

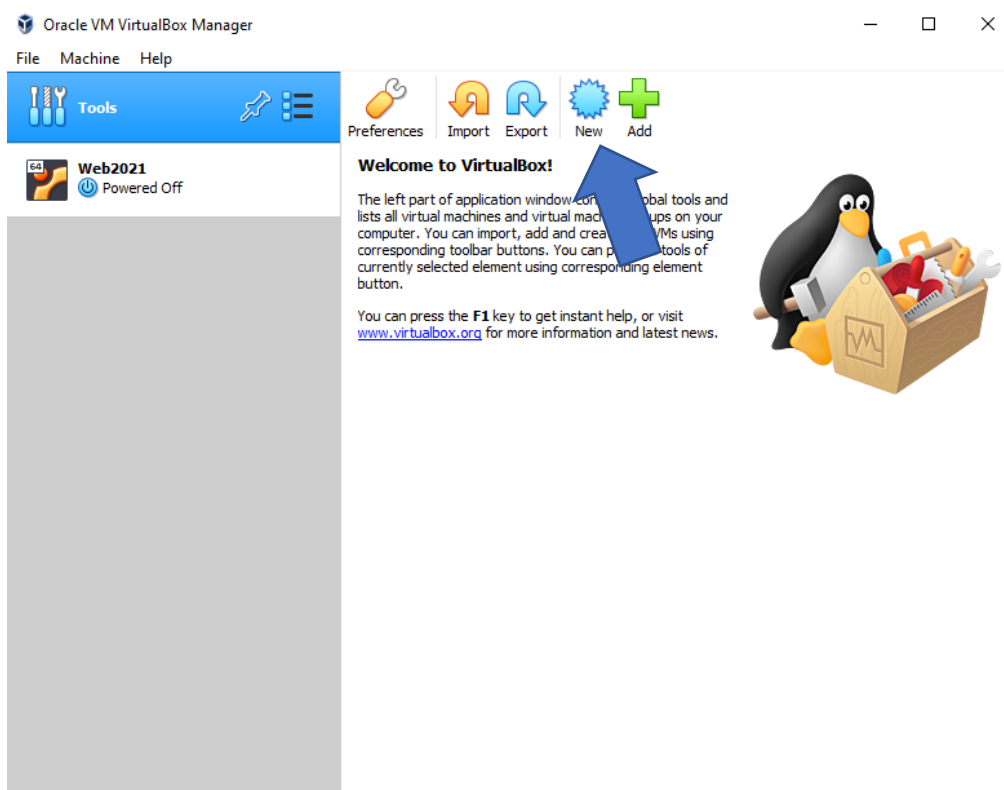
VM aufsetzen mit Ubuntu-Server

Installation von LAMP und Node.js

Zuerst muss Oracle VM Virtualbox installiert werden.

(<https://www.virtualbox.org/wiki/Downloads>)

In Virtualbox kann man nun über den „New“ Button eine neue VM erstellen. Ich benutze von jetzt an den „Expert Mode“, der im ersten Fenster nach „New“ ausgewählt werden kann. Dieser Modus macht die ganze Sache nicht schwerer.



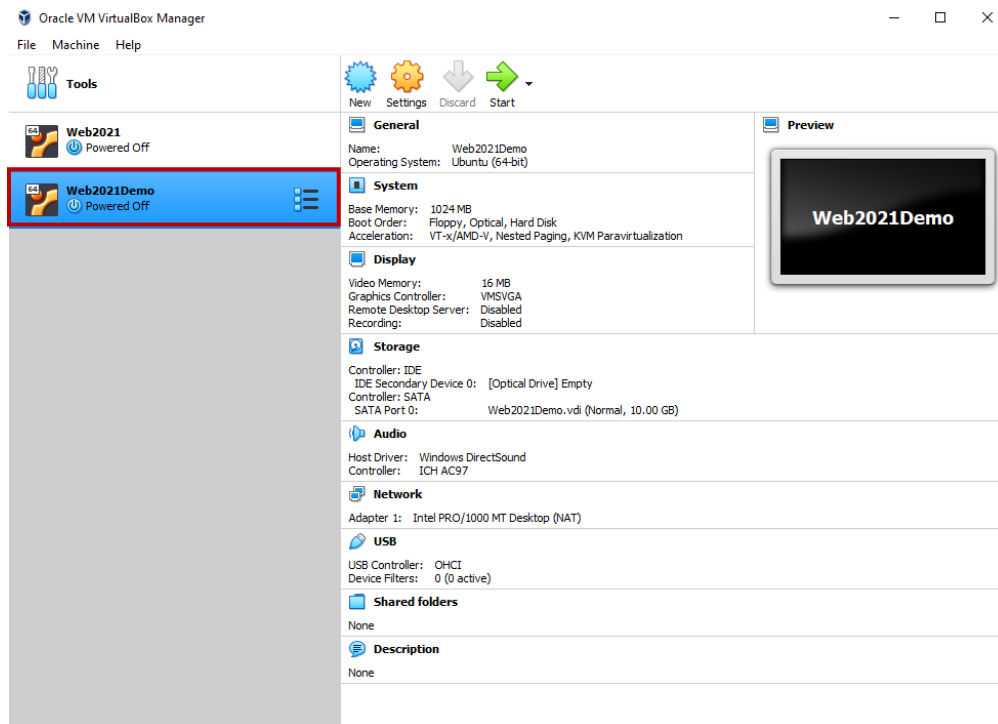
Beim Erstellen kann der Name und der Speicherort frei gewählt werden. In dieses Beispiel müssen nur „Type“ auf „Linux“ und „Version“ auf „Ubuntu (64-bit)“ geändert werden.

The screenshot shows the 'Create Virtual Machine' window. The 'Name and operating system' section has 'Name' set to 'Web2021Demo' and 'Machine Folder' set to 'C:\Users\Nils\VirtualBox VMs'. The 'Type' is set to 'Linux' and the 'Version' is set to 'Ubuntu (64-bit)'. The 'Memory size' is set to 1024 MB. The 'Hard disk' section has 'Create a virtual hard disk now' selected, and the file name is 'Web2021.vdi (Normal, 10.00 GB)'. At the bottom, there are buttons for 'Guided Mode', 'Create', and 'Cancel'.

Nun muss noch eine virtuelle Festplatte erstellt werden. Hierbei habe ich 10 GB als Größe gewählt. Dies müsste völlig ausreichen. Auch hier kann der Speicherort frei gewählt werden, jedoch ist es Empfehlenswert die virtuelle Festplatte im selben Ordner, wie die VM zu speichern.

The screenshot shows the 'Create Virtual Hard Disk' window. The 'File location' is set to 'C:\Users\Nils\VirtualBox VMs\Web2021Demo\Web2021Demo.vdi'. The 'File size' is set to 10.00 GB. The 'Hard disk file type' section has 'VDI (VirtualBox Disk Image)' selected. The 'Storage on physical hard disk' section has 'Dynamically allocated' selected. At the bottom, there are buttons for 'Guided Mode', 'Create', and 'Cancel'.

Nachdem „Create“ betätigt wurde, ist nun eine neue VM im Manager zu sehen.



Diese VM hat jedoch noch kein Betriebssystem. Hierzu muss zuerst ein .iso File heruntergeladen werden (<https://ubuntu.com/download/server>). Ich benutze den Ubuntu-Server 20.04.2 LTS.

