Do not quote. Work in progress!

Fedora on External Rented Server
The Example of Hetzner

Abstract: Installation eines eigentständigen Servers auf gemieteter Hardware, wie es für kleinere und mittlere Unternehmen sich anbietet oder auch im privaten Bereit oder selbstständge Unternehmer. Installation in schritten erläutert, jeweils mit der unterliegenden Leitlinie (best practice)

Table of Content

1. Why Rent a Dedicated Server?

Even though cloud services, virtual servers, SAAS are on everyone's lips, complete control over hardware and software is attractive or even indispensable for companies and also private individuals in many cases. Data protection and confidentiality considerations, special software needs, or simply the joy of experimentation are reasons for doing so.

Rental servers in a data centre offer the advantage of a professional set-up and, above all, a *technically optimal internet connection*, as it can be afforded on one's own premises only by large corporations. They are offered by various companies. This guide is based on an offer from *Hetzner*, a large European company, which offers among others quite affordable options through its "Server Exchange". In principle, the instructions can be adopted to offers from other providers.

2. Preliminary considerations

The objective is to install and maintain a server that is not directly accessible locally. Various publicly accessible services are to be offered securely and reliably. These include standard services such as mail and web applications, but also sector-specific software.

The most important guideline is to carefully seal off access to the server as far as possible. It must be prevented by all means that it can be compromised or even hijacked. Due to the remote, not directly accessible location, a failure of unobstructed access is comparatively laborious to remedy and thus "expensive". Appropriate measures are:

- Access to the server exclusively via ssh and key-based identification
- The server operates exclusively as a "host" for public services without offering them directly
- All services such as mail or web are encapsulated and run either in virtual machines or a container (frontend as guest system)
- No accounts or only a few accounts required for system maintenance are created on the host itself. All other accounts required are placed in a guest system.
- Internal processing of data can be performed directly on the server (backend). This is particularly useful for I/O-intensive applications such as a database for performance reasons.
- Communication between VMs, containers and backend takes place via a protected internal, non-public network.

The aforementioned provisions are largely "best practice" for any server installation today, but become even more important when outsourcing off premise.

Hetzner rental servers are usually operated with a local hard disk or 2 disks in a RAID array. The strict separation of system and user data under this condition also serves to make administration as simple as possible. The system area, i.e. the operating system including installed utility programs and software such as a database system, must be maintainable completely independently of the storage of user data. System maintenance must not jeopardise user data under any circumstances. If necessary, it must be possible to unmount user data.

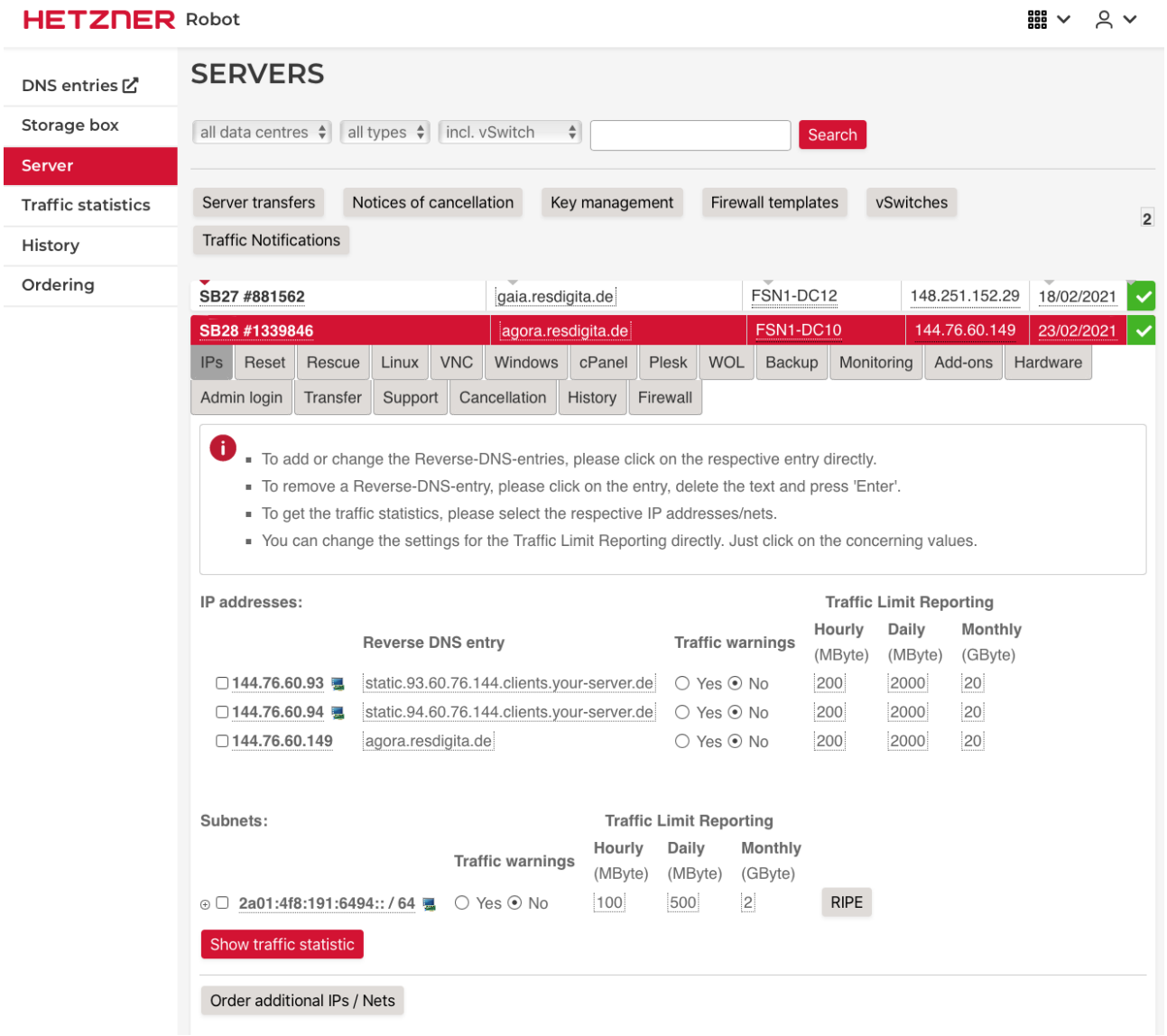
For precisely this reason, Fedora Server creates by default a small /boot partition and in the remaining area a partition with a volume group (VG). Therein, a logical volume of approx. 15 GB (the exact value depends on the disk capacity) is created for the operating system and its software. The other available space remains free for the creation of logical volumes (LVs) for user data, which are mounted in the appropriate positions in the directory tree of the system area (details later).

We will go a little further and create another small partition and VG for the operating system in addition to the partition for /boot (sysvg approx. 30 GiB). LVs for the directory tree and additional runtime environment are created therein and free space is left for disposal as needed. The remaining area of the hard disk is filled by a large partition and VG for user data (usrvg). Similar to the standard partition, all user data is created as LVs in usrvg and mounted in corresponding directories of the system area. This is the maximum possible separation of system and user data if only one hard disk is available.

3. Preparations

3.1 Preparations made in Hetzner Web Interface

Login into your Hetzner account and select „Server“ from the left hand menue. Expand your server.



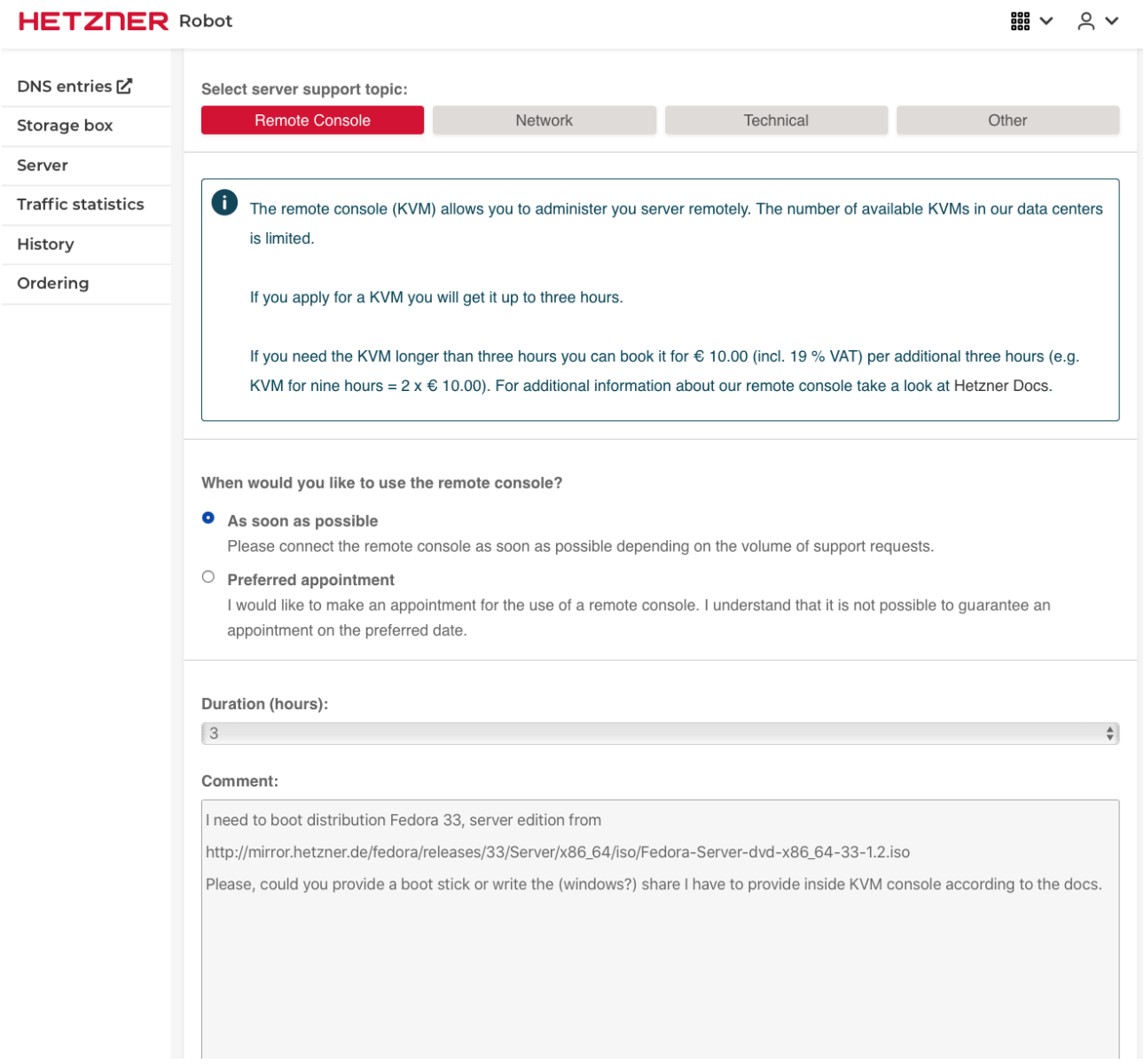
When you rent the server you get a /64 IPv6 subnet, but only one single IPv4 address. If you plan to use publicly available virtual machines with both IPv4 and IPv6 (dual stack, what you definitely want these days), you need more IPv4 addresses. You can order either various single IPs or an IPv4 subnet using the order button at the bottom. For the tutorial we use additional single Ips. On the screenshot above you see 2 single IPs added.

