that is not present in `dir`. Good for differential backups.

Rsync – Basic Commands (2)

Command	Purpose
`rsync -avze "ssh -p 2222" user@server.example.org:/opt/dir targetDir`	Copy remote directory `/opt/dir` into local `targetDir` using non-standard SSH port.
`rsync -avP file1.txt file2.txt`	Copy `file1.txt` to `file2.txt` and show the progress while copying.

Secure Copy – SCP

- If `rsync` is not available (for whatever reason), you can still copy files from one machine to another using "Secure Copy" (`scp`).
- Also uses SSH connection to transfer data.
- Has less features than `rsync` but is more likely to be available on arbitrary systems.
- Can also copy full directories (requires `-r` flag)
- Still requires the remote server to allow console access. If server only allows SFTP subsystem (and no shell), `scp` will not work.

```
1 florian@Florian-Laptop:~$ scp test.pdf user@server.example.org:remoteDir
2 test.pdf 100% 1771KB  24.1MB/s   00:00
```

Secure Shell (SSH)
—
X-Window Forwarding

Redirect GUI windows

- Sometimes you want to run applications that open a graphical interface on a remote server.
  - Mathematica on KIP servers (needs license)
  - Virtuoso for full-custom chip design (needs license and large design kit files)
  - Vivado for FPGA design (needs license and a lot of computing power)
- We can run a local **X-Server** on our private machine that displays remote windows (the environment is called “X Window System”...).
- Remote window contents are redirected through SSH connection and displayed on local machine.

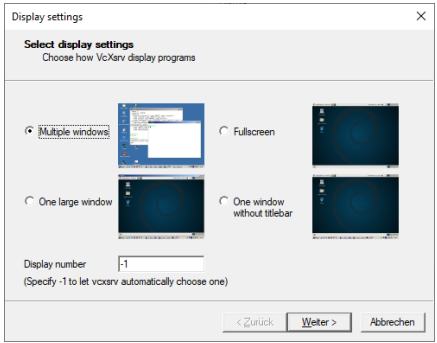
What we need:

- Local X-Server
  - VcXSrv: Easy to deploy – works best (for me).
  - Uni Tübingen has a list (`https://tinyurl.com/y8hvf9hn`) of various X-Servers.
- Special PuTTY config option to allow X-Forwarding.
- Connection to SSH server that allows X-Forwarding.

X-Server Setup

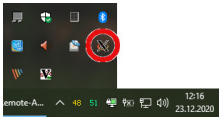
Mostly relevant for Windows users – X-Server should be installed on most Linux machines by default.

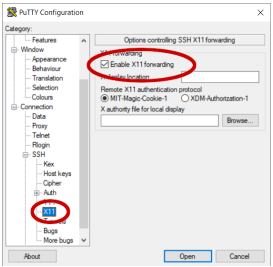
- Download and install VcXSrv: `https://sourceforge.net/projects/vcxsrv/`.
- Alternative: I have extracted VcXsrv files and compressed in ZIP file – no installation required: `https://tinyurl.com/y866bthb`.
- Start the `xlaunch.exe` in the VcXsrv directory.
- You can click "Next" through all the steps.
- A firewall warning may occur on the last step. I am reasonably sure that you can dismiss it and it should still work.



X-Server Setup (2)

- After starting X-Server an icon should be visible in your taskbar:
- Now, we need to alter the PuTTY profile for `physik1.kip.uni-heidelberg.de`
  - Go to *Connection* → *SSH* → X11.
  - Enable the X11-Forwarding – leave the remaining settings untouched.
- Don't forget to **save** the profile!