CANONICAL We are hiring Products ▾

ubuntu® Enterprise ▾ Developer ▾ Community ▾ Download ▾ Search 🔍 Sign in

Downloads Server > ARM POWER s390x Provisioning

Get Ubuntu Server

Option 2: Manual server installation

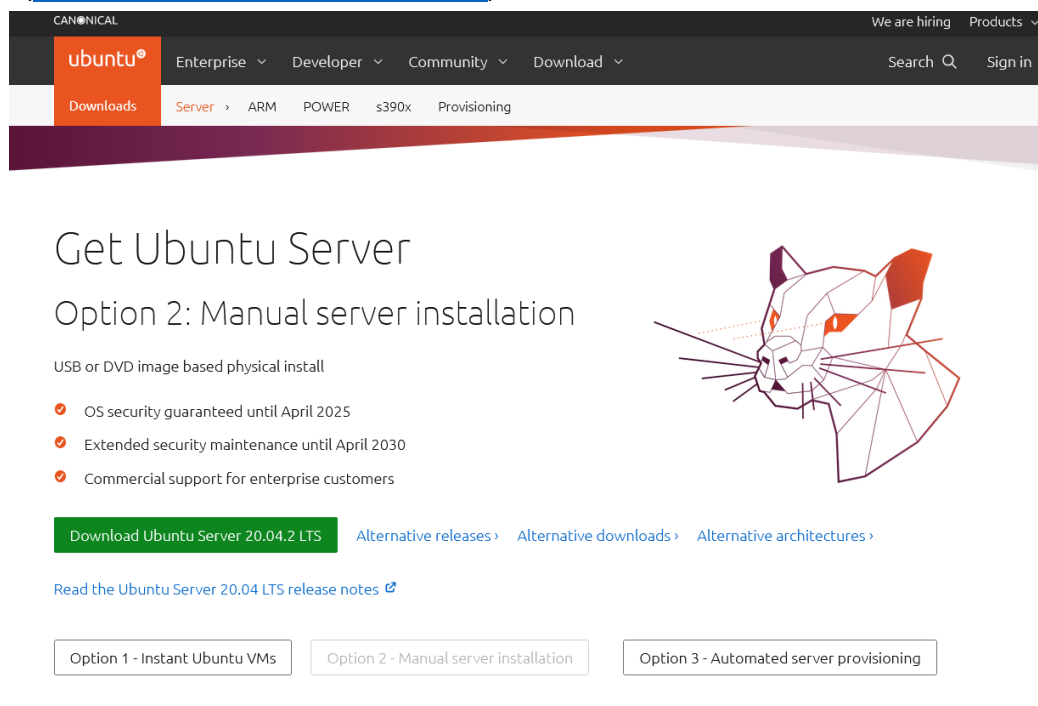
USB or DVD image based physical install

- ✓ OS security guaranteed until April 2025
- ✓ Extended security maintenance until April 2030
- ✓ Commercial support for enterprise customers

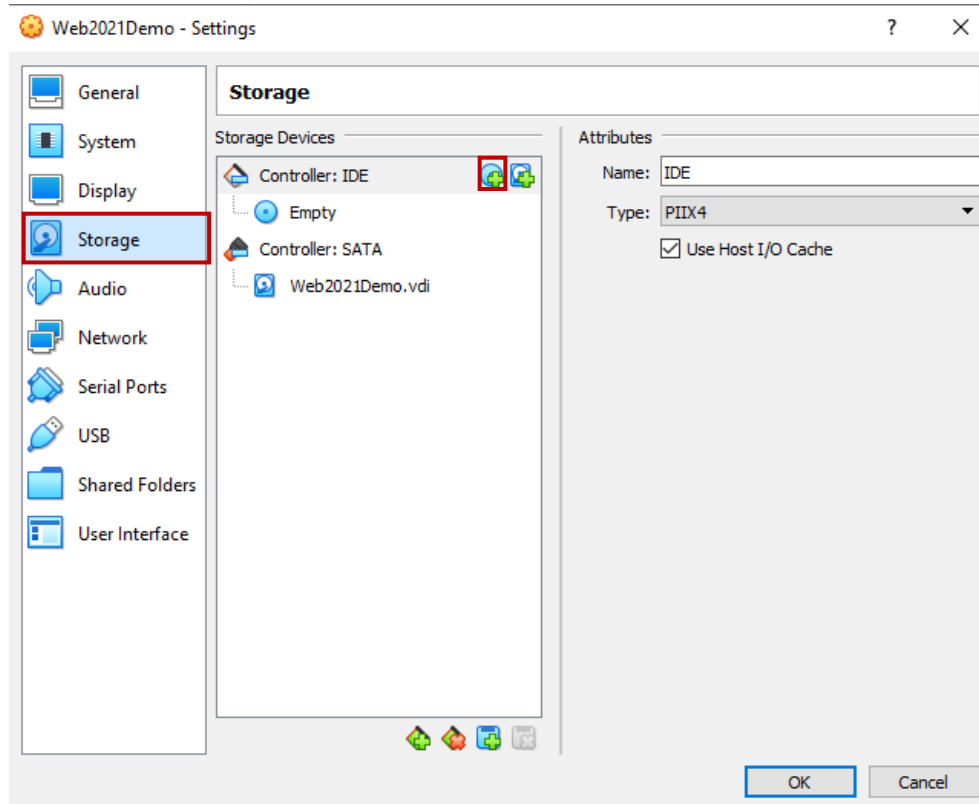
[Download Ubuntu Server 20.04.2 LTS](#) [Alternative releases >](#) [Alternative downloads >](#) [Alternative architectures >](#)

[Read the Ubuntu Server 20.04 LTS release notes](#) [↗](#)

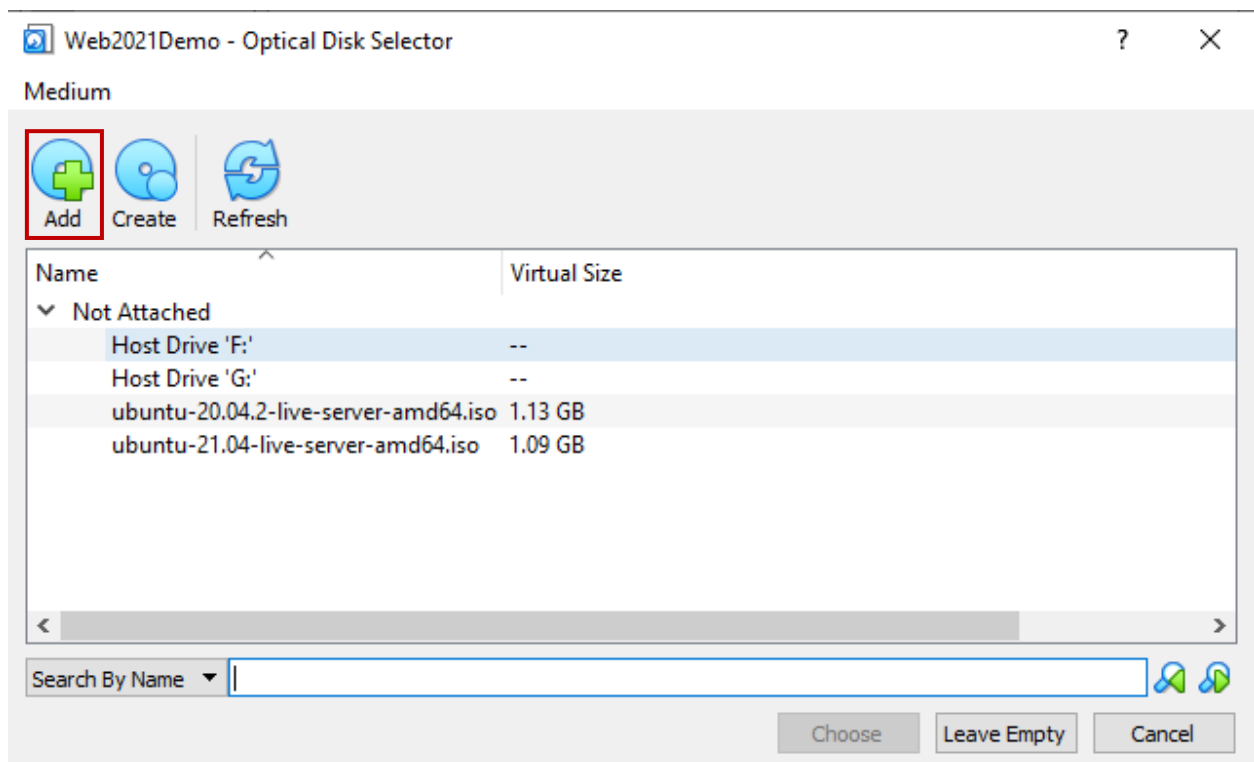
Option 1 - Instant Ubuntu VMs Option 2 - Manual server installation Option 3 - Automated server provisioning

The image is a screenshot of the Ubuntu Server 20.04.2 LTS landing page. It features the Canonical logo at the top left and navigation links for Enterprise, Developer, Community, and Download. The main heading is 'Get Ubuntu Server' with a sub-heading 'Option 2: Manual server installation'. Below this, it mentions 'USB or DVD image based physical install' and lists three key benefits: OS security guaranteed until April 2025, extended security maintenance until April 2030, and commercial support for enterprise customers. There are links for downloading the Ubuntu Server 20.04.2 LTS ISO, alternative releases, alternative downloads, and alternative architectures. At the bottom, there are three buttons: 'Option 1 - Instant Ubuntu VMs', 'Option 2 - Manual server installation' (which is highlighted), and 'Option 3 - Automated server provisioning'. On the right side of the page, there is a stylized illustration of a cat's head, which is the Ubuntu logo.