Next you have to ensure your rented server’s main IPv4 and IPv6 addresses are known in the DNS. Here we follow the convention for IPv6 to use ‚::2‘ from the subnet range for the host.

Hetzner offers various ways to install the operating system. Fedora is only available via VNC or via a remote console (KVM console). Via VNC you often don’t get the latest Fedora version and then it

used to be based on the everything DVD. You can select ‚Fedora Server' from the options list, but it is not the Fedora Server Edition. So it is recommended to use KVM console. It is not permanently connected to a server, but must be ordered from Hetzner Support for a limited period of time. It is free of charge for 3 hours, which is quite sufficient for an installation.

To order a KVM console klick on „Support" in the servers Web interface. Product type „Server" should already be selected as well as your server if you have only one. At the bottom click on „Remote Console". The lower part of the screen expands and an order form is displayed.

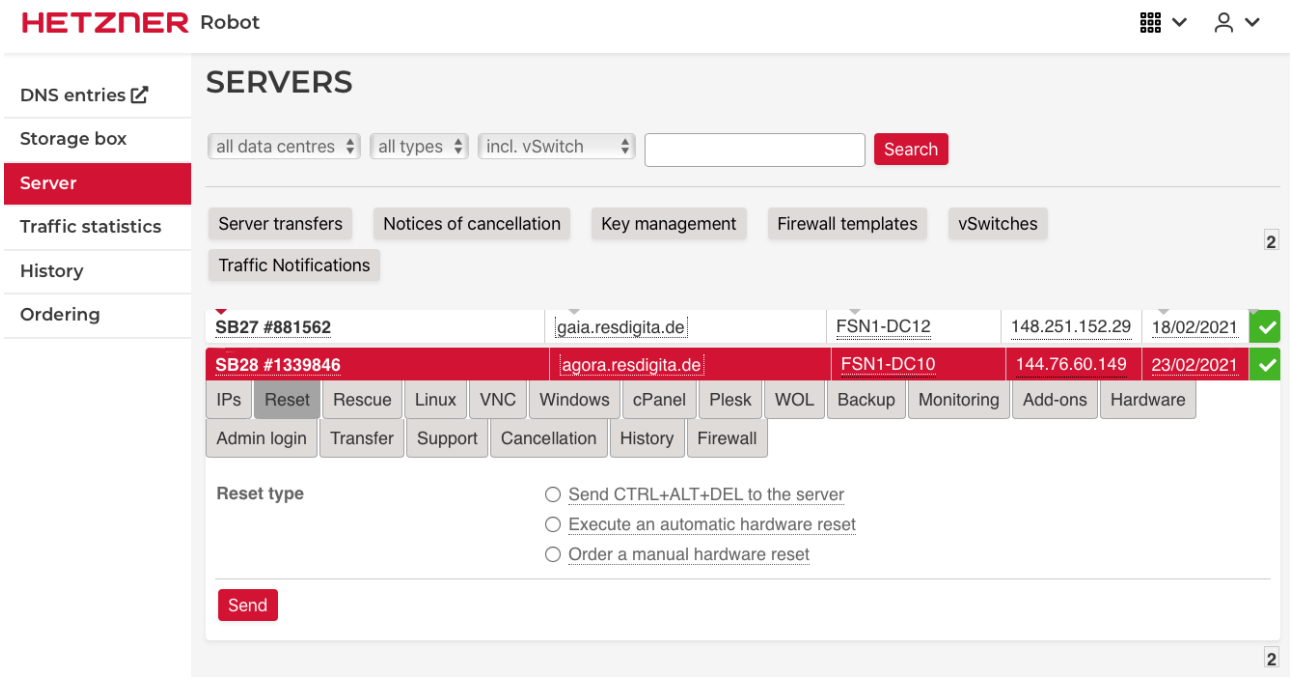


Fill in the form as needed and submit the request. The Hetzner mirror includes all Fedora versions. You find Fedora Server at http://mirror.hetzner.de/fedora/releases/33/Server/x86_64/iso/Fedora-Server-dvd-x86_64-33-1.2.iso. Usually the technicians provide a USB stick. Otherwise you need the share name.

The Remote Console requires Java 8. If not installed, you need to install it now.

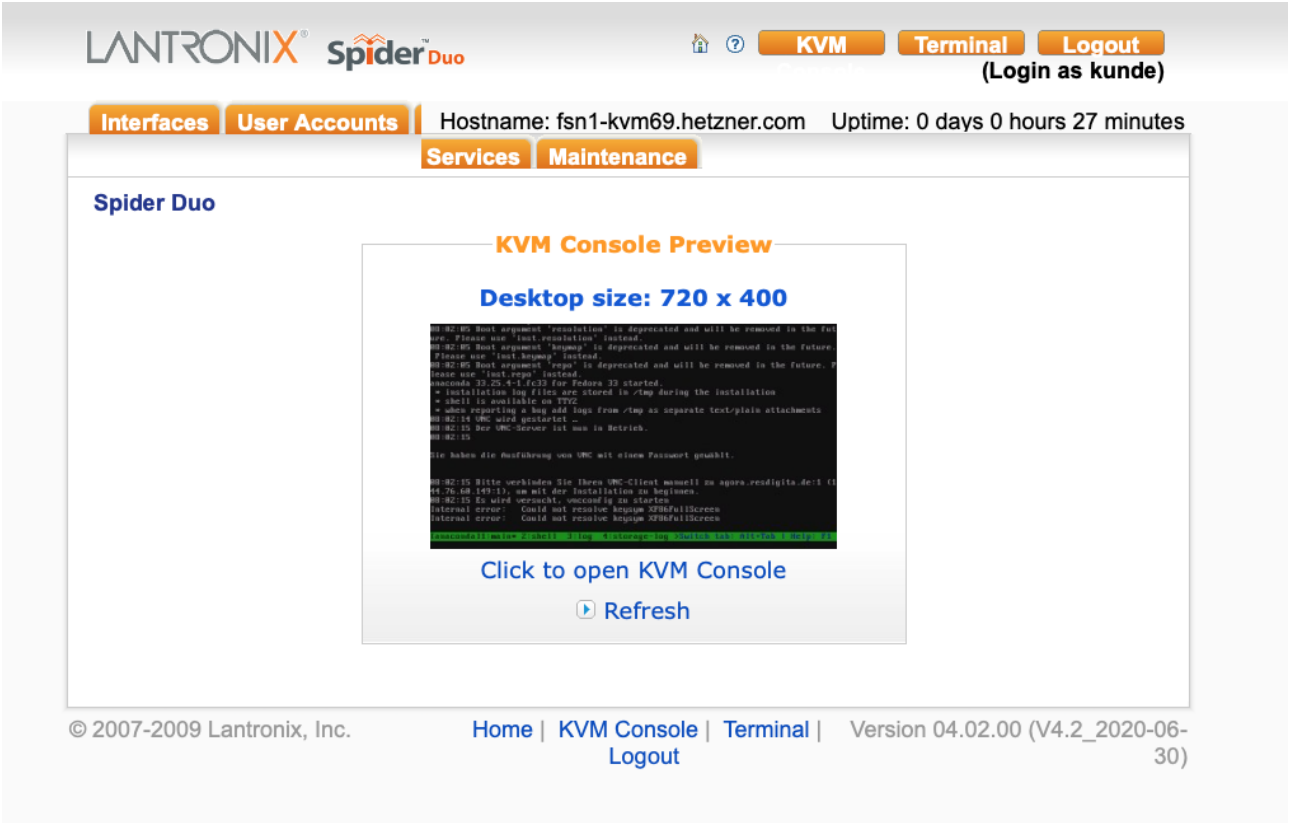
3.2 Working with the Hetzner KVM Console

When the remote console is ready you get an email. Now you have to reboot the server to get remote console up and running. The easiest way is to use the „Reset“ service in the Web interface.



Select the hardware reset option. Wait some minutes for reboot to complete and then open the address provided in the email. You should be greeted by the login screen.

Once the KVM console is active, you will see the familiar Fedora Installation messages.

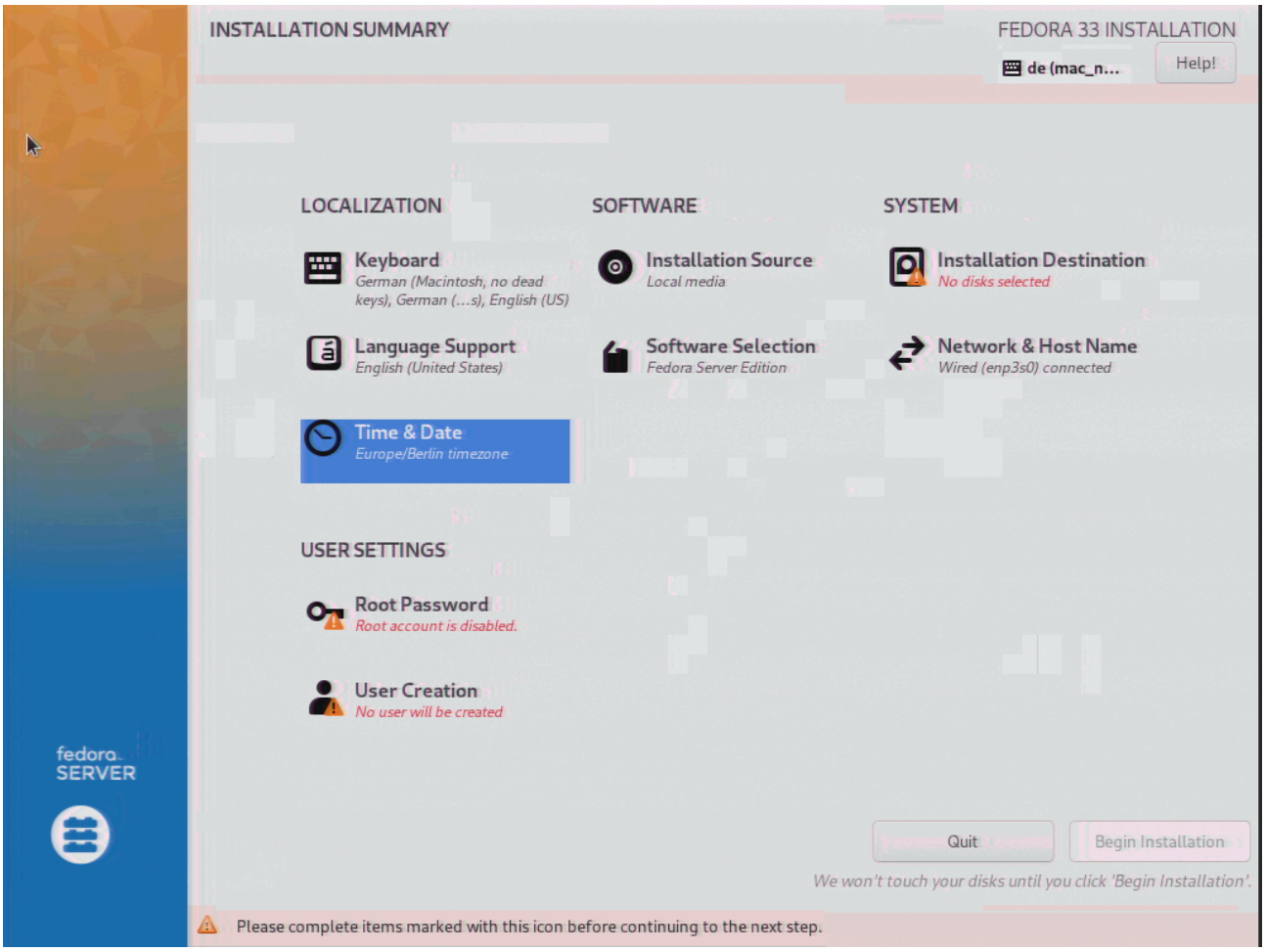


What exactly is displayed after login depends on how long the server has been running. You may still see the boot messages of the firmware, or Fedora's boot message for checking the installation medium, or Anaconda is already waiting with the language selection.