X-Forwarding with Unix

- X-Server should be installed by default on Linux.
- Needs some special packet on Mac (XQuartz?).
- Allow X-Forwarding via `-X` flag with the `ssh` command

```
1 florian@Florian-Laptop:~$ ssh -X jb007@physik1.kip.uni-heidelberg.de
2 jb007@physik1.kip.uni-heidelberg.de's password:
```

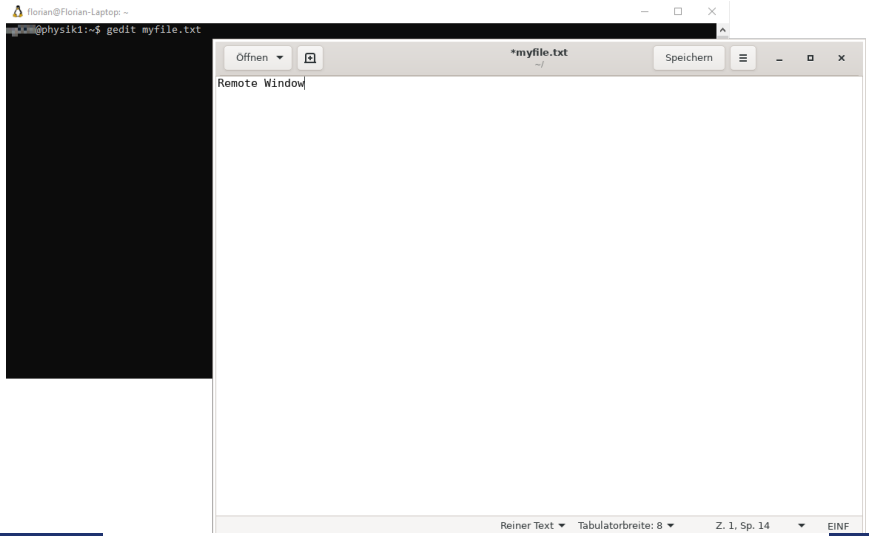
Test the X-Forwarding

- Now we can open the connection (ensure that you are connected to VPN again, if accessing from home) and log in.
- Graphical applications should open on your machine now.

Edit a File

- Connect to `physik1.kip.uni-heidelberg.de` with X11-Forwarding enabled.
- Maybe use personal notebook instead of CIP-pool PCs.
- Ensure that your local X-Server is running (icon in the taskbar).
- Open a graphical text editor over the shell (`gedit myfile.txt`).

Test the X-Forwarding (2)



Accessing Remote Desktops

Accessing Remote Desktops

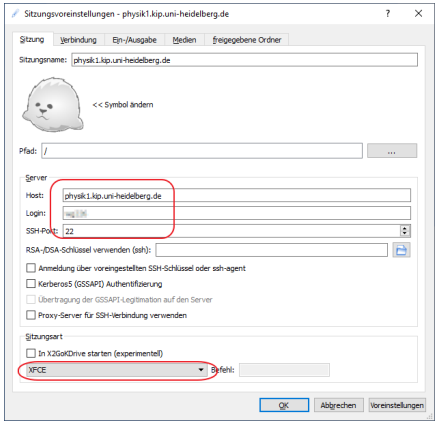
- Sometimes just redirecting windows is not enough.
- Accessing a full desktop is sometimes desirable.
- Latency of SSH X-Forward is crucial: Cannot use SSH X-Forward over WAN connections properly (especially with DSL/cable modem).
- Other means of using graphical applications remotely are better:
  - VNC: Godfather of remote screen access – usually low performance due to working principle. Works almost everywhere.
  - X2Go: More or less an optimized version of X-Forwarding that performs way better than the "direct" method.
  - Starnet FastX: Similar to X2Go, but faster and requires paid license.
  - Microsoft Remotedesktop: Very powerful tool to connect to RDP-Servers.

X2Go

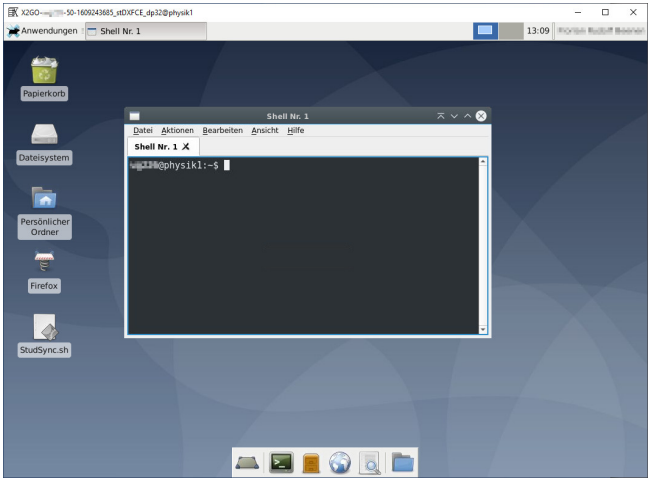
- Consists of server and client.
- Remote system must have X2Go server installed, otherwise we can stop here ☺.
- Client is available for all usual operating systems.
- Enables to open virtual desktop in a local window.
- Also supports redirection of audio (for playing music on remote server).
- Traffic is tunneled through SSH again.