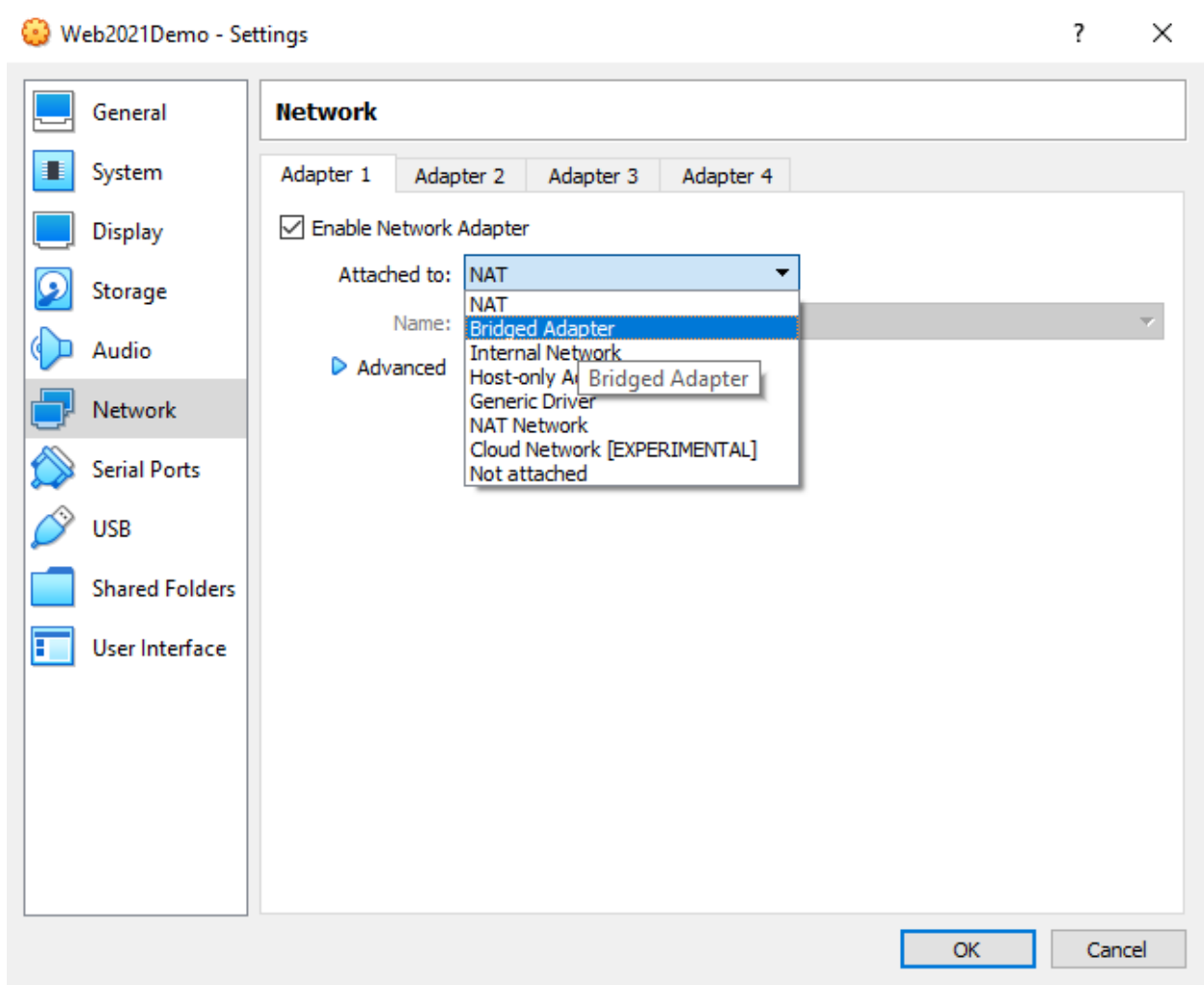
Dies File muss jetzt in den Settings der VM ausgewählt werden. Dazu muss unter „Storage“ auf die kleine Disk mit dem Plus bei „Contoller: IDE“ gedrückt werden.



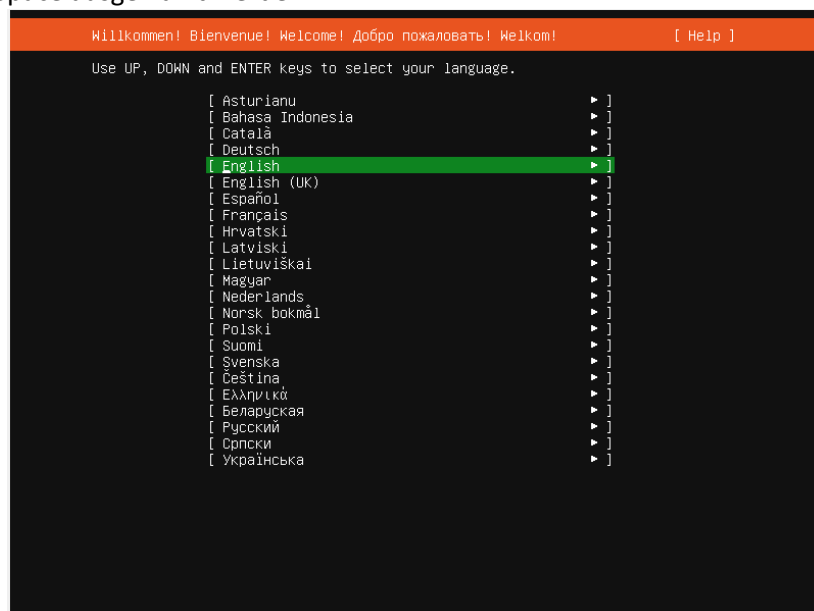
Nun öffnet sich dieses Fenster und die .iso Datei kann ausgewählt werden. Falls die Datei nicht wie bei mir Vorhanden ist kann diese durch einen Click auf „Add“ aus dem Dateisystem hinzugefügt werden.



Nun muss nur noch die Netzwerkeinstellung zu „bridged adapter“ geändert werden unter „Network“.



Nun können die Einstellungen geschlossen werden und die VM kann über „Start“ gestartet werden. Es öffnet sich ein neues Fenster in dem die VM startet. Einfach so lange warten bis die Sprachauswahl von Ubuntu erscheint. Es wird navigiert mit den Pfeiltasten und mit Enter wird ausgewählt. Bei Optionen können diese mit Space ausgewählt werden.



Nun folgen ein paar Einstellungen die jeder so einstellen kann wie er möchte. Bei Network, Proxy und Ubuntu archive mirror habe ich nichts geändert.

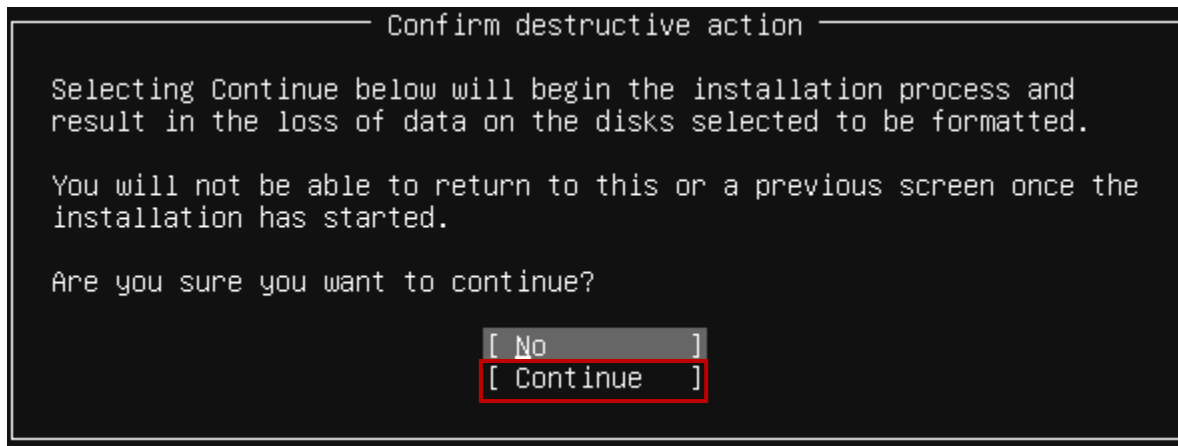
Nun kommt man zur storage configuration. Hier habe ich die komplette Disk für das Betriebssystem ausgewählt.