Clicking in the window opens the console that fills the viewport.

4. Basic Installation

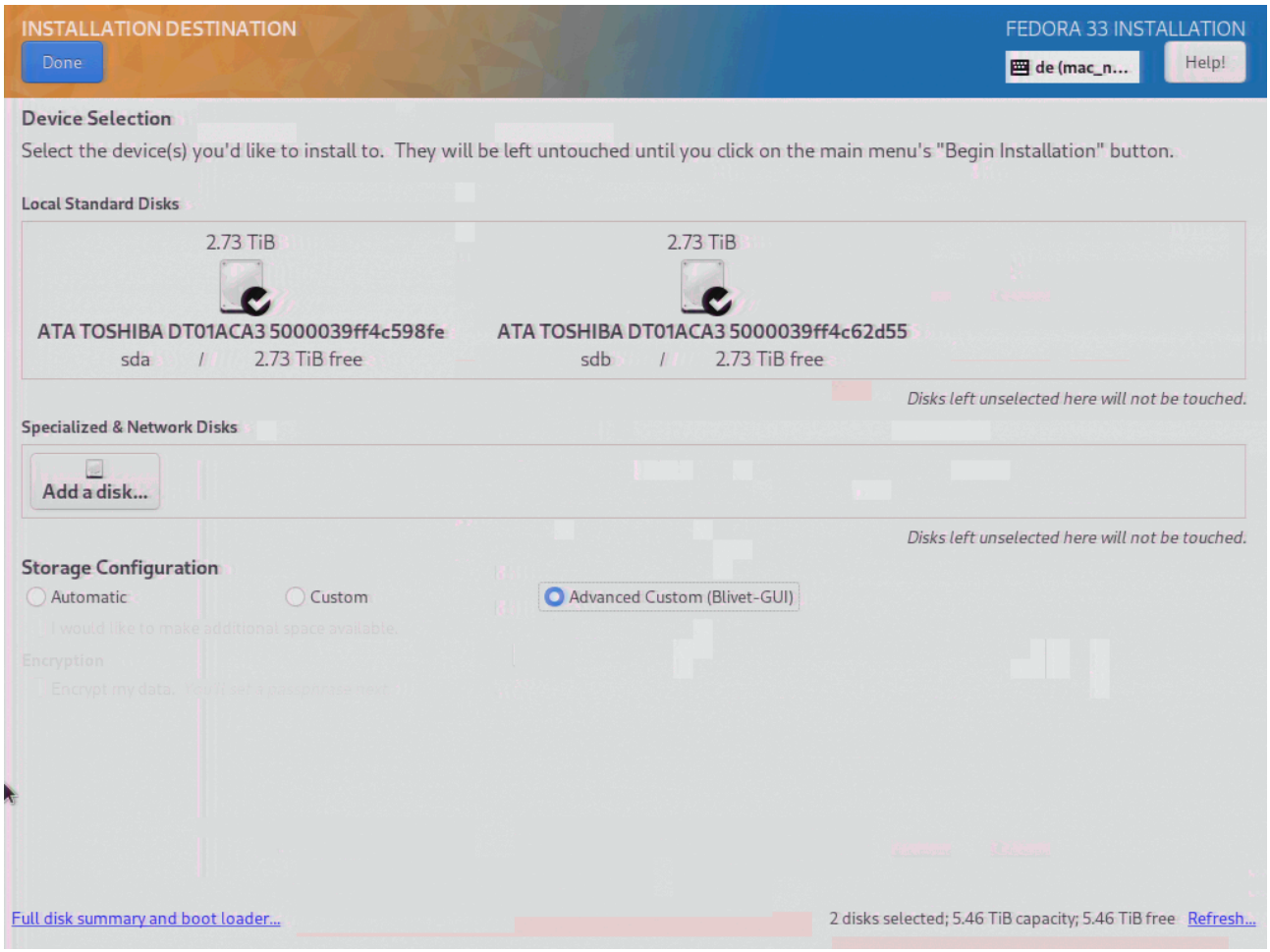
After selecting the language, you will be taken to the familiar Anaconda summary.



Most of it is already correctly preset.

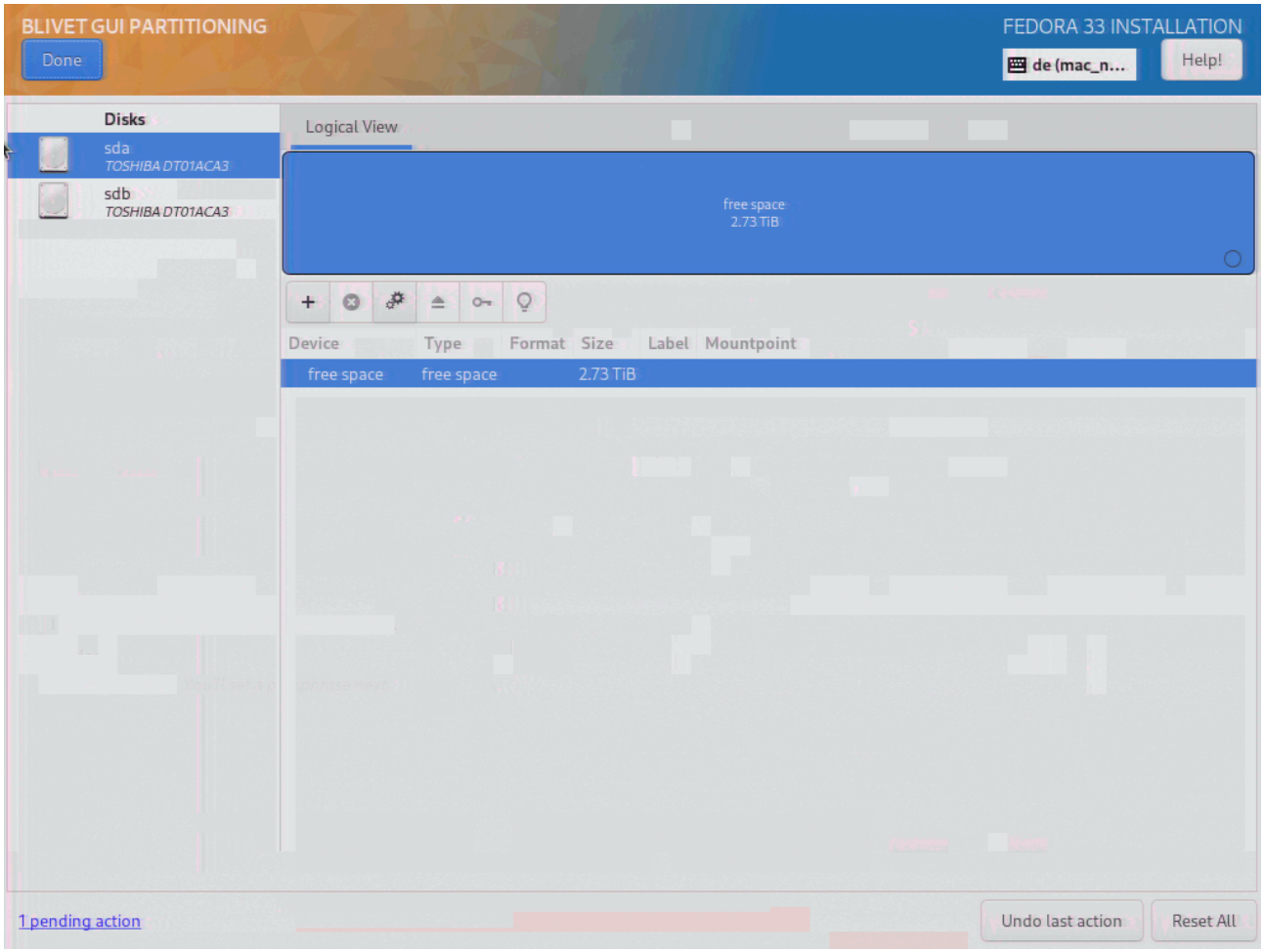
It is nevertheless useful to briefly check the items "Keyboard Layout", "Language Support", "Software Selection" (Fedora Server without additional options), and "Time & Date" (Check that an NTP server is configured as well). Don’t modify „Installation Source“. The remaining items require more intensive work.

Selecting the item „Installation Destination“, a selection of the disks and the configuration options are displayed.



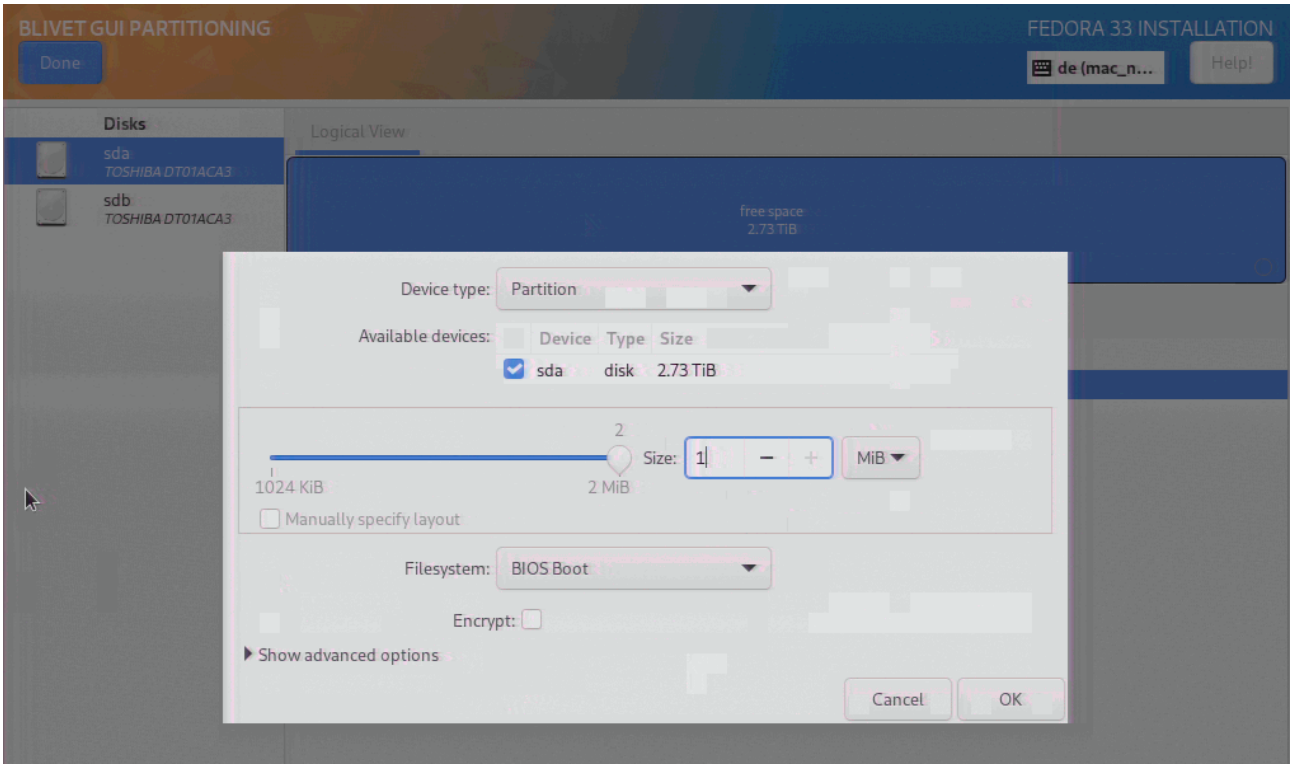
Both installed hard disks must be selected. For configuration, „Advanced Custom (Blivet-GUI)" has to be selected. In most cases, either "Automatic" or "Custom" is preset. With neither of these options a software raid with hard disks larger than 2 TB can be set up correctly! An installation will remain not fully functional and there may be problems with booting.