X2Go Setup

- Download and install X2Go client from here: `https://wiki.x2go.org/doku.php/download:start`.
- Open the client and configure session:
  - Create a new session.
  - Enter the host `physik1.kip.uni-heidelberg.de`
  - Use your Uni-ID as username.
  - Leave the port at 22 (default).
  - Select XFCE instead of KDE as the session type.
  - Do not bother with SSH keys (still broken at KIP).
- Save and open the connection.
- Enter your password and accept the warning that the host key is unknown.
- Remember to open VPN prior to connecting to KIP resources!

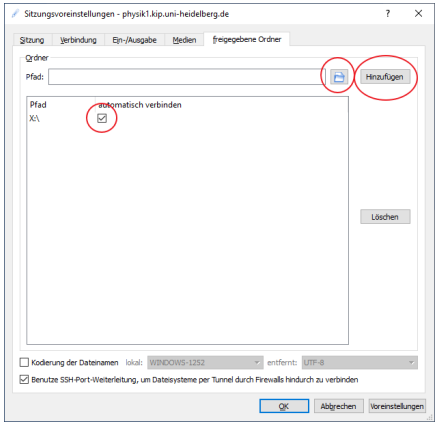


X2Go Desktop



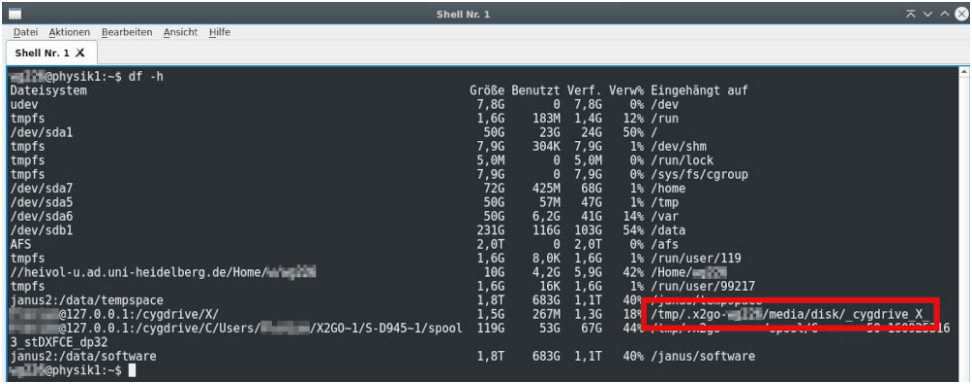
X2Go Filesharing

- It is possible to mount a *local* folder to the remote server.
- Allows for easy data exchange between local machine and remote server.
- Configure folder sharing:
  - Open X2Go session configuration.
  - Select rightmost tab for shared directories
  - Add a local directory.
  - Select *connect automatically*.
  - Save the session and connect.

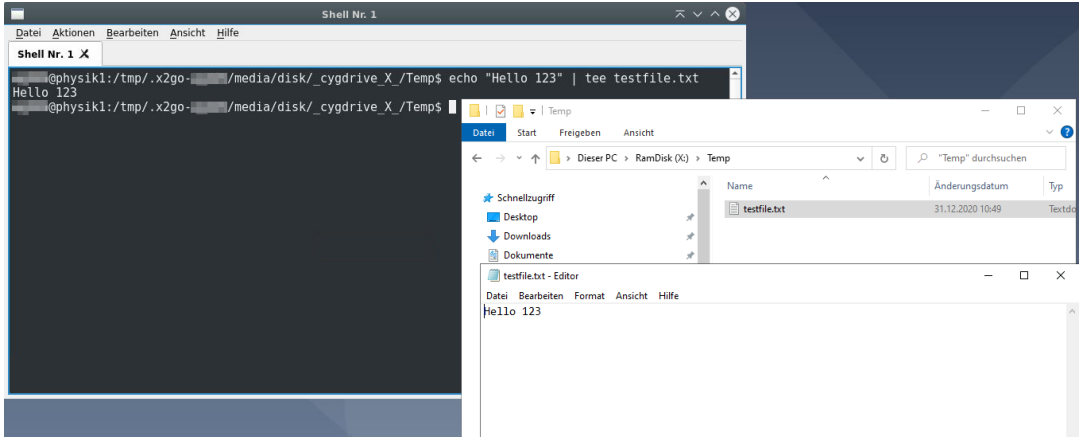


X2Go Filesharing (2)

