# Practice M1: Introduction to Linux and Virtualization

\* NOTE: Some of the exercises in this practice guide are unsuitable for execution in WSL or Docker environments. A virtual infrastructure should be used instead.

For this practice, we will need at least one virtual machine with **AlmaLinux OS 9.x** (or **CentOS Stream 9**, **Oracle Linux 9.x**, or **Rocky Linux 9.x**), **openSUSE Leap 15.x,** or **Debian 12.x** (or **Ubuntu Server 22.04/24.04**) installed. Of course, another version of the listed or another distribution can be used, but there can be some differences.

All commands that we are going to use in this practice will be shown in combination with a prompt. This way, it will be easier for us to understand which user and on which station we are working.

The next steps will be executed on an **AlmaLinux** machine. If there are any discrepancies between **AlmaLinux** and the other two distribution families, there will be a note stating it clearly.

## Part 1

No practice for the first part of this module.

## Part 2

During this part, we will install **VirtualBox** and create our first Linux based virtual machine

For the rest of this part, the following assumption is made – we are working on a Windows 10 based machine with an account that has administrative privileges. Our machine supports hardware virtualization and has enough free resources to accommodate two or three virtual machines running simultaneously.

### Install VirtualBox

First, we must open a browser and navigate to <https://virtualbox.org>

Then click on the big blue button **Download**. On the next screen, we must click on the **Windows hosts** option and save the installation file locally.

*There is something else we can download as well. It is the* ***VirtualBox Extension Pack****. This is NOT mandatory. The download link is labeled* ***Accept and download*** *(it is next to the individual platform packages).*

*Should you want to use an earlier version, you must scroll down and click on the* ***VirtualBox older builds*** *link.*

Now that we have the necessary files, we can navigate to the folder where we saved them and double click on the main installation file (for example, for version ***7.0.18*** *it is called* ***VirtualBox-7.0.18-162988-Win.exe***) to initiate the installation process. Now, we must follow the steps provided by the installation wizard:

* On the first screen, click **Next**;
* Check if customization of the installation is needed. For example, we can change the target directory or remove some of the components. I would suggest that we leave everything as it is and click on the **Next** button;
* Then we can choose which shortcuts to be created and if the specific file extensions will be associated with VirtualBox. Leave everything as it is and click **Next**;
* A warning is displayed that our network will be temporarily disconnected. Click on **Yes**;
* Click on **Install** to initiate the actual installation;
* If prompted to accept driver installation, confirm;
* Finally, click on **Finish**. The wizard will close, and **VirtualBox** will start;

Okay, we have **VirtualBox** up and running. As the next step, we could install the extension pack that we downloaded earlier. Please note that this is NOT mandatory.

In order to install the extension pack (you can skip this), we can either:

* Navigate to the extension pack file, double click on it, and then follow the prompts;
* Or navigate to the **File** > **Preferences** > **Extensions**. Then click on the **Adds new package** button, navigate to the downloaded extension pack file, confirm by clicking **Open**, and follow the prompts;

Whichever way you choose, you will have to do the following:

* Click **Install**;
* Scroll down and click on **I Agree**;
* Finally, click **OK**

### Download Installation Media

Our virtualization solution is ready. As the next step, we must download installation media for one or more of the Linux distributions we are going to work with. Here is a short list of URLs:

* **AlmaLinux** - <https://almalinux.org/get-almalinux/> or here <https://mirrors.almalinux.org/isos.html>
* **CentOS** - <https://www.centos.org/centos-stream/>
* **Debian** - <https://www.debian.org/distrib/>
* **Fedora Server** - <https://getfedora.org/en/server/download/>
* **openSUSE Leap** - <https://get.opensuse.org/leap/15.6/> or here <https://get.opensuse.org/>
* **Oracle Linux** - <https://yum.oracle.com/oracle-linux-downloads.html>
* **Rocky Linux** - <https://rockylinux.org/download>
* **Ubuntu Server** - <https://ubuntu.com/download/server>

### Creating a Virtual Machine

We are ready to create our first virtual machine. Let's assume that we downloaded **AlmaLinux** image named **AlmaLinux-9.5-x86\_64-minimal.iso**

In order to create a virtual machine, we must click on the **New** button or choose **Machine** > **New** option in the menu. Then we must follow the wizard.