Once all the details have been entered correctly, clicking on "Done" takes you to the editing screen.



A server from the "Server Exchange" usually operates as a Bios system (and not as UEFI). If such a system is to use hard disks larger than 2 TB, a GPT partition table will be generated. It requires a small BIOSBoot partition for the Grub boot loader at the beginning. The disks are to be mirrored in RAID 1 so that the system remains operational when one disk fails. Therefore the boot sector as well as the BIOSBoot partition must be created on both disks (and this is precisely what is not possible with the simple custom method nor the default partitioning).

Usually, the first Disk sda is already selected and at the top there is a large blue bar which represents disk sda storage area. Otherwise, a click onto the left side sda symbol and then into the central blue area activates it as well as the "plus" symbol just beyond the central sda area to start the creation of a partition.

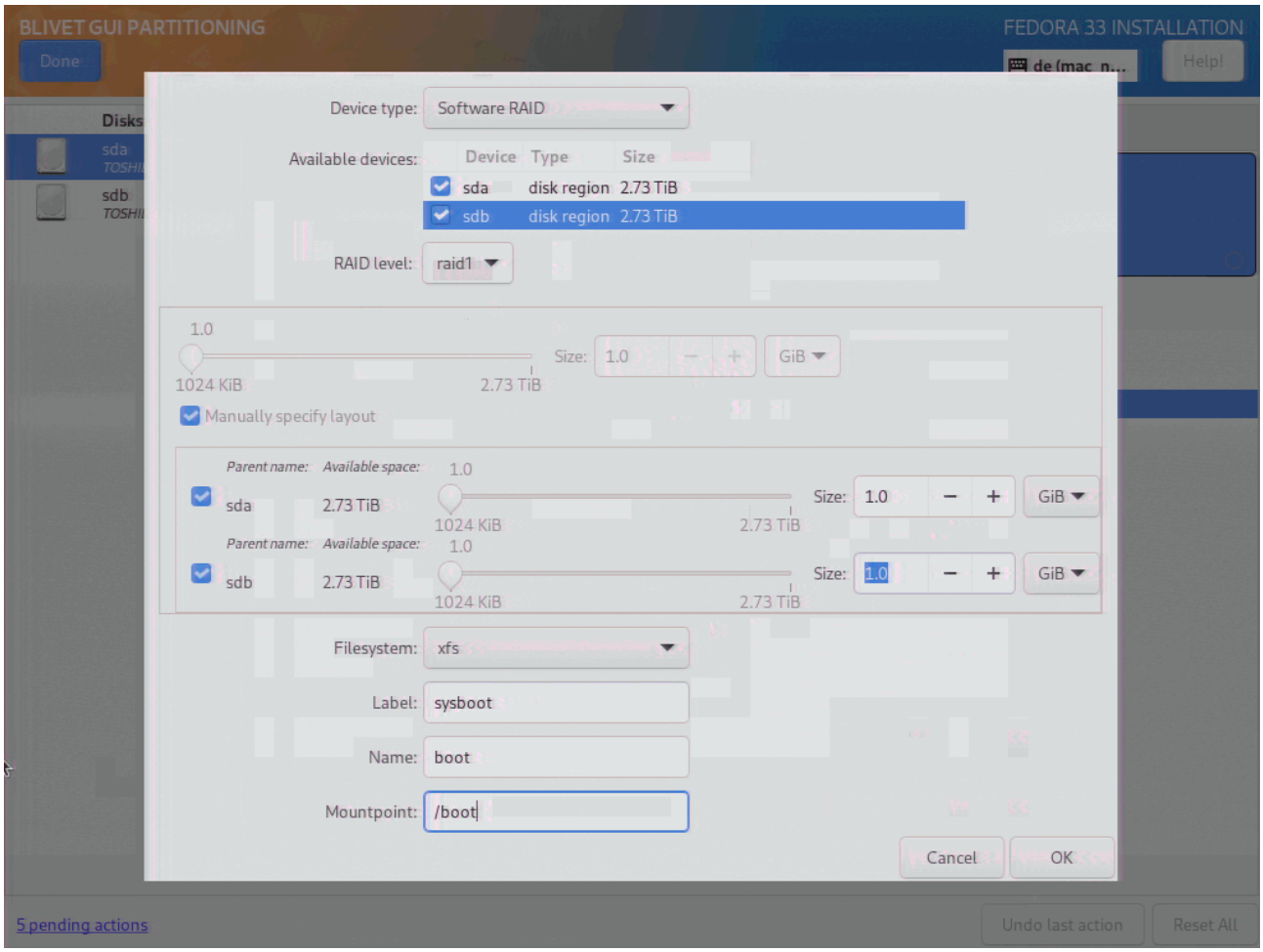


First select "Bios Boot" in the „Filesystem“ selection box and then enter 1 MiB as the size. An OK creates the partition and updates the display of the free area. 1 MiB is quite sufficient, but sometimes the dialogue insists on 2 MiB and corrects the entry accordingly. It’s OK, too.

Next, click on sdb and thus activate it for editing. Repeat the process to create a BIOSBoot partition as before.

When both BIOSBoot partitions have been created, check again whether the size and position are actually the same in both cases! You can switch between the two disk views by clicking on the sda1 and sda2 icons respectively.

In the next step the /boot partition for grub2 has to be created as a Raid 1 filesystem.

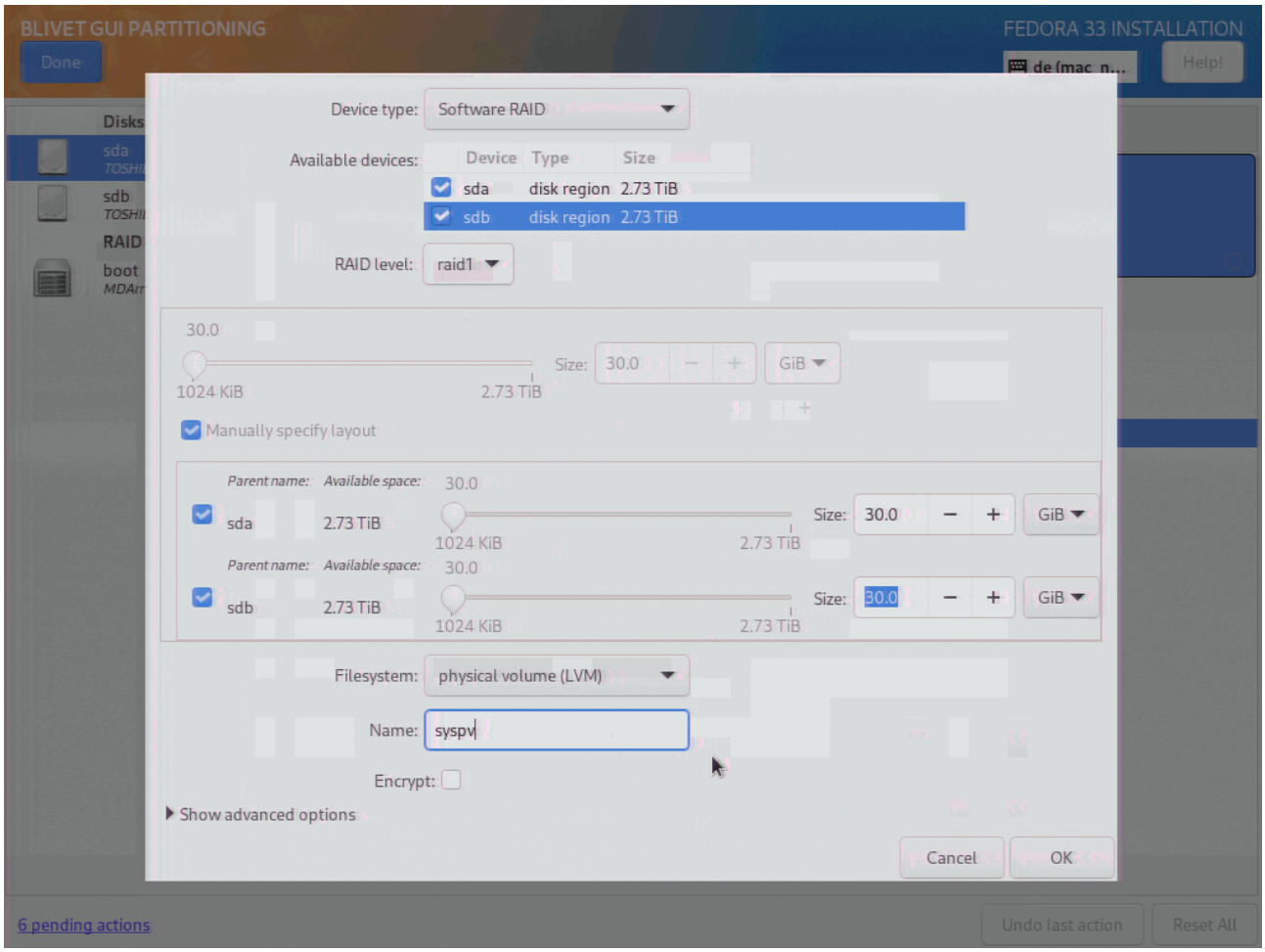


The following editing steps must be executed

- Click on sda on the left and then into the free area in the right-hand central area to activate the hard disk and release the plus icon.
- Clicking on the plus icon opens the editing form.
- Select "Software Raid" in the Device type field.
- Activate sda (already active) and sdb
- As „Raid level“ select „raid1“ (mirroring)
- Set space to 1 GiB
- Selection field „Filesystem“: Keep xfs
- Input field Label: e.g. sysboot (at will)
- Input field Name: boot (becomes part of the device name)
- Clicking OK creates the Raid Partition

The editing window closes and a new icon appears in the left column representing the file system based on Raid.