- Open terminal and execute `df -h`.
- Locate the shared mount point.
- Change directory to the mount point.



X2Go Filesharing (3)



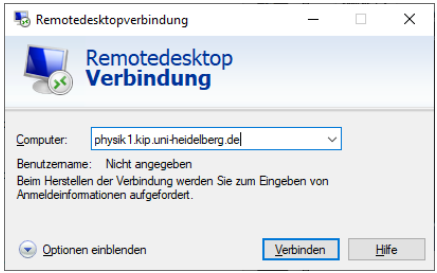
Microsoft Remotedesktop

- Microsoft Remotedesktop is a built-in tool to connect to Windows machines.
- **The** remote administration tool for graphic environments.
- Has many extremely useful features:
  - View and control a (virtual) remote desktop (obviously).
  - Exchange files.
  - Redirect audio devices (listen to remote audio, redirect local microphone to remote server).
  - Pass-through of local printers: Print document from remote server on local printer.
  - Sharing of clip-board (Ctrl+C, Ctrl+V between local system and remote system).
- Uses the *Remote Desktop Protocol* (RDP), Port TCP/3389.
  - Can be encrypted (but does not have to be).
  - Usage of RDP has many security implications because of interactions with the Windows host.
  - Usage is almost always firewalled ➥ Not available from the internet.

Remotedesktop Software

What we need:

- Remotedesktop Client:
  - Pre-installed on every Windows client. Accessible via *Windows-Key* + *R* → `mstsc` → *Enter*.
  - Open Source clients for Linux: `rdesktop`, `remmina`.
  - Microsoft App for Mac clients: `https://apps.apple.com/de/app/microsoft-remote-desktop/id1295203466?mt=12`.
- Remotedesktop Server:
  - Practically included with every Windows system. Deactivated due to licensing on Windows Home editions. Needs to be activated manually and local firewall needs to allow RDP connections.
  - For Linux: Use `xrdp`. Requires separate display manager and desktop environment (e. g. XFCE).



Remotedesktop in University

- University has different Remotedesktop servers for different purposes:
  - `physik1.kip.uni-heidelberg.de` ... `physik5.kip.uni-heidelberg.de`: Hosted by KIP, Linux machines with `xrdp`. Can run Mathematica.
  - `tsneu.ad.uni-heidelberg.de`: Hosted by URZ, Windows 2016 Server. Accessible from internet ☺.
  - `ts-mita.ad.uni-heidelberg.de`: Only accessible for employees of university (incl. HiWi). Has MS Office installed and more power than `tsneu`. Also accessible from internet.
- You can use these servers to surf the internet from university network (without VPN, SSH, ...).
- Manage your files in your Uni-ID account.
- Use as jump host for SSH connections for X-Forwarding (➥ see later, very useful).
- You cannot run Minecraft servers, Cryptominers, etc. Jobs will get killed after period of non-interactivity.

Test URZ Remotedesktop

Try the Remotedesktopserver

- Open the Remotedesktop client of your choice.
- Connect to `tsneu.ad.uni-heidelberg.de` (does not need VPN).
- Login with Uni-ID (e. g. jb007) username in format: `ad\jb007`.
- Open a web browser and check that you are indeed in the university network (e. g. download the IEEE paper from Slide 15 again).
- Open the Windows Explorer and check that you can see and open your files.
- Open a video from e. g. YouTube in the remote web browser and check that you can hear the audio on your local computer. You will most likely be able to watch 1080p videos with proper frame rate without problems.
- This works best from Windows clients: Mark a file on your local system and copy it (Ctrl+C). Switch to the remote session, open the Windows Explorer there and paste the file somewhere (Ctrl+V). This works vice versa. Try it!