The screenshot shows the 'Guided storage configuration' window. It has an orange header with 'Guided storage configuration' on the left and '[Help]' on the right. The main text says 'Configure a guided storage layout, or create a custom one:'. There are three main options: 1. '(X) Use an entire disk' (selected), which shows a sub-option '[VBOX_HARDDISK_VB7ecd2879-8884895d local disk 10.000G ▾]'. 2. '[X] Set up this disk as an LVM group'. 3. '[] Encrypt the LVM group with LUKS', which has sub-options for 'Passphrase:' and 'Confirm passphrase:'. At the bottom is '() Custom storage layout'. At the very bottom are two buttons: '[Done]' and '[Back]'.

Im nächsten Screen auch einfach weiter mit „Done“.

The screenshot shows the 'Storage configuration' window. It has an orange header with 'Storage configuration' on the left and '[Help]' on the right. The main section is titled 'FILE SYSTEM SUMMARY' and contains a table with columns 'MOUNT POINT', 'SIZE', 'TYPE', and 'DEVICE TYPE'. The table has two rows: one for '/' with size 8.996G, type 'new ext4', and device type 'new LVM logical volume'; and one for '/boot' with size 1.000G, type 'new ext4', and device type 'new partition of local disk'. Below this is the 'AVAILABLE DEVICES' section, which says 'No available devices' and has two options: '[Create software RAID (md) ►]' and '[Create volume group (LVM) ►]'. Then is the 'USED DEVICES' section, which contains a table with columns 'DEVICE', 'TYPE', and 'SIZE'. It lists 'ubuntu-vg (new)' as an 'LVM volume group' of size 8.996G, 'ubuntu-lv' as 'new, to be formatted as ext4, mounted at /' with size 8.996G, and the 'VBOX_HARDDISK_VB7ecd2879-8884895d' as a 'local disk' of size 10.000G, which is further detailed as having three partitions: 'partition 1 new, bios_grub' (1.000M), 'partition 2 new, to be formatted as ext4, mounted at /boot' (1.000G), and 'partition 3 new, PV of LVM volume group ubuntu-vg' (8.997G). At the bottom are three buttons: '[Done]' (highlighted in green), '[Reset]', and '[Back]'.

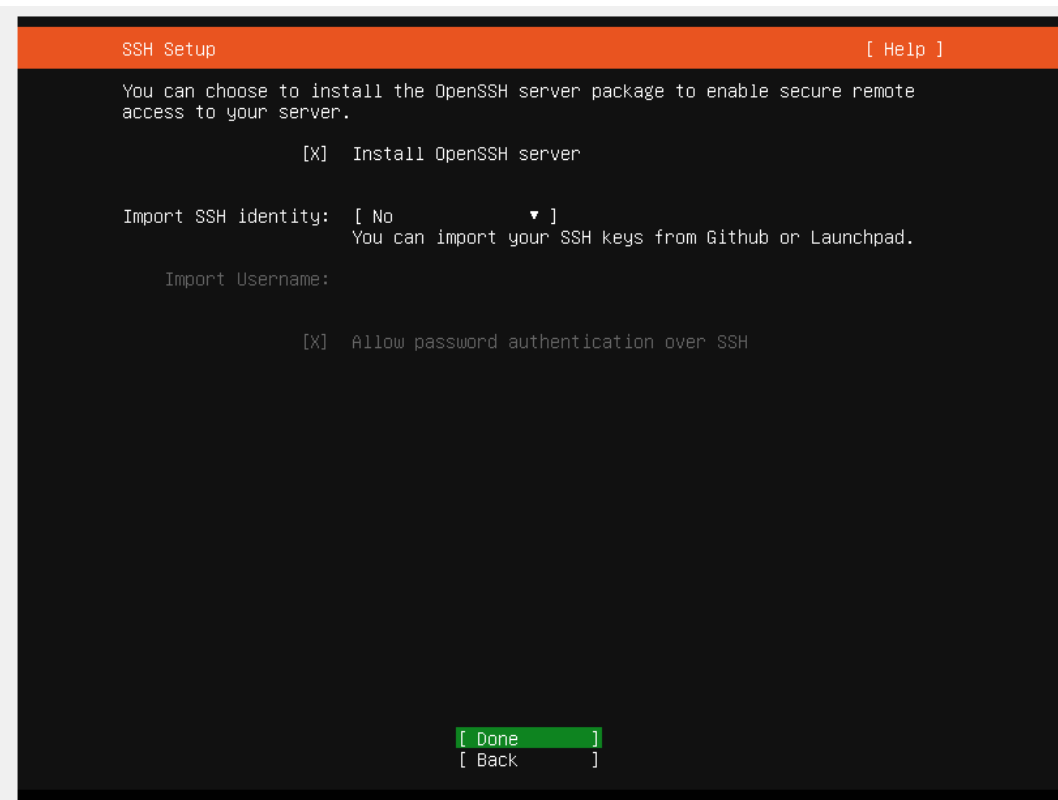
Und bei der Warnung auf „Continue“.



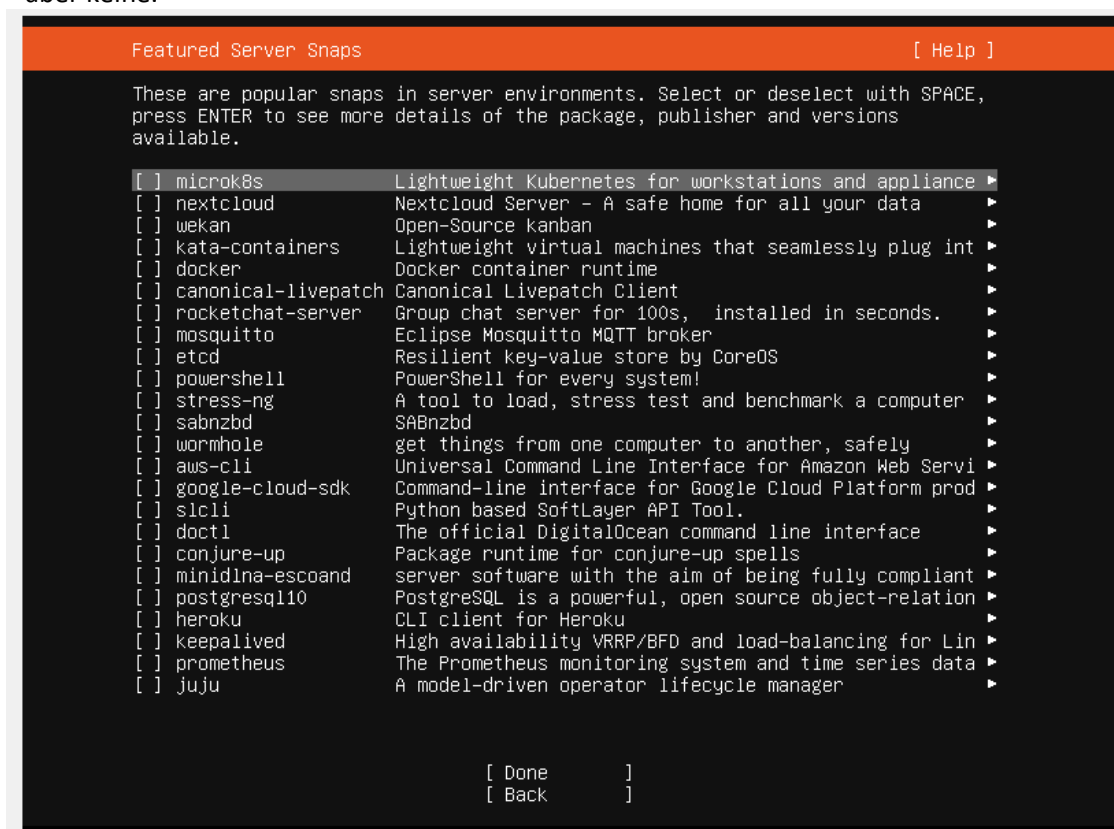
Nun werden Name und Passwort abgefragt. Wichtig hierbei ist das Passwort und der Username, da man diese zum Login auf dem Server braucht. Server Name wird in der Kommandoline immer angezeigt also sollte dieser sinnvoll sein.