In the next step we have to create the *system* volume group. Again, click onto sda (left side) and then into the free space of the central blue sda representation. Then click on the plus sign and the



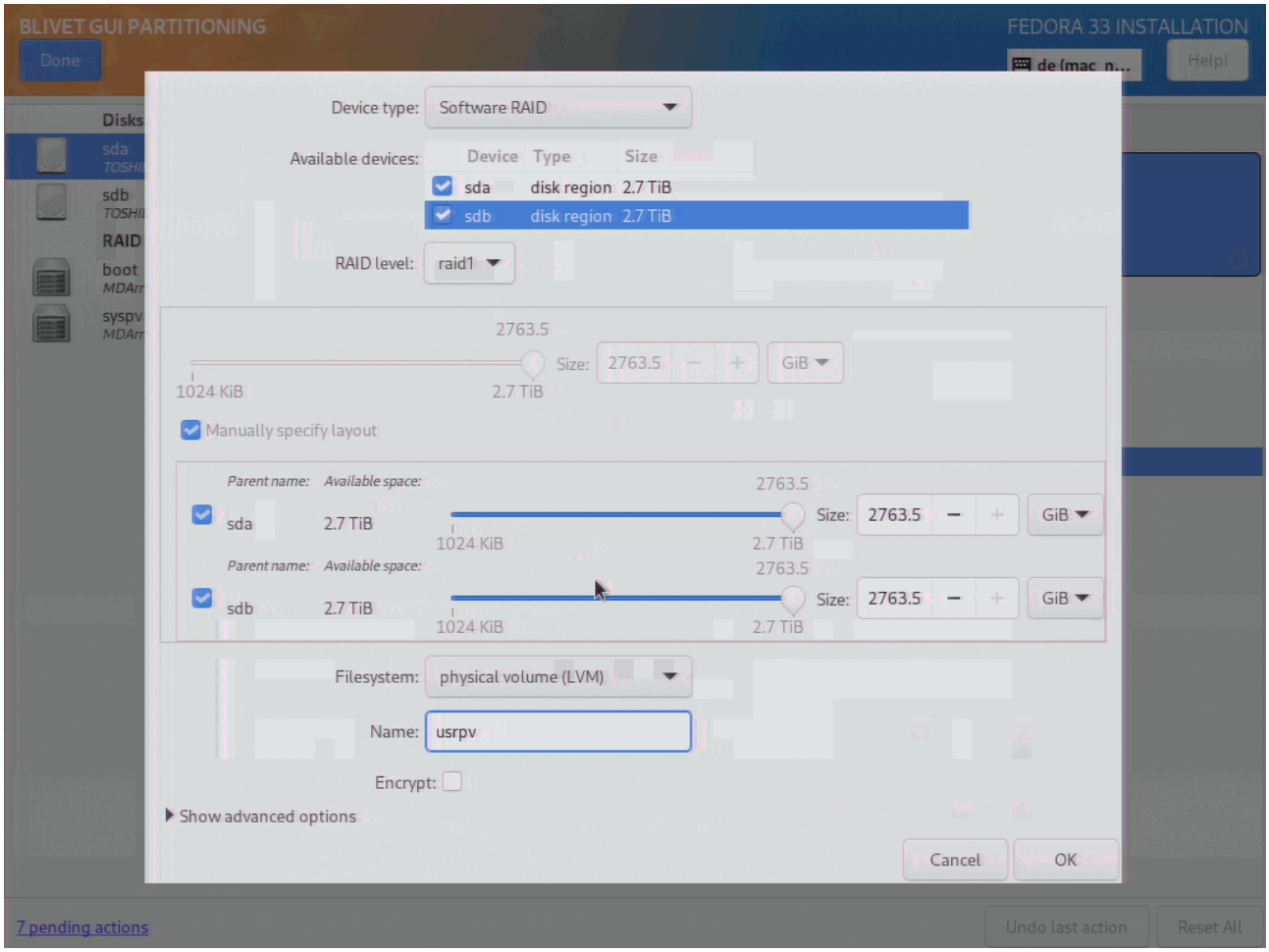
editing form opens again. Complete the following steps:

- In the „Device type“ Menu select „Software Raid“
- Activate sda (already selected) and sdb
- As „Raid level“ select raid1 (mirroring)
- Set space to 30 GiB
- As file system select „physical volume (LVM)
- As Name enter „syspv“
- Click OK to create the raid partition

In the left bar yo see a new symbol representing the syspv raid array.

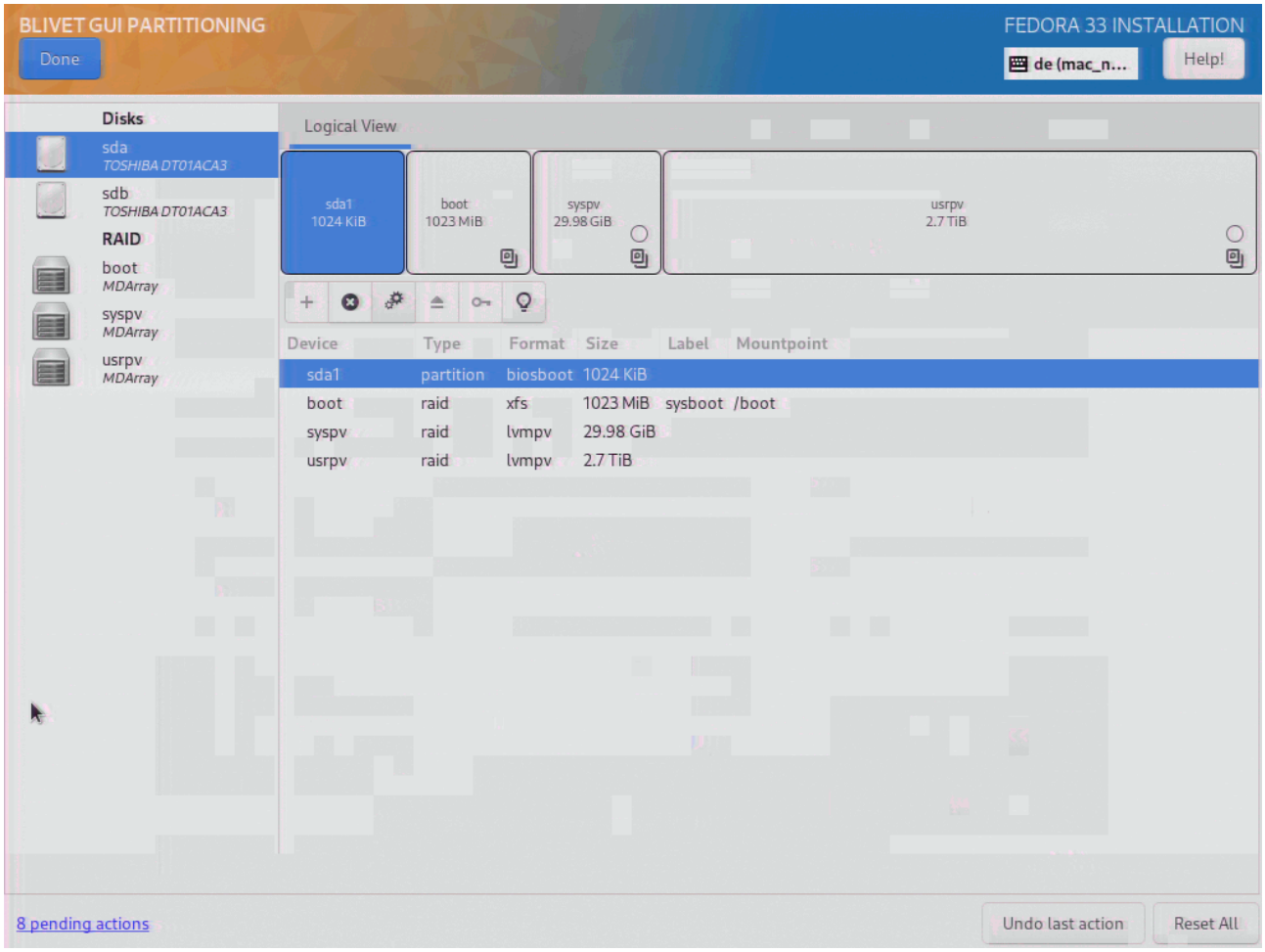
In contrast to other GUIs, a volume group (VG) is not created immediately, but a "physical volume" is created first. Many other GUIs perform this step implicitly and automatically.

The remaining available space on the hard disks is completely dedicated to user data.. Again, click into the free space of sda and create a usrpv repeating the above steps accordingly.



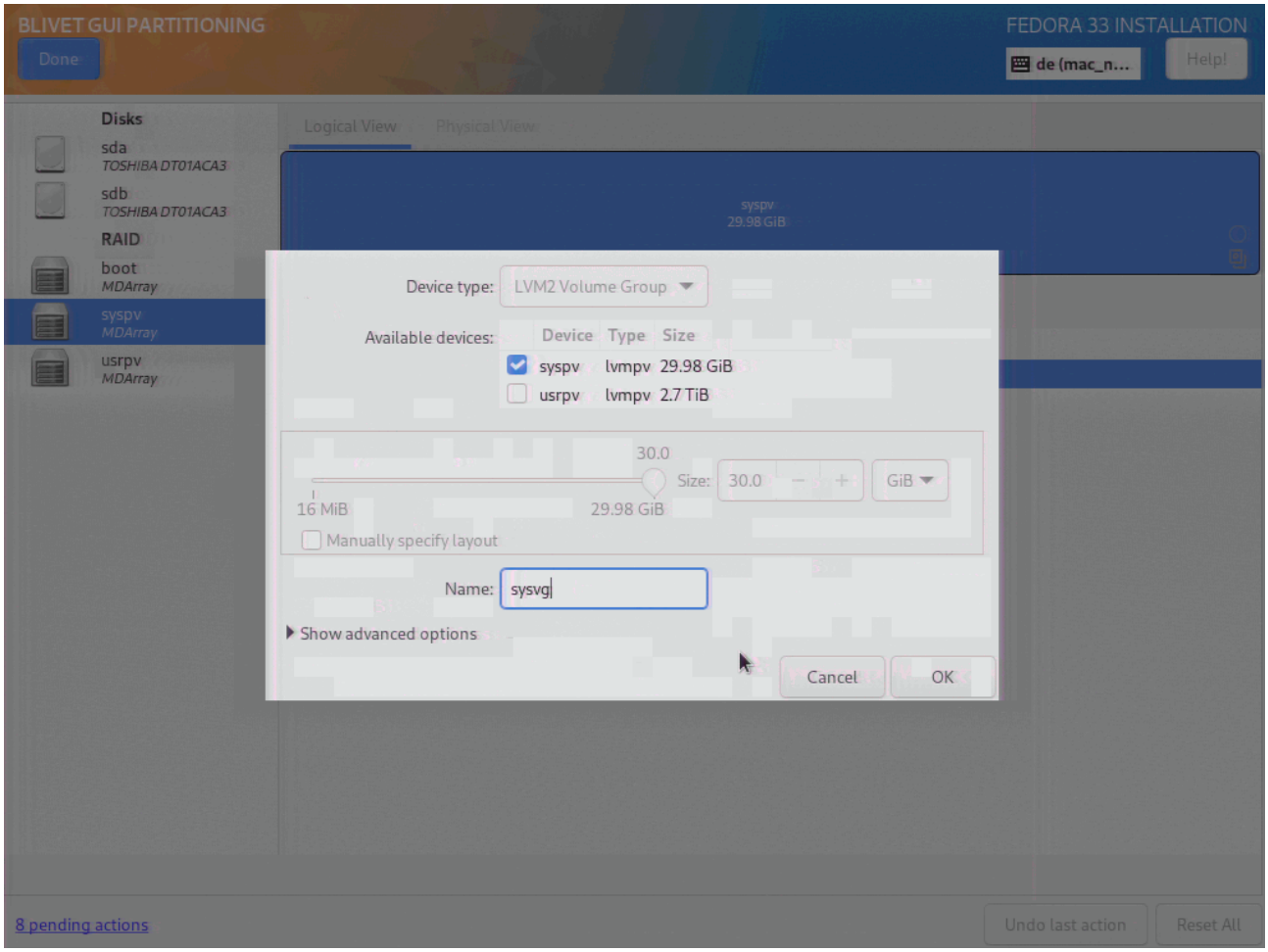
We have now created the basic structure and must finally create the volume groups and logical volumes.

On the left sice you see the symbols for the boot Raid partition and the two physical volumes below



the two hard disks.