Remotedesktop Options

- There are a few configuration options in the client that you may want to check.
- Tab “Local Resources”:
  - Control if the audio should be passed through to you or be played back on the physical audio interface of the remote machine (useful to prank colleagues in the office).
  - Control the mapping of the clip-board: It is often useful to copy+paste text or objects from the remote session to the local session and vice versa.
  - Pass through devices like storage media, printers, etc.
- Tab “Experience” / “Leistung”:
  - Tune performance of the connection.
  - If you have enough bandwidth, set everything to maximal settings ☺. Most of the time you don’t need more than 2 MBit/s (as long as you don’t play videos).

Remotedesktop Summary

- Remotedesktop has superior performance in comparison to X2Go, etc. on Windows machines.
- Is installed on every Windows client by default.
- Supports many useful features for productivity (File access, audio redirection, seamless Ctrl+C, Ctrl+V interaction).
- Needs a proper server application to be installed/enabled/licensed.
- My opinion: Performs identical to X2Go for connections to Linux servers (that use `xrdp`).

Other Useful Tools

Introduction

- The following contains an overview of tools that I personally find very useful.
- Listing may be biased a bit towards stuff that I do a lot...
- We will look at:
  - Having a proper Linux Shell in Windows (WSL).
  - Sharing files in a less shady way (without Dropbox, WeTransfer).
  - Using your smartphone as webcam.
  - Splitting and merging PDF files.
  - Creating "scans" of documents in PDF format with a smartphone.

Other Useful Tools

—

Windows Subsystem for Linux

What is WSL

- WSL is a feature in Windows 10 that allows to work in a “emulated” Linux environment.
- Directly integrated in Windows – does not need external tools like Cygwin.
- A selection of popular environments is available (e. g. Ubuntu, OpenSuse, Fedora, CentOS).
- Is integrated in networking stack: Can share IP / ports with Windows system (WSL 1 only).
- Is integrated in storage system: Can mount all drives visible to the respective Windows user (incl. network drives mounted by the Windows user).
- Is bound to the current Windows user: Every Windows user has their own Linux environment.

➥ I (personally) can do all the things I want to do in a Linux environment. Therefore I can replace Dual-Boot with Windows only and WSL (you can like this or not ☺).

What is WSL (2)

- There are two versions of WSL (you can check in Powershell with `wsl -l -v` what you have running).
- WSL 1 is something like a compatibility layer between Windows and Linux: System calls are emulated.
- Performance measurements yielded that the Linux inside WSL does not run much slower than a direct install.
- Docker and other kernel voodoo does not work in WSL 1.
- WSL 2 is more comparable to a virtual machine (VM).
- WSL 2 runs one single Linux kernel which is shared by all the WSL instances that you may open.
- WSL 2 allows to run Docker, etc.

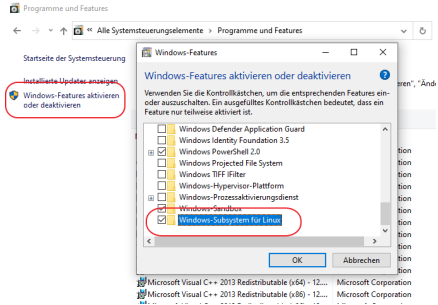
Why I don't like WSL 2

- Accessing files on the Windows host seem to be slower (according to Microsoft).
- Network is separated from host network: If server software is run inside WSL (e. g. Webserver), you need tricky proxying or NATing from Windows to WSL.
- WSL 2 does not support IPv6 at the moment.

➥ At the moment I only run on WSL 1. Microsoft advises to use WSL 2 whenever possible since the overall performance is supposed to be better.

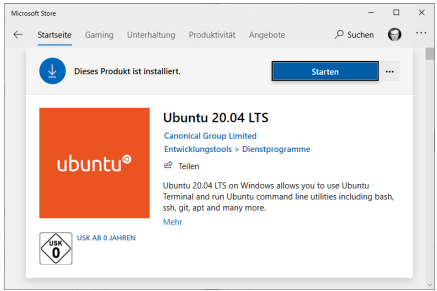
Activating WSL

- Open the “old” *System Control* panel and open *Programs and Features*.
- On the left side click *Turn Windows Features on or off*.
- Scroll down – one of the last options is *Windows Subsystem for Linux*. Activate it.
- Reboot your machine.