Please note that in the recent versions of the product, the wizard starts in **Expert Mode**. In this mode, most important settings are combined in a single page/window. Should you want to use the usual approach, you must click the **Guided Mode** button.

In the next paragraphs we will assume that we are working in **Guided Mode**.

On the first screen:

* We must enter a name for the VM. We will enter **AlmaLinuxOS-Template**;
* Then, we can change the folder where the VM will be stored. Let's leave the default;
* For **Type** we must select **Linux**;
* For **Version** we must select **Red Hat (64-bit)**

We can see that the last two steps are done automatically because as the name for the VM, we entered the name a recognized Linux distribution.

Once we are okay with the name and operating system settings, we can click on **Next**.

The second screen is about memory size. There is an amount offered – **2048 MB**. It is based on the operating system type and version selected earlier. We can leave it as it is (or set it for example to **1024**) and click on **Next**.

The third screen requires selecting a hard disk for the VM. We can use an existing one or create a new disk which is the option selected by default. Leave the selection to **Create a Virtual Hard Disk Now** and adjust the size should you want to. Once done, click on the **Next** button.

You will be presented with a summary of what is going to happen.

Should you want to change something, you can click the **Back** button.

Let’s assume that we are fine and click the **Finish** button.

This ends the process of VM creation. Now we have our first empty VM ready.

### Installing a Linux Distribution

In order to install the operating system, we must attach the installation media to the VM.

With the VM in a stopped state and selected, click on the **Settings** button, or select the **Machine > Settings** option in the menu.

Here, we can change the virtual hardware of our virtual machine. Let's navigate to the **Storage** section in the list.

Now, we must select the **Empty** virtual optical device first. Then on the left side of the window, click on the small DVD-shaped button and select the **Choose Virtual Optical Disk File** option. Then if the installation media is not in the list, we must click the **Add** button and navigate to the install media downloaded earlier and click **Open**. Finally, select the desired item and confirm the changes by clicking on the **Choose** button. In order to apply the configuration change, you must click the **OK** button.

To start the VM, either click the **Start** button or select the **Machine > Start** option in the menu.

A window showing the screen of the VM will appear. When we click with the mouse inside the window, a message will appear informing us that the mouse cursor will be captured and that if we want to release it, we must use the **host key,** which by default is **Right Ctrl**. Select the **Do not show this message again** option and click on **Capture**.

*Please note that the following steps and input required will vary amongst distributions and distribution versions. In general, we must select a* ***language****,* ***installation destination****,* ***time zone****,* ***software packages to be installed****,* ***network connectivity,*** *and* ***users to be created during the installation****.*

#### Red Hat-based

##### AlmaLinux

On the screen, there are three options. Using the arrow keys, select the **Install AlmaLinux 9.5** option and hit the **Enter** key.

Once the setup program is initiated, we will see many rows of text appearing on the screen. After a while, the **Welcome Screen** of the installation wizard will appear.

On the first screen, we must select what language we want to use during the setup. Leave it with **English** selected and click **Continue**.

Next, we are present with the **Installation Summary** window. Here we will do the following:

* Click on **Date & Time**, select for region **Europe** and for city **Sofia** and click on **Done**;
* Click on **Network & Host Name**, and turn the switch in the upper right corner in **ON** position. Then in the **Host name** field, enter **almalinux.lsa.lab** and then click **Apply**. Finally, confirm with **Done**;
* If we are tight on resources, we can turn off the **KDUMP** option. In order to do it, click on **KDUMP**. Then uncheck **Enable kdump** and confirm with **Done**;
* Click on the **Installation Destination** to verify where the OS will be installed and confirm with **Done**;
* Depending on the image in use (it applies mostly to the **boot** image), you must click on the **Installation Source** option and enter a **repository URL**. You should check which is the correct one in the documentation of the selected distribution;
* Adjust the software to be installed in the **Software Selection** section (select **Minimal Install**)and confirm with **Done**;
* Click on the **Root Password** option and enter something simple for a password. For example, enter **Parolka1!**. If we used a weak password, we must hit the **Done** button twice in order for it to be accepted;
* Next, click on the **User Creation** item to create an additional user. Enter the following:
  + Enter a name (**LSA User**) in **Full name**. It can contain spaces
  + Enter or change the proposed **User name** to **lsauser**
  + Check the **Make this user administrator** option
  + Enter the password (for example **Parolka2@**) for the user in the two password fields. Again, if the password is considered weak (for example like **Password1**), you must click the **Done** button twice

Now that we have the initial configuration done, we can hit the **Begin installation** button to start the actual installation process.