Double-click onto the syspv icon in the left bar to activate it and then on the Plus sign to open a



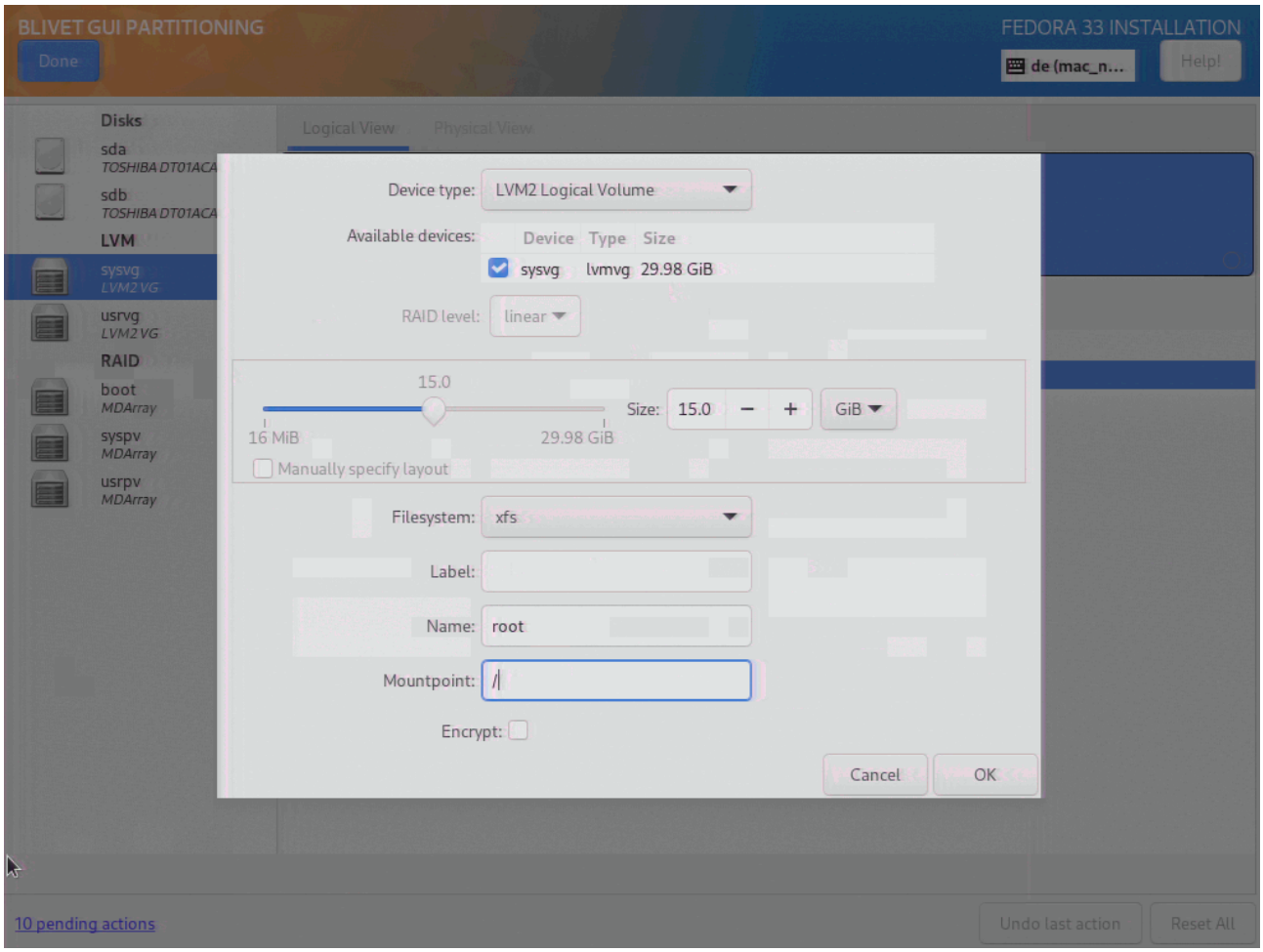
device creation form.

Select LVM2 Volume Group as device type, leave syspv active an enter sysvg as name. Click OK to create the volume group. A corresponding icon appears in the left column.

In an analogous way, the volume group usrvg is created on usrpv.

Finally, the file systems must be created.

As in the previous steps, clicking on symbol sysvg activates the unit for editing and clicking on the



free area releases the plus sign. A click on the plus sign opens the dialogue for setting up a file system.

- Device Type: LVM Logical Volume
- Keep sysvg activated
- Set a size of 15 GiB
- Keep file system xfs
- Label: sysroot
- Name: root
- Mountpoint: /

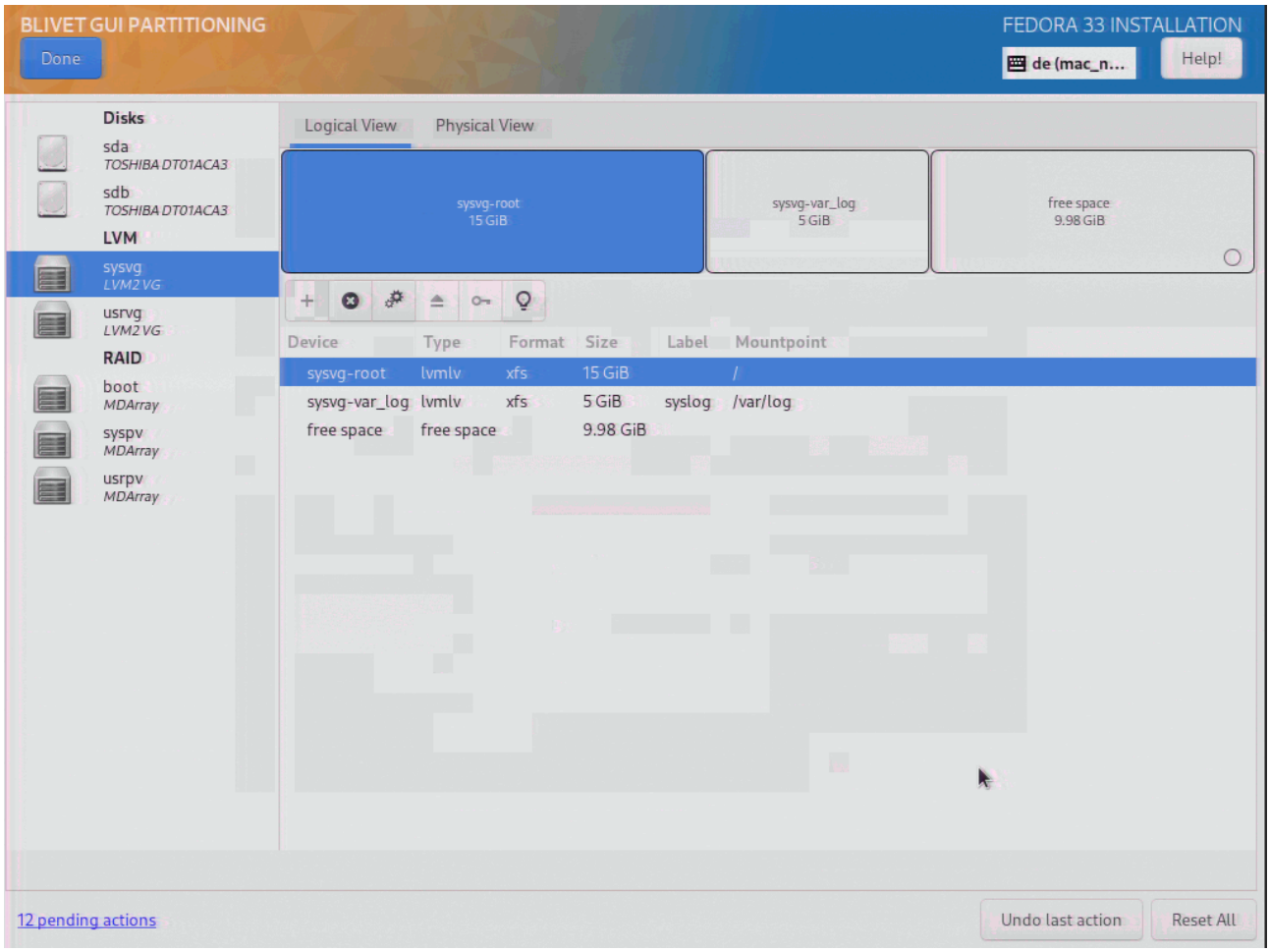
Because all user data is stored in usrvg, a size of 15 GiB is completely sufficient. In an emergency, there would still be room to expand the logical volume.

Traditionally, one would create another partition for the log files to prevent the root file system from overflowing with logs in the event of an error. This would definitely be on the safe side. The following entries would then have to be made:

- Device Type: LVM Logical Volume
- Keep sysvg activated
- Set a size of 5 GiB
- Keep file system xfs

- Label: syslog
- Name: var_log
- Mountpoint: /var/log

This completes the particularly critical step of setting up the hard drives.



Selecting "Done" in the upper left corner leads to the next step, where the setup can be completely checked again.

The next step is to configure the network connection. It is only important to insert the complete name as specified in the DNS into the "Host Name" field . An IPv4 connection is already set up via DHCP, an IPv6 connection is easier to configure later via the terminal.

Finally, the details for root and another user must be entered.

For user root, the two checkboxes for blocking root access and for allowing an ssh login via password should not be activated under any circumstances. This way, secure access via ssh key file is still an option and, in an emergency, also with a password via a KVM console.

To be able to access the server at all after completing the installation and restarting, it is essential to create another, non-privileged user. In this case, both checkboxes must be activated. This user can then obtain root privileges via sudo and perform administrative tasks.

Click on "Begin installation" to run the installation and restart the server after completion.

Do not release the KVM console yet, it may still be needed for a subsequent step.

5. Post Installation Configuration

After the installation is complete, the system is restarted. Afterwards, login via ssh (non root) as well as via the Web Admintool Cockpit is available.

1. An RSA key is required to log in as root. This should also be set up later for other accounts. If not yet available, a key must be created (on the local desktop).

   It is best to create the key in the .ssh directory of the desktop user. It should not be secured by password to enable automatic processing.

   On the local desktop
   ```
   […]# mkdir ~/.ssh
   […]# cd ~/.ssh
   […]# ssh-keygen -t rsa -b 4096  -C "root@example.com" -f id_<outputkeyfile>_rsa
   ```

2. Logging on to the server as root is not possible, instead the unprivileged administration account must be used (here hostmin).

   ```
   […]$ ssh  hostmin@example.com
   ```

   After logging in to the server, acquire root permissions

   ```
   […]$ sudo su -
   ```

5.1 Some Preliminary Work

An update must bring the system up to date. In addition, the network configuration has to be completed.

3. Immediately after rebooting, update and install a decent editor. Do not reboot yet.

   ```
   […]# dnf update
   […]# dnf install vim
   ```

4. Control of the IP addresses

   ```
   […]# ip a
   1:  lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc n
           …
   2:  enp3s0: <BROADCAST,MULTICAST,UP,LOWER
           …
   […]# nmcli con
   NAME    UUID                                  TYPE      DEVICE
   enp3s0  dabaa33b-25b0-3bfd-8a74-b6b40847a7a4  ethernet  enp3s0
   ```

   Usually, only a local address is configured (fe80::...) for IPv6 after installation. A fixed public address must be explicitly set up.

5. Set up a fixed IPv6 address.
   It is set up for the physical interface, enp3s0 in the above example. By convention, [...::2] is used for this address.