A "Profile setup" screen with an orange header bar containing the title "Profile setup" and a "[Help]" link. The main area has a dark background with white text. It instructs the user to "Enter the username and password you will use to log in to the system. You can configure SSH access on the next screen but a password is still needed for sudo." Below this are five input fields: "Your name:", "Your server's name:" (with a subtext "The name it uses when it talks to other computers."), "Pick a username:", "Choose a password:", and "Confirm your password:". At the bottom, there is a "[Done]" button.

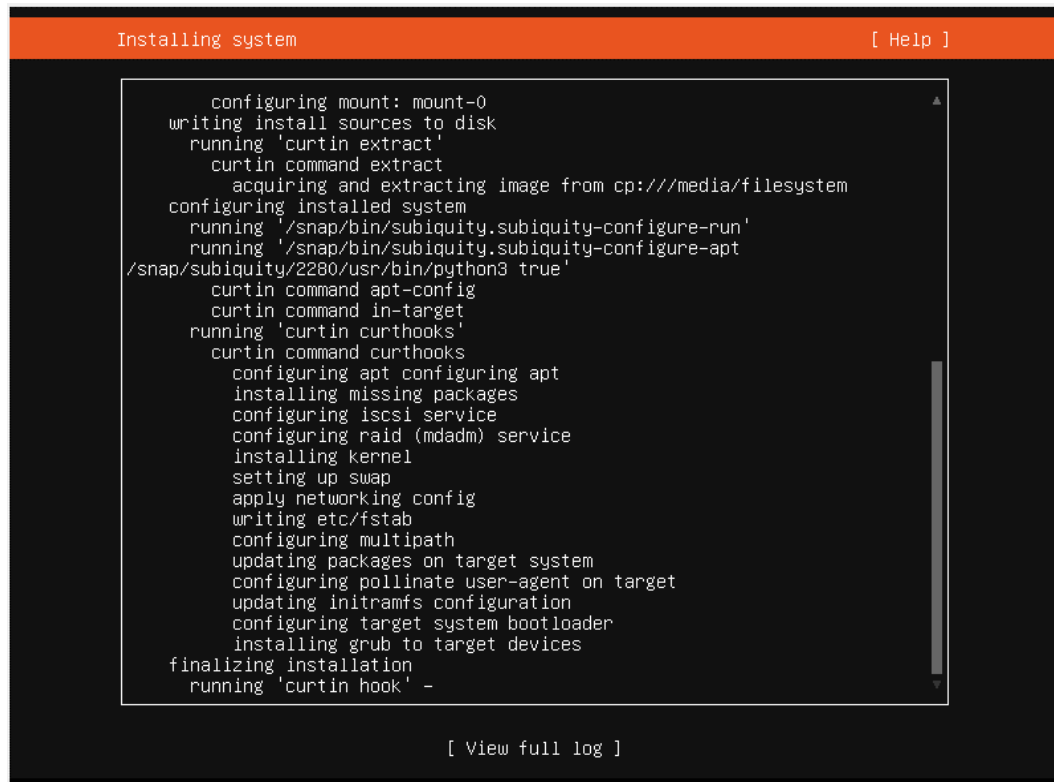
Auf dem nächsten Screen wird man gefragt, ob man den OpenSSH server installiert haben möchte. Ich habe diesen ausgewählt mit Space, da ich git über ssh benutze.



Auf dem nächsten Screen wird weitere Software zum Installieren vorgeschlagen. Hiervon brauchen wir aber keine.



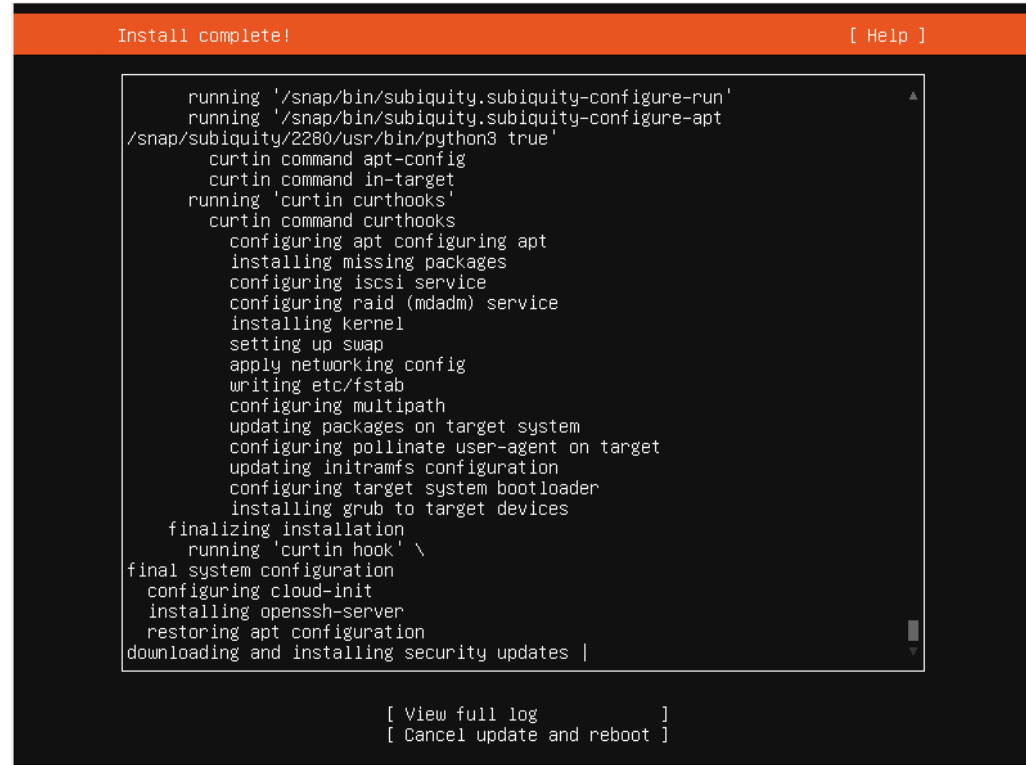
Nun wird das System und danach noch Updates installiert.



The screenshot shows a window titled "Installing system" with a "[Help]" button in the top right corner. The main area contains a list of installation steps in a terminal-like font. The steps include configuring mount, writing install sources, running 'curtin extract', acquiring and extracting an image, configuring the installed system, running subiquity configuration scripts, running 'curtin apt-config', 'curtin in-target', and 'curtin curthooks', configuring apt, installing missing packages, configuring iSCSI and RAID services, installing the kernel, setting up swap, applying networking, writing fstab, configuring multipath, updating packages, configuring pollinate and initramfs, configuring the bootloader, and finally installing grub. The window ends with a "[View full log]" button.

```
configuring mount: mount-0
writing install sources to disk
  running 'curtin extract'
    curtin command extract
      acquiring and extracting image from cp:///media/filesystem
configuring installed system
  running '/snap/bin/subiquity.subiquity-configure-run'
  running '/snap/bin/subiquity.subiquity-configure-apt
/snap/subiquity/2280/usr/bin/python3 true'
  curtin command apt-config
  curtin command in-target
  running 'curtin curthooks'
  curtin command curthooks
    configuring apt configuring apt
    installing missing packages
    configuring iSCSI service
    configuring raid (mdadm) service
    installing kernel
    setting up swap
    apply networking config
    writing etc/fstab
    configuring multipath
    updating packages on target system
    configuring pollinate user-agent on target
    updating initramfs configuration
    configuring target system bootloader
    installing grub to target devices
finalizing installation
  running 'curtin hook' -
```