If the installation process is done, before you configure the users, then a **Finish configuration** button will appear. Click on it.

Once the whole installation process is done, we must click on the **Reboot System** button to reboot the machine.

The VM will reboot, and if everything went okay, we would be prompted with a login prompt.

#### openSUSE

On the screen will appear four options. Using the arrow keys, select the **Installation** option and hit the **Enter** key.

After a while, depending on the selected image and virtual machine configuration, a screen to select the keyboard type and accept the license will appear. Just click the **Next** button.

When asked to activate the online repositories, confirm by clicking on the **Yes** button. Please note that if, due to something, the machine is not connected to the Internet, here the **No** button must be clicked instead.

Adjust the selection of the enabled repositories. It is safe to accept the defaults and click on the **Next** button.

On the **System Role** screen, select the **Server** option and click the **Next** button.

On the **Partitioning** screen, accept the proposed partitioning scheme by clicking on the **Next** button.

Next, select the appropriate time zone. For example, select **Bulgaria**. This can be done either by selecting items in the **Region** and **Time Zone** drop-down lists or by clicking on the map. Once done, click on the **Next** button.

Next, on the **Local User** screen, make sure the **Create New User** option is selected and enter the following:

* Enter a name (**LSA User**) in **Full name**. It can contain spaces;
* Enter or change the proposed **User name** to **lsauser**;
* Enter the password for the user (for example **Parolka2@**) in the two password fields

You can set the same password both for the user you are about to create and for the **root** user. Should you want to do this, make sure that the **Use this password for system administrator** item is selected. *Or if you prefer to have different passwords, uncheck the* ***Use this password for system administrator*** *and then, on the next screen, you have to enter the password for the* ***root*** *user.*

Once done, click on the **Next** button. If the password is simple, you will be asked if you are sure. Confirm by clicking on the **Yes** button.

Explore the **Installation Settings** summary window and adjust any setting that you may need to. Once done, click the **Install** button to initiate the actual installation process. Confirm by clicking the **Install** button in the dialog box.

Now, sit back and relax while watching the installation process.

Once the whole installation process is done, either click on the **OK** button or wait a few seconds for the machine to be restarted.

The VM will reboot, and if everything goes okay, we will be prompted with a login prompt.

If you forgot to unmount the install media, the installation would start again instead of the installed operating system. Just select the **Boot from Hard Disk** option, and after a while, the login prompt will appear.

#### Debian-based

##### Debian

A screen will appear asking us to select the type of installation or another action to be taken.

Select **Graphical install** and press **Enter.**

On the next screen, we must select the language to be used during the installation. Accept the defaults (**English**) by pressing **Enter** again.

Then we must select our location. By default, it is set to **United States**. Let’s change it to **Bulgaria**.

Move the selection to **other** and press **Enter.** Then select **Europe** and press **Enter**. Finally, select **Bulgaria** and press **Enter.**

Next, we must select a locale. Use the default (**United States**) and either press **Enter** or click **Continue**.

On the next screen, we must select a keyboard type. Accept the defaults by pressing **Enter** again.

As a next step, we must set the host’s name. Use the default (**Debian**) and confirm by pressing **Enter**.

Then, enter the domain name. Enter **lsa.lab** and press **Enter**.

Enter the password (in both fields) for the **root** user. Enter **Parolka1!** and confirm by clicking **Continue**.

Next, enter the full name of the regular user. Enter **LSA User** and press **Enter**.

For the account name, enter **lsauser** and press **Enter**.

Enter the password (in both fields) for the **regular user**. Enter **Parolka2@** and confirm by clicking **Continue**.

Confirm the proposed **Guided – use entire disk** partitioning method by clicking **Continue**.