   ```
   […]# nmnmcli con mod 'enp3s0' ipv6.method manual \
     ipv6.addresses <YOUR_IPv6_PRAEFIX>::2/64 \
   ```

```
  ipv6.gateway fe80::1 \
  ipv6.dns "2a01:4f8:0:1::add:1010 2a01:4f8:0:1::add:9999"
[…]# nmnmcli con up <NAME>
[…]# nmnmcli con reload
```

Check on the local desktop if a ping6 works:

```
[…]# ping6 <YOUR_IPv6_PRAEFIX>::2
[…]# # e.g. ping6  2a01:4f8:191:6494::2
```

6. Optional: Static reconfiguration of IPv4

```
[…]# nmnmcli con mod 'enp3s0' ipv4.method manual \
 ipv4.addresses <YOUR_IPv4>/27 \
 ipv4.gateway <GATEWAY> \
 ipv6.dns "213.133.98.98 213.133.99.99 213.133.100.100"
[…]# nmnmcli con up <NAME>
[…]# nmnmcli con reload
```

Check from the local desktop if a ping6 works:

```
[…]# ping <YOUR_IPv4>
```

7. The NetworkManager configuration file must contain the entries made above:

```
[…]# less /etc/NetworkManager/system-connections/enp3s0.nmconnection
```

8. Perform a reboot. If the server is then accessible via ssh with IPv4 and IPv6, return the KVM console.

```
[…]# reboot
```

5.2 Set up login for root via key file

9. Create a directory .ssh on the server in /root/ and transfer the created key. As a login from outside via sftp as root is not possible, the easiest way to do this is via copy&paste.

Execute on the server:

```
[…]# mkdir /root/.ssh
[…]# cd /root/.ssh
[…]# vi /root/.ssh/authorized_keys
```

Alternatively, transfer the key via sftp to the home directory of the unprivileged administration account and copy it over.

```
[…]# mkdir /root/.ssh
[…]# cd  /root/.ssh
[…]# mv /home/hostmin/id* ./
```

10. Updating the privileges of the .ssh directory

```
[…]# chown  -R  root.root  /root/.ssh
[…]# chmod 700 /root/.ssh
[…]# chmod 600 ~/.ssh/*
[…]# /sbin/restorecon  -R  -vF  /root/.ssh
```

11. Testing access as root with key

Execute from your desktop:

```
   […]# ssh -i ~/.ssh/id_<outputkeyfile>_rsa  root@example.com
```

5.3 Disable password login, allow for individual users

A password login can be prevented with little effort. This saves a lot of warning messages in the log file and makes it easier to check them. For individual users, password login is allowed again as a fallback solution in case something goes wrong with the key files.

12. Create a configuration file and fill it in as follows:

```
[…]# vi /etc/ssh/sshd_config.d/60-local.conf
# Local custimization: disable password login except for
# one (optionally add some more) user as a fallback option.
PasswordAuthentication no

Match User hostmin
 PasswordAuthentication yes
#Match User hostmin2
#       PasswordAuthentication yes
```

13. Reload the sshd daemon

```
[…]# systemctl  reload sshd
```

14. Test that everything works as expected
    - Is an authorised user able to log in?
    - Is anyone else is rejected with the message "Permission denied (publickey,gssapi-keyex,gssapi-with-mic)"?
    - If this does not work: Check whether the latest update has been installed. The file /etc/ssh/sshd_config.d/50-redhat.conf there should no include a line „PasswordAuthentication yes“ (as this is already the default and should not be repeated in order not to hinder other configurations).

5.4 Installation fail2ban

15. Installation of the software and the required Postfix

```
[…]# dnf install fail2ban postfix
```

16. Create and fill configuration file:

```
[…]# vi /etc/fail2ban/jail.local
# Jail configuration additions for local installation

# Adjust the default configuration's default values
[DEFAULT]
# Optional enter an trusted IP never to ban
#ignoreip = www.xxx.yyy.zzz/32
bantime  = 6600
backend = auto

# The main configuration file defines all services but
# deactivates them by default. We have to activate those neeeded
[sshd]
enabled = true
```

17. Activate software

```
[…]# systemctl  enable  postfix  --now
[…]# systemctl  enable  fail2ban  --now
```

18. Control in the log
    […]# tail -f /var/log/fail2ban.log

5.5 Install and configure Logwatch

19. Install:
    […]# dnf install logwatch
20. For the configuration, a real address must be entered in /etc/aliases as a forwarding for root:
    […]# vi /etc/aliases
    …
    ```
    # Person who should get root's mail
    #root:          marc
    root:           real@address.for.root
    ```

5.6 Securing Cockpit Access

21. The cockpit login should remain secured behind a firewall to prevent any brute force attempts from the outset.
    […]# firewall-cmd --permanent --remove-service=cockpit
    […]# firewall-cmd --reload

    To access Cockpit you may use a ssh tunnel, eg

    ```
    ssh host.example.com -L 9090:host.example.com:9090
    ```

    ```
    localhost:9090
    ```

    or add Cockpit service temporarily on demanded

    ```
     ssh host.example.com
    firewall-cmd –add-service=cockpit
    ```

5.7 Refining and Ensuring Various Features

22. Checking the correct hostname
    [***…]# hostnamectl***

    Set hostname if required:
    […]# hostnamectl set-hostname <FWDN>
23. Control time, time zone, time synchronisation
    […]# timedatectl

    If necessary, activate time synchronisation:
    timedatectl set-ntp true

    Correct time if necessary:
    […]# timedatectl set-time <TIME>

6. Infrastructure for Virtual Machines (libvirt)

Libvirt is the standard virtualisation method in Fedora and provides a management toolkit for KVM / QEMU. This includes a local virtual network for protected communication between the virtual guest systems with each other and with the host.

6.1 Preparation

Libvirt stores its data including the virtual disk image files for the guest systems in `/var/lib/libvirt`. According to the installation concept, these data are stored in the `usrvg` area. Therefore, before software installation create a logical volume, install a file system, and mount it at the appropriate position.

24. Creating and Mounting a libvirt Logical Volume

    The easiest way to do this is via Cockpit. Login to Cockpit according to the way you choose in 5.6.Select storage in the left side column.

    On the new page there is a column on the right called "Devices". Select `usrvg` at the bottom (just above the drives list). A new page opens with the heading "Logical volumes". Next to the title is a link "Create logical volume“.

    It opens a form with the items Name, Purpose and Size. Enter a meaningful name, e.g. libvirt and choose a suitable size. The file system used later is xfs, which cannot be shrunk. Therefore, the size should be rather sparse. A later enlargement is easy to perform.

    The created logical drive appears in the list. The arrow symbol expands the display and a "Format" selection item appears.

    Another new form opens. Specify a name for the file system and the mount point. Again, choose a meaningful name, e.g. libvirt, and enter the mount point `/var/lib/libvirt`. Check spelling twice! Leave the remaining items at the default values. Click button "format" and Cockpit executes all steps in one go: Format the file system, create directory to mount at, and mount the file system permanently. Everything very comfortable.

6.2 Installing libvirt Virtualisation Software

25. Installing the libvirt software
    […]# dnf install qemu-kvm libvirt virt-install cockpit-machines libguestfs-tools

    Package `libguestfs-tools` provides various useful tools to maintain virtual disks. Avoid to install the group @virtualisation onto a Fedora Server. It includes various graphical programs and libraries that are not usable on headless servers.

26. Check the SELinux labels and activate and start libvirtd
    […]# ls -alZ /var/lib/libvirt
    just if needed: ***[…]# /sbin/restorecon -R -vF /var/lib/libvirt***
    […]# systemctl enable libvirtd
    […]# systemctl start libvirtd

27. Control of the functionality of the internal virtual bridge and firewall zones

    By default, libvirt creates a bridge with an interface `virbr0`, the IP `192.168.122.1` and the internal name `default`. In addition, a separate firewall zone `libvirt` is set up.
    […]# ip a
    […]# firewall-cmd –get-active-zones
    ```
    FedoraServer
      interfaces: enp3s0
    ```

```
libvirt
  interfaces: virbr0
```

6.3 Adjusting libvirt Configuration

According to the installation concept, the internal network should be used for the internal, protected communication of the VMs with each other and with the host. For this purpose, it is advantageous to set up a DNS for the internal network so that the VMs can address each other and the host by their names.

For this purpose, a domain name must also be defined for the internal network. A top-level ".local" is explicitly not recommended, nor is an official top-level name. For example, you can take the official domain name and replace the top-level domain with 'lan' or 'internal'. The example domain `example.com` then becomes `example.lan`. This is done in the configuration file via the parameter `<domain name='example.lan'/>`.

28. Configuring libvirt DNS

    Modify the configuration of the internal network along the lines below. Adjust your domain name accordingly! Leave mac address and UUID untouched! The <dns> tag activates DNS for the addresses assigned via DHCP. Don’t include a forwarder address for external addresses. It will break the VMs split-dns capability. The host itself is entered statically. Furthermore, the mtu can be adjusted at the same time. See: https://libvirt.org/formatnetwork.html.

```
[…]# virsh  net-edit  default
<network>
  <name>default</name>
  <uuid>aaaaaaaa-bbbb-cccc-dddd-eeeeeeeeeeee</uuid>
  <bridge name='virbr0' stp='on' delay='0'/>
  <mac address='52:54:00:xx:yy:zz'/>
  <forward mode='nat'/>
  <mtu size='8000'/>
  <domain name='example.lan'/>
  <dns forwardPlainNames='no'>
    <forwarder domain='example.lan' />
    <host ip='192.168.122.1'>
      <hostname>host</hostname>
      <hostname>host.example.lan</hostname>
    </host>
  </dns>
  <ip address='192.168.122.1' netmask='255.255.255.0'>
    <dhcp>
      <range start='192.168.122.2' end='192.168.122.254'/>
    </dhcp>
  </ip>
</network>
```

29. Converting the Host DNS Configuration

    Local DNS systemd-resolved introduced with Fedora 33 does not yet work with libvirt network configuration, so DNS-split for forwarding internal name queries to libvirt DNS does not work. Instead, NetworkManager's dnsmasq plugin must be enabled and systemd-resolved disabled.

```
[…]# vim /etc/NetworkManager/conf.d/00-use-dnsmasq.conf
# /etc/NetworkManager/conf.d/00-use-dnsmasq.conf
#
# This enabled the dnsmasq plugin.
[main]
dns=dnsmasq
```

```
[…]# vim /etc/NetworkManager/dnsmasq.d/00-example-lan.conf
# /etc/NetworkManager/dnsmasq.d/00-example-lan.conf
#
# This file directs dnsmasq to forward any request to resolve
# names under the .example.lan domain to 192.168.122.1, the
# local libvirt DNS server.
server=/example.lan/192.168.122.1
```

```
[…]# rm /etc/resolv.conf
```

30. Activate the modified configuration

```
[…]# virsh net-destroy default
[…]# virsh net-start default
[…]# systemctl  stop  systemd-resolved
[…]# systemctl  disable  systemd-resolved
[…]# rm /etc/resolv.conf
[…]# nmcli con mod enp3s0 ipv4.dns-search 'example.lan'
[…]# nmcli con mod enp3s0 ipv6.dns-search 'example.lan'
[…]# systemctl  restart  NetworkManager
```

31. Check the functionality of the name resolution with internal and external addresses:

```
[…]# ping host
[…]# ping host.example.lan
[…]# ping host.example.com
[…]# ping guardian.co.uk
```

6.4 Setting Up a Brouter Bridge for Public Network Access

It is a peculiarity of the Hetzner infrastructure that all IP addresses assigned to a server are routed to its physical Ethernet interface, i.e. its MAC address. For IPv4, it is possible to define additional MAC addresses for this interface, but not for IPv6. If virtual machines are to be available with both address families, which is the standard today, a conventional bridge (which works with MAC addresses) is not sufficient. The bridge must route based on the IP address (bridge in routing mode or brouter). The configuration is a bit more complex.