[View full log]



The screenshot shows a window titled "Install complete!" with a "[Help]" button in the top right corner. The main area contains a list of post-installation steps. It starts with running subiquity configuration scripts, followed by 'curtin apt-config', 'curtin in-target', and 'curtin curthooks'. Then it lists the same configuration steps as the previous window (configuring apt, installing packages, configuring iSCSI and RAID, installing kernel, setting up swap, applying networking, writing fstab, configuring multipath, updating packages, configuring pollinate and initramfs, configuring the bootloader, and installing grub). After 'finalizing installation' and running 'curtin hook', it proceeds to 'final system configuration', which includes configuring cloud-init, installing openssh-server, restoring apt configuration, and downloading and installing security updates. The window ends with two buttons: "[View full log]" and "[Cancel update and reboot]".

```
running '/snap/bin/subiquity.subiquity-configure-run'
running '/snap/bin/subiquity.subiquity-configure-apt
/snap/subiquity/2280/usr/bin/python3 true'
  curtin command apt-config
  curtin command in-target
  running 'curtin curthooks'
  curtin command curthooks
    configuring apt configuring apt
    installing missing packages
    configuring iSCSI service
    configuring raid (mdadm) service
    installing kernel
    setting up swap
    apply networking config
    writing etc/fstab
    configuring multipath
    updating packages on target system
    configuring pollinate user-agent on target
    updating initramfs configuration
    configuring target system bootloader
    installing grub to target devices
finalizing installation
  running 'curtin hook' \
final system configuration
  configuring cloud-init
  installing openssh-server
  restoring apt configuration
  downloading and installing security updates |
```

[View full log]
[Cancel update and reboot]

Einfach so lange warten bis unten steht „Reboot now“ und dies dann auch betätigen.

Nun startet Ubuntu neu und es wird wahrscheinlich erst dieser Screen kommen. Einfach kurz warten und Enter drücken.

[illegible]

Nun startet Ubuntu-Server richtig. Wenn Ubuntu fertig ist, erscheint dies hier. Dies ist zuerst sehr unübersichtlich, da ein paar Dinge noch passieren, aber wenn nichts mehr passiert, will der Server einen Login. Als erstes gebt ihr den Username an und bestätigt mit Enter und dann das noch mal mit dem Passwort. Achtung: Das Passwort wird nicht angezeigt auch nicht mit ****, einfach eingeben und Enter.

Wenn ihr alles richtig gemacht habt, kommt eine kleine Willkommensnachricht und ihr seid auf der Komandozeile von eurem Server. Ubuntu-Server hat nämlich keine visuelle Oberfläche.

```
webdemo login: nils
Password:
Welcome to Ubuntu 20.04.2 LTS (GNU/Linux 5.4.0-73-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

System information as of Fri 14 May 2021 01:45:53 PM UTC

System load:  0.02           Processes:           90
Usage of /:   43.2% of 8.79GB Users logged in:       0
Memory usage: 19%           IPv4 address for enp0s3: 192.168.178.66
Swap usage:   0%

60 updates can be installed immediately.
0 of these updates are security updates.
To see these additional updates run: apt list --upgradable


The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

nils@webdemo:~$
```

Da man in dem VM Fenster nicht einfügen kann, würde ich empfehlen sich mit dem Tool PuTTY zu Server zu verbinden. PuTTY ist eigentlich ein Tool für Remote Komandozeilen Zugriff über SSH, kann aber auch für unseren Server verwendet werden. Hierzu braucht man die IP des Servers, dazu später mehr. (<https://www.chiark.greenend.org.uk/~sgtatham/putty/latest.html>) PuTTY gibt es wohl nur auf Windows. Auf Linux und Mac müsse solch ein Tool schon existieren.

Als erstes gibt man am besten „sudo -i“ ein. Man wird nochmal nach dem Passwort gefragt, aber man arbeitet jetzt als „root“ und muss nicht immer das Passwort neu eingeben oder „sudo“ vor jeden Command schreiben. Man befindet sich nun im Home-Folder, wir wollen aber zum Root-Folder des Betriebssystems. Dazu führen wir „cd ..“. Mit „ls“ kann man sich alle Dateien und Ordner im aktuellen Verzeichnis anzeigen lassen.

```
nils@webdemo:~$ sudo -i
root@webdemo:~# cd ..
root@webdemo:/# ls
bin  cdrom  etc  lib  lib64  lost+found  mnt  proc  run  snap  swap.img  tmp  var
boot  dev  home  lib32  libx32  media  opt  root  sbin  srv  sys  usr
```

Um die IP-Adresse des Servers herauszufinden müssen wir zuerst das network-tool über „apt install net-tools“. Wenn dies fertig ist kann man mit „ifconfig“ die IP-Adresse anzeigen lassen.

```
root@webdemo:/# ifconfig
enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
    inet 192.168.178.66  netmask 255.255.255.0  broadcast 192.168.178.255
    inet6 fe80::a00:27ff:fe23:3def  prefixlen 64  scopeid 0x20<link>
    ether 08:00:27:23:3d:ef  txqueuelen 1000  (Ethernet)
    RX packets 8294  bytes 2314690 (2.3 MB)
    RX errors 0  dropped 2081  overruns 0  frame 0
    TX packets 392  bytes 35188 (35.1 KB)
    TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
    inet 127.0.0.1  netmask 255.0.0.0
    inet6 ::1  prefixlen 128  scopeid 0x10<host>
    loop txqueuelen 1000  (Local Loopback)
    RX packets 106  bytes 8546 (8.5 KB)
    RX errors 0  dropped 0  overruns 0  frame 0
    TX packets 106  bytes 8546 (8.5 KB)
    TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

root@webdemo:/#
```

Nun müssen nur noch die Tools (LAMP bzw. Node.js) installiert werden.

Node:

Um Node zu installieren, müssen zwei Befehle ausgeführt werden:

1. "curl -fsSL https://deb.nodesource.com/setup_14.x | sudo -E bash -"
2. "sudo apt-get install -y nodejs"

In Nummer 1 steht die 14 für die Version. Es gibt auch 12, 15 und 16. Je nach dem welche Version ihr braucht. Nachzulesen hier: (<https://github.com/nodesource/distributions/blob/master/README.md>)

1.

```
root@webdemo:/$ curl -fsSL https://deb.nodesource.com/setup_14.x | sudo -E bash -
## Installing the NodeSource Node.js 14.x repo...

## Populating apt-get cache...

+ apt-get update
Hit:1 http://de.archive.ubuntu.com/ubuntu focal InRelease
Hit:2 http://de.archive.ubuntu.com/ubuntu focal-updates InRelease
Hit:3 http://de.archive.ubuntu.com/ubuntu focal-backports InRelease
Hit:4 http://de.archive.ubuntu.com/ubuntu focal-security InRelease
Reading package lists... Done

## Confirming "focal" is supported...

+ curl -sLf -o /dev/null 'https://deb.nodesource.com/node_14.x/dists/focal/Release'

## Adding the NodeSource signing key to your keyring...

+ curl -s https://deb.nodesource.com/gpgkey/nodesource.gpg.key | gpg --dearmor | tee /usr/share/keyrings/nodesource.gpg >/dev/null

## Creating apt sources list file for the NodeSource Node.js 14.x repo...

+ echo 'deb [signed-by=/usr/share/keyrings/nodesource.gpg] https://deb.nodesource.com/node_14.x focal main' > /etc/apt/sources.list.d/nodesource.list
+ echo 'deb-src [signed-by=/usr/share/keyrings/nodesource.gpg] https://deb.nodesource.com/node_14.x focal main' >> /etc/apt/sources.list.d/nodesource.list

## Running 'apt-get update' for you...

+ apt-get update
Hit:1 http://de.archive.ubuntu.com/ubuntu focal InRelease
Get:2 https://deb.nodesource.com/node_14.x focal InRelease [4,583 B]
Hit:3 http://de.archive.ubuntu.com/ubuntu focal-updates InRelease
Hit:4 http://de.archive.ubuntu.com/ubuntu focal-backports InRelease
Hit:5 http://de.archive.ubuntu.com/ubuntu focal-security InRelease
Get:6 https://deb.nodesource.com/node_14.x focal/main amd64 Packages [768 B]
Fetched 5,351 B in 2s (3,325 B/s)
Reading package lists... Done

## Run 'sudo apt-get install -y nodejs' to install Node.js 14.x and npm
## You may also need development tools to build native addons:
    sudo apt-get install gcc g++ make
## To install the Yarn package manager, run:
    curl -sL https://dl.yarnpkg.com/debian/pubkey.gpg | gpg --dearmor | sudo tee /usr/share/keyrings/yarnkey.gpg >/dev/null
    echo "deb [signed-by=/usr/share/keyrings/yarnkey.gpg] https://dl.yarnpkg.com/debian stable main" | sudo tee /etc/apt/sources.list.d/yarn.list
    sudo apt-get update && sudo apt-get install yarn

root@webdemo:/$
```