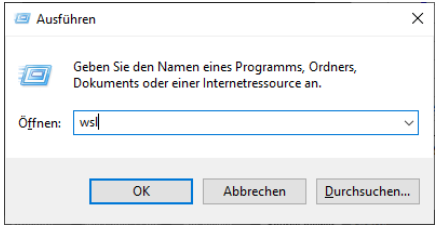
Installing Linux Flavor

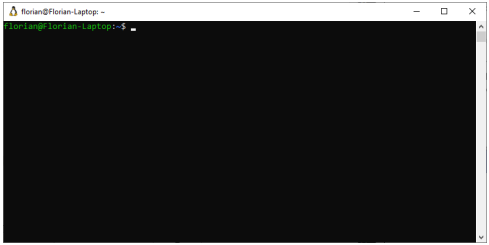
- Next, you need to select your Linux version (e. g. Ubuntu, Fedora, etc.).
- The easiest way is through the Microsoft Store:
  - Search for “Ubuntu” or which flavor you want.
  - You may be forced to log in with a Microsoft account. If you use a local Windows user instead of MS account, make sure that Windows did not change this when you logged in now!!
  - Install the selected Linux flavor.
- Alternative way without MS Store: `https://docs.microsoft.com/en-us/windows/wsl/install-manual`.



Starting WSL

- You can start your WSL instance...
  - ...by searching for *Ubuntu* in the Start Menu.
  - ...by pressing *Windows Key* + *R* → `wsl` → *Enter*.
- On first launch you need to create a Linux user. User name and password can be chosen freely.





Installing Software in WSL

- If you need additional software in your WSL, you can easily install it via `apt` (when using Ubuntu).
- Try to install Git by typing `sudo apt install git`.
- Sudo is required for this and you need to type in your WSL password.
- After successful install you can use Git normally via the `git` command.
- Remember to keep your software updated:
  - Execute `sudo apt update` to receive metadata from repository servers.
  - Execute `sudo apt dist-upgrade` to install new software versions automatically.

Other Useful Tools

—

Sharing Files

Sharing Files without Dropbox

- It happens that you need to send someone a file that cannot/should not be attached to an E-Mail.
- You use file sharing services like OneDrive, Dropbox or WeTransfer...
- The university has its own sharing service that is better, less shady and more trustworthy:
  - heiBOX: `https://heibox.uni-heidelberg.de`
  - GigaMove: `https://gigamove.rwth-aachen.de`
- Data is stored in university network.

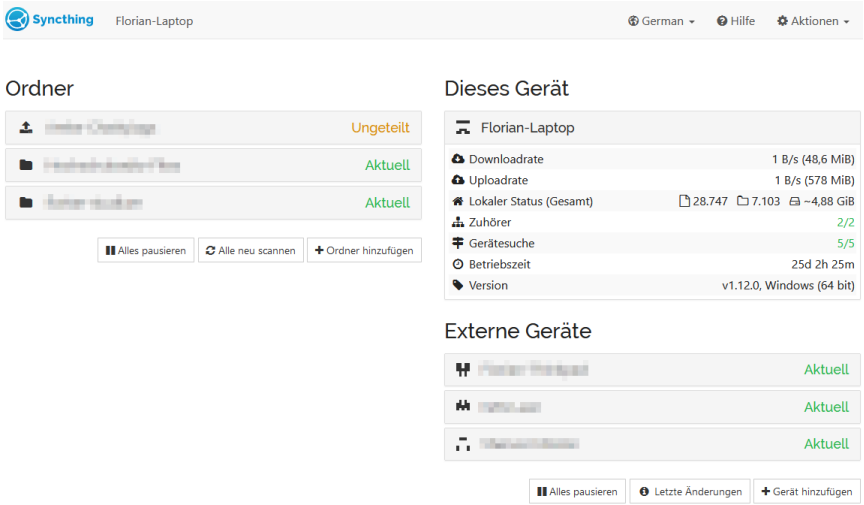
Sharing Files without Dropbox (2)