32. Check the Forwarding Configuration:

```
[…]# cat /proc/sys/net/ipv4/ip_forward
[…]# cat /proc/sys/net/ipv6/conf/default/forwarding
```

In both cases, a value of 1 must be output. Libvirt will activate IPv4 forwarding, but probably not IPv6. If necessary, activate forwarding temporarily:

```
[…]# echo 1 > /proc/sys/net/ipv4/ip_forward
[…]# echo 1 > /proc/sys/net/ipv6/conf/all/forwarding
```

For permanent setup, the following file must be set up:

```
[…]# vim /etc/sysctl.d/50-enable-forwarding.conf
# local customizations
#
# enable forwarding for dual stack
```

```
net.ipv4.ip_forwarding=1
net.ipv6.conf.all.forwarding=1
```

33. Create a new zone brouter that allows forwarding.

```
[…]# vim /etc/firewalld/zones/brouter.xml
<?xml version="1.0" encoding="utf-8"?>
<zone target="ACCEPT">
    <short>brouter</short>

    <description>
        The default policy of "ACCEPT" allows all packets to/from
        interfaces in the zone to be forwarded, while the (∗low priority∗)
        reject rule blocks any traffic destined for the host, except those
        services explicitly listed (that list can be modified as required
        by the local admin). This zone is intended to be used only by
        libvirt virtual networks - libvirt will add the bridge devices for
        all new virtual networks to this zone by default.
    </description>

</zone>
```

34. Check the existing interfaces

```
[…]# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
 …
2: enp3s0: <BROADCAST,MULTICAST,UP,LOWER_UP> …  state UP group default qlen 1000
 …
 inet6 2a01:xxx:yyy:zzzz::2/64 scope global noprefixroute
 …
3: virbr0: <BROADCAST,MULTICAST,UP,LOWER_UP> …  state UP group default qlen 1000
   …
```

As the listing shows, the external IPv6 subnet is a full /64 network. This must be changed to trigger IPv6 forwarding..

35. ✓ Modify the IPv6 Subnet to trigger forwarding to routing bridge

```
[…]# nmcli con mod enp3s0  ipv6.addresses '2a01:xxx:yyy:zzzz::2/128'
```

36. Creating a routing bridge
The (public) bridge is named vbr3s0 in reference to the corresponding public inferface. The IP addresses are the same as of the external interface, but different network prefix! For IPv4 we use peer-to-peer connectivity (hence /32 prefix)

```
[…]# nmcli con add  con-name vbr3s0  ifname vbr3s0  type bridge  stp off
ipv4.method manual  ipv4.addresses 'xx.yy.zz.ww/32'  ipv6.method manual
ipv6.addresses '2a01:xxx:yyy:zzzz::2/64'   ipv6.addr-gen-mode eui64
connection.zone brouter
```

37. Add IPv4 routing information, replace placeholder by the additional IPs

```
[…]# nmcli con mod vbr3s0 +ipv4.routes "xx.yy.zz.uu/32"
[…]# nmcli con mod vbr3s0 +ipv4.routes "xx.yy.zz.vv/32"
```

38. Reboot to activate the modified Network Configuration

```
[…]# reboot
```

7. Creating a VM Using Fedora Cloud Base Image

Since Fedora 33 virt-install provides a new option --cloud-init. It adds the capability to simulate a nocloud datasource iso using user provided information via two simple config files. It drastically simplifies and speeds up the process of installing a VM by using a pre-installed, ready-to-run Cloud Base Image. A base image incorporates a generic preconfiguration, as opposed to system-specific preconfigurations, e.g. for Vagrant or a specific Cloud Platform. Most distributions offer such an image. We use a Fedora Cloud Base Image.

39. If not already done download the Fedora Cloud Base Image. In the libvirt directory structure, it is stored in the subdirectory `boot`.

    ```
    […]# wget https://download.fedoraproject.org/pub/fedora/linux/releases/33/Cloud/
    x86_64/images/Fedora-Cloud-Base-33-1.2.x86_64.qcow2  -O
    /var/lib/libvirt/boot/Fedora-Cloud-Base-33-1.2.x86_64.qcow2
    […]# wget https://getfedora.org/static/checksums/Fedora-Cloud-33-1.2-x86_64-
    CHECKSUM -O /var/lib/libvirt/boot/Fedora-Cloud-33-1.2-x86_64-CHECKSUM
    […]# cd  /var/lib/libvirt/boot
    […]# sha256sum --ignore-missing -c /var/lib/libvirt/boot/Fedora-Cloud-33-1.2-
    x86_64-CHECKSUM
    ```

    The check should result in one `OK` (and several `Not Found` if you left off the parameter).

40. The image is used directly in the virtual machine to be created. According to the libvirt directory structure, the virtual disk images are located in the `images` directory.

    ```
    […]# cp  /var/lib/libvirt/boot/Fedora-Cloud-Base-33-1.2.x86_64.qcow2 \
             /var/lib/libvirt/images/<my-vm>.qcow2
    ```

41. The maximum size of the virtual disc is just under 5GB. For more space adjust to the expected size. You can resize the virtual disk later, too. Therefore, there is no reason to plan too generously in terms of size now. Resizing does not affect the current size. It is dynamically adjusted as needed up to the maximum specified.

    ```
    […]# qemu-img  resize  /var/lib/libvirt/images/<my-vm>.qcow2  10G
    […]# qemu-img  info    /var/lib/libvirt/images/<my-vm>.qcow2
    ```

42. ✓ The first configuration file to prepare refers to the subparameter meta-data, therefor named base-meta-data. It contains runtime environment data. In a nocloud environment there is just one mandatory parameter whose value you can freely.

    The systematic location for this file is the libvirt directory `boot` and therein in a subdirectory `cloud-init`. Of course, any other location is possible.

    ```
    […]# mkdir /var/lib/libvirt/boot/cloud-init
    […]# vim /var/lib/libvirt/boot/cloud-init/base-meta-data
    instance-id: base-vm
    ```

43. The other configuration file to prepare refers to the subparameter user-data, therefor named base-user-data. It contains user or installation specific data and holds the main configuration work in a nocloud environment. The tool is very powerful and it can do all the post-installation work that was previously done manually following Anaconda. Detailed explanations are in comment lines.

    ```
    […]# vim /var/lib/libvirt/boot/cloud-init/base-user-data
    #cloud-config
    ```

```
# First line is a mandatory kind of sheban

# (1) setting hostname
preserve_hostname: False
hostname: base
fqdn: base.example.com

# (2) set up root and a fallback account (here hostmin) including
# rsa key copied from public key file into this file. Account
# hostmin is permitted to login via password just in case a
# key file login is not feasible
users:
    - name: root
      ssh-authorized-keys:
        - ssh-rsa AAAAB3NzaC1yc2EAAAADAQA...jSMt9rC4uKDPR8whgw==

    - name: hostmin
      groups: users,wheel
      ssh_pwauth: True
      ssh-authorized-keys:
        - ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAAIAQDix...Mt9rC4uKDPR8whgw==

# (3) Generate a one-time password for both accounts to enable
# an initial login
chpasswd:
  list: |
    root:myPassword
    hostmin:topSecret
  expire: True

# (4) install additional required packages
packages:
  # cloud image doesn't install a firewall by default
  - firewalld
  # further protect access by fail2ban, postfix is a
  # dependency
  - postfix
  - fail2ban
  # need a full featured editor
  - vim
  # any server may need certificates nowadays
  - letsencrypt

# (5) some packages need additional configuration files
write_files:
  # configure fail2ban
  - path: /etc/fail2ban/jail.local
    content: |
      # /etc/fail2ban/jail.local
      # Jail configuration additions for local installation

      # Adjust the default configuration's default values
      [DEFAULT]
      ##ignoreip = <your save network>/24  <your save permanent IP>/32
      bantime  = 6600
      backend = auto

      # The main configuration file defines all services but
      # deactivates them by default. We have to activate those neeeded
      [sshd]
      enabled = true

# (6) perform a package upgrade
package_upgrade: true
```

```
# (7) several configuration commands are executed on first boot
runcmd:
  # (a) static configuration of the external interface, peer-to-peer
  # connection to brouter. By default a dhcp configuration is generated
  - nmcli con mod path 1  ipv4.method static ipv4.addresses '<THIS_VM_IPv4>/32'
  - nmcli con mod path 1  ipv4.gateway '<HOST_IPv4>'
  - nmcli con mod path 1  ipv4.dns '213.133.98.98 213.133.99.99'
  - nmcli con mod path 1  ipv6.method static  ipv6.addresses '<THIS_VM_IPv6>/64'
  - nmcli con mod path 1  ipv6.gateway '<HOST_IPv6>'
  - nmcli con mod path 1  ipv6.dns '2a01:4f8:0:1::add:1010 2a01:4f8:0:1::add:9999'
  # bring interface up
  - nmcli con up  path 1
  #
  # (b) persist internal interface, assign a zone as well as some
  # other adaptations.
  # By default this second interface is not persisted but configure
  # at each boort anew. Makes it impossible to assign a zone.
  # Modification results in the writing of a configuration file
  # IMPORTANT:
  # internal interface has to be specified SECOND after external!
  - nmcli con mod path 2 con-name eth1  connection.zone trusted
  - nmcli con mod path 2 con-name 'System eth1' ipv6.method disabled
  - nmcli con up  path 2
  #
  # (c) activate and configure firewall and additional services
  - systemctl  enable  firewalld  --now
  - systemctl  enable  fail2ban  --now
  #
  # (d) try to grow partition and filesystem just in case disk was enlarged
  - growpart  /dev/vda  1
  - resize2fs -p /dev/vda1
  #
  # (e) finally disable cloud-init and reboot
  - systemctl  disable  cloud-init
  - reboot
# done
```

44. Execute virt-install to create VM

```
[…]# virt-install   --name base \
 --memory 3072  --cpu host --vcpus 3  --graphics none \
 --os-type linux --os-variant fedora33  --import \
 --disk /var/lib/libvirt/images/base.qcow2,format=qcow2,bus=virtio  \
 --network bridge=vbr3s0,model=virtio \
 --network bridge=virbr0,model=virtio \
 --cloud-init meta-data=/var/lib/libvirt/boot/cloud-init/base-meta-data,user-
data=/var/lib/libvirt/boot/cloud-init/base-user-data
```

It takes some time, be patient. After a while a login prompt is shown. Don’t try to login immediately. After some seconds the initialization process will continue. Finally, you see a message like `[  OK  ] Finished man-db-cache-update.service. A <return>`

45. Finally, login and check the installation. Name resolution should work without further configuration

```
[…]# resolvectl domain
Global:
Link 2 (eth0): ~.
Link 3 (eth1): <YOUR_DOMAIN>.lan

[…]# resolvectl dns
Global:
```

```
Link 2 (eth0): 213.133.98.98 2a01:4f8:0:1::add:1010
Link 3 (eth1): 192.168.122.1
```

```
[…]# ping <YOUR_INTERNAL_HOST_NAME>.lan
[…]# ping <SOME_EXTERNAL_NAME>
[…]# ping6 <SOME_EXTERNAL_NAME>
```

46.