Then confirm the selected disk by clicking **Continue**.

Accept the proposed partition layout (**All files in one partition**) and confirm by clicking **Continue**.

Confirm once again by clicking **Continue**.

Then change **No** to **Yes** and click **Continue**.

On the selection of an **extra media**, leave the selection to **No** and click **Continue**.

Then accept **Bulgaria** as a location to be searched for mirrors and click **Continue**.

Select a **mirror** (or accept the default) and click **Continue**.

On the **proxy** screen (or enter a proxy if needed) click **Continue**.

On the **package survey** screen accept the selection (**No**) and click **Continue**.

Next, on the **software selection** screen, **deselect** the following:

* Debian desktop environment
* GNOME

And select **SSH server**. Then click **Continue**.

Once done, answer with **Yes** on the question for the **GRUB loader** and click **Continue**.

Select the drive (**/dev/sda**) and confirm with **Continue**.

Once the whole installation process is done, click **Continue** to reboot.

The VM will reboot, and if everything goes okay, a login prompt will appear.

##### Ubuntu

A screen will appear asking us to select a language for the installation. Make sure that **English** is selected and hit the **Enter** key.

On the next screen, we must select a keyboard type. Accept the defaults by pressing **Enter** again.

Select the install type. You can choose between **Ubuntu Server** and **Ubuntu Server (minimized)**. For the sake of simplicity, pick the second (minimized) option. Then move the selection to **Done** and press **Enter**.

Then we must configure the network adapter. By default, it is set to acquire settings via **DHCP**. Accept the defaults by pressing the **Enter** key.

On the **proxy** configuration screen, press the **Enter** key again to move forward.

Then we must enter a **mirror**. Usually, the best option is selected already. Just confirm by pressing the **Enter** key.

Next is the **disk** partitioning phase. Let's accept the defaults by pressing the **Enter** key.

Then, confirm once again with the **Enter** key and when asked, select the **Continue** option.

It is time to set the machine's name and create the first user. You can use any values that you like. For example:

* **LSA User** for full user name
* **ubuntu** for a server name
* **lsauser** for user name
* **Parolka2@** for a password

Once done, select the **Done** option and press the **Enter** key.

Make sure that the **Install OpenSSH server** option is selected. This can be done by pressing the **Space** key

Then select the **Done** option and press the **Enter** key to continue.

On the final screen, you can select from the list of additional packages. Let's continue without making any changes. Select the **Done** option and press the **Enter** key to start the installation process. Sit back and relax while watching how the installation progresses.

Once the whole installation process is done, select the **Reboot Now** option and press the **Enter** key to reboot.

Press again the **Enter** key to initiate the reboot.

The VM will reboot, and if everything goes okay, a login prompt will appear.

### Export the Virtual Machine as Template

We can use one template and produce multiple VMs out of it. This way, we can save time, avoid any potential misconfiguration and automate the process eventually.

In order to export a VM as a template, it must be in power off state. It is good practice to de-attach any ISO files.

Ensure that the VM we installed earlier is switched off. Then select the VM and choose **File > Export Appliance**.

The **Export Virtual Appliance** wizard offers two modes – **Guided** and **Expert**. You can switch between modes by clicking the **Guided Mode** / **Expert Mode** button in the bottom-right corner of the window.

No matter the mode, we can accept most of the settings with their default values. The only option that we could want to change, depending on our case, is the **MAC Address Policy**. We could want or need to set it to **Strip all network adapter MAC addresses**. This way, if we create multiple VMs out of one and the same template and connect them in a common network, there won't be two identical MAC addresses and thus no issues with the network communication.

The Actual export process is initiated by clicking the **Export** button.

### Import New Machine from Template

Let's create a new VM based on the template we prepared earlier.

This wizard offers two modes as well – **Guided** and **Expert**. You can switch between modes by clicking the **Guided Mode** / **Expert Mode** button in the bottom-right corner of the window.

First, click on **File > Import Appliance**. Then click on the browse button and navigate to the template. Select it and confirm. If we are in guided mode, we must click the **Next** button. In the next step, we must adjust the new VM settings.