2.

```
root@webdemo:/$ sudo apt-get install -y nodejs
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following NEW packages will be installed:
  nodejs
0 upgraded, 1 newly installed, 0 to remove and 62 not upgraded.
Need to get 24.8 MB of archives.
After this operation, 120 MB of additional disk space will be used.
Get:1 https://deb.nodesource.com/node_14.x focal/main amd64 nodejs amd64 14.17.0-deb-1nodesource1 [24.8 MB]
Fetched 24.8 MB in 4s (6,822 kB/s)
Selecting previously unselected package nodejs.
(Reading database ... 71383 files and directories currently installed.)
Preparing to unpack .../nodejs_14.17.0-deb-1nodesource1_amd64.deb ...
Unpacking nodejs (14.17.0-deb-1nodesource1) ...
Setting up nodejs (14.17.0-deb-1nodesource1) ...
Processing triggers for man-db (2.9.1-1) ...
root@webdemo:/$
```

Und schon ist Node installiert.

Um nun euren Node Server zu starten, braucht ihr erst mal eure Dateien in Ubuntu. Am einfachsten geht dies über git, falls ihr eure Projektdateien in einem git liegen habt. Wenn nicht bitte erstellt ein git auf github oder dem Hochschulgit. (Wie git funktioniert sollte man eigentlich mittlerweile wissen). Ich clone das repo gerne in den Ordner var. Um in diesen zu kommen, gebt ihr ein „cd var/“. Mit cd [Ordnername] springt ihr in den angegebenen Ordner und mit „cd ..“ springt ihr einen Ordner nach oben. Und mit „ls“ könnt ihr euch jeder Zeit alle Dateien und Ordner im aktuellen Verzeichnis anzeigen lassen. Im Ordner „var“ erstelle ich mit jetzt gerne einen neuen Ordner namens „www“ mit dem Befehl „mkdir www“ und dann springe ich in diesem Ordner mit „cd www/“. Nun klonst ihr euer repo in diesen Ordner mit „git clone [url zum repo]“.

```
root@webdemo:/var# mkdir www
root@webdemo:/var# cd www/
root@webdemo:/var/www# git clone https://github.com/bkotikov/web-projekt.git
```

Nun könnt ihr wieder mit cd in euren Projektordner wechseln und in dem dann euren Node-Server starten über „node app.js“ z.B. Der Server läuft dann auf dem Port den ihr angegeben habt in z.B. app.js

Node startet nicht automatisch und man muss dies erst selbst konfigurieren. Dazu muss man wieder zum Datei root mit „cd ..“ (eventuell mehrmals „cd ..“). Nun springt man in dem Ordner /etc/init.d/ mit „cd etc/init.d/“. Alle Dateien in diesem Ordner werden immer nach dem Boot ausgeführt, auch ohne einen Login. Nun erstellen wir eine neue Datei mit „nano node“ und es öffnet sich ein Editor in den ihr jetzt ein Template copiert. Link zum Template: <https://raw.githubusercontent.com/fhd/init-script-template/master/template> . Hierzu empfiehlt sich PuTTY, da man in Virtualbox nicht einfügen kann.

```
root@webdemo:/# cd etc/init.d/
root@webdemo:/etc/init.d# nano node
```

```
GNU nano 4.8 node Modified
^_/bin/sh
### BEGIN INIT INFO
# Provides:
# Required-Start: $remote_fs $syslog
# Required-Stop: $remote_fs $syslog
# Default-Start: 2 3 4 5
# Default-Stop: 0 1 6
# Short-Description: Start daemon at boot time
# Description: Enable service provided by daemon.
### END INIT INFO

dir=""
cmd=""
user=""

name='basename $0'
pid_file="/var/run/$name.pid"
stdout_log="/var/log/$name.log"
stderr_log="/var/log/$name.err"

get_pid() {
    cat "$pid_file"
}

is_running() {
    [ -f "$pid_file" ] && ps -p $(get_pid) > /dev/null 2>&1
}

case "$1" in
    start)
        if is_running; then
            echo "Already started"
        else
            echo "Starting $name"
        fi
    ;;
    stop)
        if is_running; then
            echo "Stopping $name"
        else
            echo "Already stopped"
        fi
    ;;
    restart)
        stop
        start
    ;;
    status)
        if is_running; then
            echo "$name is running"
        else
            echo "$name is not running"
        fi
    ;;
    *)
        echo "Usage: $0 {start|stop|restart|status}"
        exit 1
    ;;
esac
```

^G Get Help ^O Write Out ^W Where Is ^K Cut Text ^J Justify ^C Cur Pos ^M-U Undo ^M-A Mark Text ^M-J To Bracket
^X Exit ^R Read File ^N Replace ^U Paste Text ^I To Spell ^H Go To Line ^M-B Redo ^M-G Copy Text ^H Where Was

Nun müsst ihr nur noch bei `dir=""` den Pfad zu eurem node Ordner angeben (z.B.: `dir="/var/www/webdemo"`). Auch müsst ihr den Befehl angeben, der ausgeführt werden soll in diesem Ordner, bei `cmd=""` (z.B.: `cmd="node app.js"`). Bei `user=""` gebt ihr einfach „root“ an.

```
dir="/var/www/webdemo"
cmd="node app.js"
user="root"
```

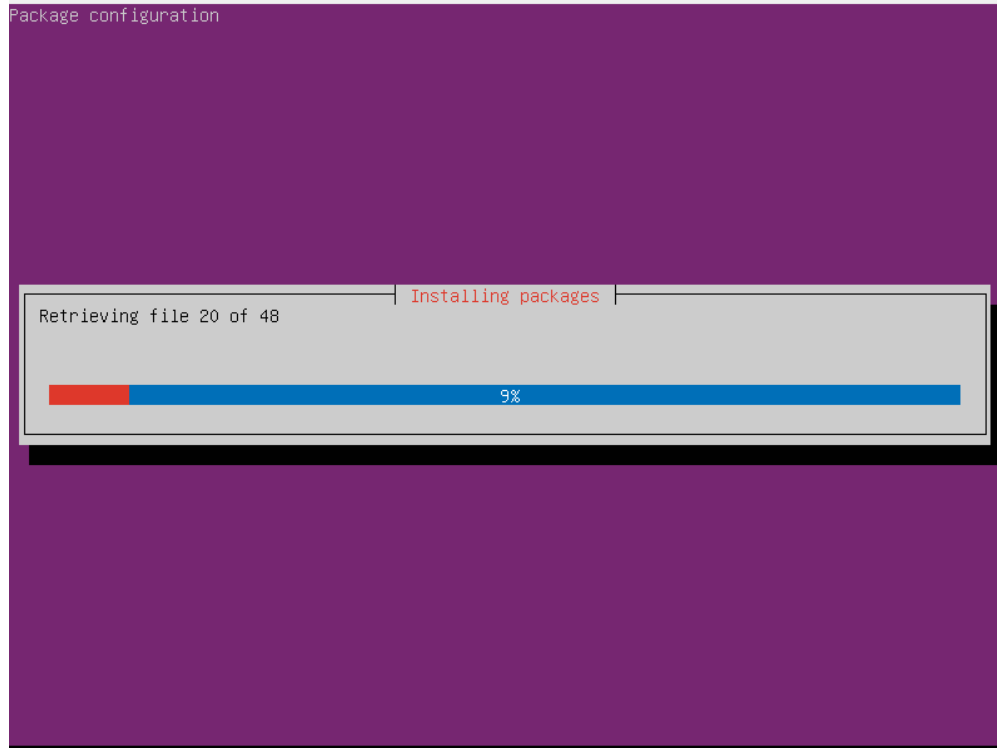
Der Server sollte jetzt immer beim Boot starten. Wenn ihr den Server vorher schon starten wollt gebt einfach „`/etc/init.d/node start`“ ein und der Server startet. „start“ könnt ihr auch mit „restart“ oder „stop“ ersetzen.