- heiBOX
  - Behaves like Dropbox. Is powered by Seafile.
  - Every student has 10 GB free online storage (heiBOX), every employee (incl. HiWi) has 30 GB storage.
  - Unlimited data retention period (➥ Files are not deleted).
  - Can create download and upload links for files and directories: public links or only for selected users.
  - Can invite other people (without Uni-ID) for working together on shared files.
  - Supports WebDAV access ➥ Can write scripts for automated up- and downloads ☺.
- GigaMove
  - Behaves like WeTransfer.
  - Is hosted by RWTH Aachen and accessible for people with DFN accounts (= Uni-ID).
  - Everyone has 1 TB storage, files get deleted automatically after 7 to 14 days.
  - Easy and useful alternative for sending large E-Mail attachments (large is $\geq$ 5 MB for me).

File Synchronization with Syncthing

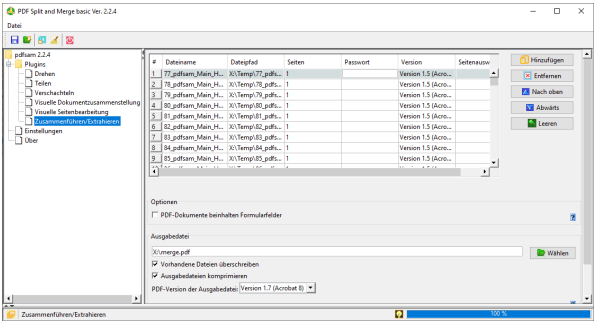
- If you have multiple devices and need to keep files synchronized, you need a proper tool for this.
- Often, you also want to have backups and different versions of your files.
- Syncthing (`https://syncthing.net/`) is free and open source and browser-based.
- Can install on all major operating systems.
- Runs in the background, observes configured directories and synchronizes files between hosts.
- Supports file versioning and “trashcanning”.
- File transfer is encrypted, data is only stored on your machines.
- Connections are end-to-end (if possible) or via relay servers (if there are firewall issues).

File Synchronization with Syncthing (2)



Reordering PDFs

- Sometimes you need to split, merge, rotate PDF files or remove pages from them.
- PDFsam can do this (`https://pdfsam.org/en/pdfsam-basic/`). Free version is sufficient for basic needs.
- Disadvantages: Cannot edit PDF metadata, does not recognize shared objects among merged PDF elements.



Scanning PDF Files

- When generating PDF documents from physical papers, a “real” scanner is best (you can get really good ones for around 50 Euro).
- If you don’t have a real scanner, you can use your smartphone.
- Please don’t just take a photo and send it to someone (e. g. exercise submission for university). Your professor/tutor will be most displeased ☹.
- Use an app that performs trapezoid correction and document extraction:
  - Office Lens[1]
    - Developed by Microsoft and free to use.
    - More or less fool proof ☺.
  - Adobe Scan[2]
    - Also free to use but requires log in (e. g. with Google account).
    - OCR works better (Office Lens supposedly supports OCR, but I can’t get it to work).
    - Has good color correction (white is really white).
    - Less straightforward to use (in my opinion).

[1] https://play.google.com/store/apps/details?id=com.microsoft.office.officelens&hl=de&gl=US
[2] https://play.google.com/store/apps/details?id=com.adobe.scan.android&hl=de&gl=US

Smartphone as Webcam

- People with laptops can use integrated webcam (that has debatable quality).
- When using a non-laptop computer or when using a docking station with laptop you need an external webcam.
- Good webcams can be expensive. Really good webcams can be really expensive ☺.
- Your smartphone can produce pictures/videos of reasonable high quality. You can use an app to connect your phone with your PC.
- Android users may want to use DroidCam[1]:
  - Requires app on smartphone and program on your PC.
  - Data is transferred via WiFi (consumes quite some bandwidth: 5 MBit/s to 10 MBit/s).
  - You must (manually) type in hostname/IP of your phone/tablet computer.
  - Free version only delivers up to 480p video stream. Sufficient for most applications.
- You can mount your smartphone on a tripod over your computer screen.

[1] https://play.google.com/store/apps/details?id=com.dev47apps.droidcam&hl=de&gl=US