No matter which mode we choose, there are two important settings – the name of the VM and **MAC Address Policy**. The name must be unique. What about the MAC address policy? If we are not sure how the template was exported, it is always a good decision to set the policy to **Generate a new MAC address for all network adapters**. This way, the network adapters of the new VM will have their own unique MAC address

The actual import process is initiated by clicking on the **Import** button.

## Part 3

Let's start the virtual machine if not already started.

### Configure Connectivity of the VM

We have different means to connect to a VM depending on its network adapters and how they are attached to the virtual network medium. There are a few popular modes:

* **NAT** – has access to the Internet but is isolated from other VMs running in NAT mode;
* **Bridged** – part of the external network to which our physical network adapter is attached;
* **Internal** – no connection to the external world, but on the same network with other VMs attached to the same internal network;

By default, a network card is set to NAT mode. This means that it has access to the external world and Internet, but in order to communicate from the outside to a process running inside the VM, we must set up a port forwarding.

In order to set or modify a port forwarding rule, we must do the following (no matter if the VM is powered on or off):

* Select the VM and enter configuration mode;
* Go to the **Network** section and select the desired adapter. It is typically **Adapter 1**;
* Open the **Advanced** section and click on the **Port Forwarding** button;
* Let's assume that we want to create a rule that will allow us to establish SSH connection to the VM:
  + To add a new rule, click on the **Add** button in the top-right corner;
  + For **Name** enter something meaningful. For example, **SSH Rule**;
  + The protocol is typically set to **TCP**, but this depends on the service to which we want to connect;
  + Leave the **Host IP** empty;
  + For **Host Port** set an available port on your host (physical machine). For example, **10022**;
  + Leave the **Guest IP** empty;
  + Set the **Guest Port** to **22**. This is the default port of the SSH service;
* We would not add more rules, at least not now, so we will click on **OK**;
* As we don't want to modify other settings, we close the window by clicking on the **OK** button;

Now, we should be able to connect to the VM, assuming that there is a working SSH service from the host.

### Connect to the VM

Connecting to a machine using SSH protocol is easy, and in a Windows environment, there are many client applications we can use.

**NOTE (3.1.1):** *You should assign only available host ports and be sure to use a host port in just one rule.*

**NOTE (3.1.2):** *If your host OS is* ***macOS*** *or* ***UNIX/Linux****, then you should have the SSH client already installed.*

#### Option #1

One of the most popular options is the **PuTTY** client. You can download it from here: <https://putty.org/>

The installation process is easy and straightforward.

Now, start the application, and for the **Host Name (or IP address)** enter **localhost** or **127.0.0.1**. And for **Port** enter the **Host Port** value from the rule we created earlier.

Once ready, click **Open**.

If this is the first time you attempt to connect to this machine, you will be prompted with a dialog box asking you if you trust it or not. Confirm with **Yes**.

Then on the login screen, enter the user credentials specified during the installation and hit **Enter**.

#### Option #2

On **Windows 10 1809** or newer and **Windows Server 2019** or newer we can enable the additional feature **OpenSSH Client**. It is a command line utility that can be turned on or off. On earlier versions, we can download it as a separate package.

For both the client and server OSes we must open **Windows Settings** and then go to **Apps > Optional Features > Add a feature** and select **OpenSSH Client**.

Once we have the client installed, we can open a **Command Prompt** and type:

**ssh -p 10022 lsauser@localhost**

Where **10022** is the **Host Port** value from the rule we created earlier, and the **lsauser** is the username we specified during the installation. Yours may be different, and you should adjust them accordingly. Then hit the **Enter** key.

If this is the first time you attempt to connect to this machine, you will be asked if you trust the machine and if you want to continue with the connection. Type **yes** and hit **Enter**.

Then type the password and hit **Enter** again.

Now you should be connected over SSH to the virtual machine.

### Transfer Files Between the Host and a VM

For this exercise, we will assume that we have SSH service running on the VM, and its network is either in bridged or NAT mode (with a port forwarding rule defined).

Again, there are multiple ways, but one of the most common is to use SSH stack. It will allow us to transfer files in both directions.

#### Option #1

In the Windows environment, you can download and use the **WinSCP** application. You can download it from here: <https://winscp.net/eng/index.php>

The installation process is straightforward. Its usage resembles a combination of **PuTTY** (for the connection part) and old-style explorer application.