LAMP:

Bevor wir LAMP installieren können, brauchen wir erst noch ein Tool, dass uns die Installation vereinfacht. Dieses Tool wird mit dem Befehl „`apt-get install taskel`“ installiert.

```
root@webdemo:~# apt-get install taskel
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  laptop-detect taskel-data
The following NEW packages will be installed:
  laptop-detect taskel taskel-data
0 upgraded, 3 newly installed, 0 to remove and 62 not upgraded.
Need to get 40.0 kB of archives.
After this operation, 309 kB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 http://de.archive.ubuntu.com/ubuntu focal/main amd64 taskel-data all 3.34ubuntu16 [5,340 B]
Get:2 http://de.archive.ubuntu.com/ubuntu focal/main amd64 taskel all 3.34ubuntu16 [28.6 kB]
Get:3 http://de.archive.ubuntu.com/ubuntu focal/main amd64 laptop-detect all 0.16 [6,016 B]
Fetched 40.0 kB in 0s (194 kB/s)
Preconfiguring packages ...
Selecting previously unselected package taskel-data.
(Reading database ... 76300 files and directories currently installed.)
Preparing to unpack .../taskel-data_3.34ubuntu16_all.deb ...
Unpacking taskel-data (3.34ubuntu16) ...
Selecting previously unselected package taskel.
Preparing to unpack .../taskel_3.34ubuntu16_all.deb ...
Unpacking taskel (3.34ubuntu16) ...
Selecting previously unselected package laptop-detect.
Preparing to unpack .../laptop-detect_0.16_all.deb ...
Unpacking laptop-detect (0.16) ...
Setting up laptop-detect (0.16) ...
Setting up taskel (3.34ubuntu16) ...
Setting up taskel-data (3.34ubuntu16) ...
Processing triggers for man-db (2.9.1-1) ...
root@webdemo:~# _
```

Um LAMP nun zu installieren muss nur der Befehl „`taskel install lamp-server`“ ausgeführt werden.

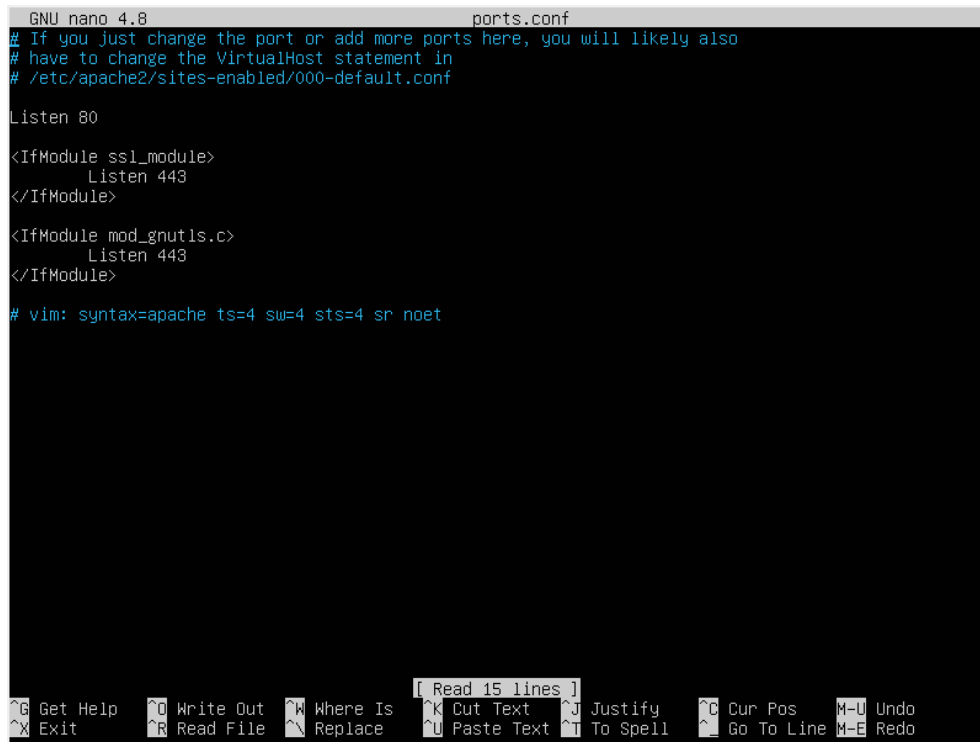


Danach ist der Server schon unter der IP-Adresse der VM mit Port 80 erreichbar. Der Server startet auch bei jedem Boot automatisch. Weitere Einstellungen am Server können natürlich gemacht werden. Dafür navigieren wir in den Ordner „cd etc/apache2/“. In diesem Ordner befinden sich alle Konfigurationsdateien des Apache Servers. apache2.conf ist wie httpd.conf die Hauptkonfigurationsdatei. Diese kann mit „nano apache2.conf“ editiert werden.

```
GNU nano 4.8                apache2.conf
# This is the main Apache server configuration file.  It contains the
# configuration directives that give the server its instructions.
# See http://httpd.apache.org/docs/2.4/ for detailed information about
# the directives and /usr/share/doc/apache2/README.Debian about Debian specific
# hints.
#
# Summary of how the Apache 2 configuration works in Debian:
# The Apache 2 web server configuration in Debian is quite different to
# upstream's suggested way to configure the web server. This is because Debian's
# default Apache2 installation attempts to make adding and removing modules,
# virtual hosts, and extra configuration directives as flexible as possible, in
# order to make automating the changes and administering the server as easy as
# possible.
#
# It is split into several files forming the configuration hierarchy outlined
# below, all located in the /etc/apache2/ directory:
#
#      /etc/apache2/
#      |-- apache2.conf
#      |   |-- ports.conf
#      |-- mods-enabled
#      |   |-- *.load
#      |   |-- *.conf
#      |-- conf-enabled
#      |   |-- *.conf
#      |-- sites-enabled
#      |   |-- *.conf
#
#
# * apache2.conf is the main configuration file (this file). It puts the pieces
# together by including all remaining configuration files when starting up the
# web server.

[ Read 227 lines ]
^G Get Help  ^O Write Out ^W Where Is  ^K Cut Text  ^J Justify   ^C Cur Pos   M-U Undo
^X Exit      ^R Read File ^_ Replace   ^U Paste Text ^T To Spell  ^_ Go To Line M-E Redo
```

In der Datei ports.conf kann der Port des Servers geändert werden. „nano ports.conf“



```
GNU nano 4.8 ports.conf
# If you just change the port or add more ports here, you will likely also
# have to change the VirtualHost statement in
# /etc/apache2/sites-enabled/000-default.conf

Listen 80

<IfModule ssl_module>
    Listen 443
</IfModule>

<IfModule mod_gnutls.c>
    Listen 443
</IfModule>

# vim: syntax=apache ts=4 sw=4 sts=4 sr noet

[ Read 15 lines ]
^G Get Help  ^O Write Out ^W Where Is  ^K Cut Text  ^J Justify   ^C Cur Pos   M-U Undo
^X Exit       ^R Read File ^_ Replace   ^U Paste Text ^T To Spell  ^_ Go To Line M-E Redo
```

Wenn man die Konfigurationsdateien ändert, muss man den Server mit dem Befehl „systemctl restart apache2“ neustarten.

Der Standard-Dokumenten-Root, also der Ordner, aus dem der Server die html Formulare lädt, ist var/www/html/. Zu diesem kann mit „cd var/www/html/“ gesprungen werden. Dieser kann aber auch geändert werden.

Bei weiteren Fragen könnt ihr euch gerne direkt an mich wenden: nils.rudolph@stud.hs-mannheim.de