#### Option #2

If we installed the SSH Client, there is one additional tool in the package – **scp**.

Open a **Command Prompt** and type:

**scp -P 10022 lsauser@localhost:/home/lsauser/.bash\_history .**

Where **10022** is the **Host Port** value from the rule we created earlier, and the **lsauser** is the username we specified during the installation. Yours may be different, and you should adjust them accordingly. Then hit the **Enter** key. Please note that here the **-P** option is upper case letter **p**.

This will copy the bash history file for the user to the current folder on the host machine.

If we want to transfer a file **readme.txt** from the host to the home folder of the user on the VM, this will look like this:

**scp -P 10022 readme.txt lsauser@localhost:/home/lsauser**

### Getting to Know the Console

Connect to the machine by using either the virtual machine terminal or a preferred SSH client application.

On the prompt, we can input the user and password that we created during the installation. Of course, we can use the **root** user, but it is not considered a good practice.

Now, let’s check where, in which folder, we are:

[lsauser@almalinux ~]$ **pwd**

/home/lsauser

[lsauser@almalinux ~]$

It appears that we are in our home folder. When we are there, the folder part of the prompt usually becomes ~.

Let’s check what we have here:

[lsauser@almalinux ~]$ **ls**

[lsauser@almalinux ~]$

There is nothing, or at least it appears to be this way. Of course, what we will see here on a clean new installation depends on what distribution we choose and what settings are there by default.

Even though we are not familiar with all options on the **ls** command yet, let's check if there are any hidden files and folders with:

[lsauser@almalinux ~]$ **ls -al**

total 12

drwx------. 2 lsauser lsauser 62 Apr 28 14:19 .

drwxr-xr-x. 3 root root 21 Apr 28 14:19 ..

-rw-r--r--. 1 lsauser lsauser 18 Jul 21 2020 .bash\_logout

-rw-r--r--. 1 lsauser lsauser 141 Jul 21 2020 .bash\_profile

-rw-r--r--. 1 lsauser lsauser 376 Jul 21 2020 .bashrc

[lsauser@almalinux ~]$

So, there are some files, after all. Those are considered hidden files because their name starts with the dot symbol (.). The same rule applies to folders as well.

Now, let's execute this:

[lsauser@almalinux ~]$ **ls -a**

. .. .bash\_logout .bash\_profile .bashrc

[lsauser@almalinux ~]$

We can also give arguments, not only options. For example, we can check what we have in the main (root) folder. The **/** symbol is used to state that we want to access the root of our file system:

[lsauser@almalinux ~]$ **ls -al /**

total 16

dr-xr-xr-x. 17 root root 224 Apr 28 14:18 .

dr-xr-xr-x. 17 root root 224 Apr 28 14:18 ..

lrwxrwxrwx. 1 root root 7 Nov 3 17:22 bin -> usr/bin

dr-xr-xr-x. 5 root root 4096 Apr 28 14:17 boot

drwxr-xr-x. 20 root root 3000 Apr 28 14:25 dev

drwxr-xr-x. 79 root root 8192 Apr 28 14:25 etc

drwxr-xr-x. 3 root root 21 Apr 28 14:19 home

lrwxrwxrwx. 1 root root 7 Nov 3 17:22 lib -> usr/lib

lrwxrwxrwx. 1 root root 9 Nov 3 17:22 lib64 -> usr/lib64

drwxr-xr-x. 2 root root 6 Nov 3 17:22 media

drwxr-xr-x. 2 root root 6 Nov 3 17:22 mnt

drwxr-xr-x. 2 root root 6 Nov 3 17:22 opt

dr-xr-xr-x. 108 root root 0 Apr 28 14:25 proc

**dr-xr-x---. 2 root root 114 Apr 28 14:20 root**

drwxr-xr-x. 23 root root 680 Apr 28 14:25 run

lrwxrwxrwx. 1 root root 8 Nov 3 17:22 sbin -> usr/sbin

drwxr-xr-x. 2 root root 6 Nov 3 17:22 srv

dr-xr-xr-x. 13 root root 0 Apr 28 14:25 sys

drwxrwxrwt. 7 root root 119 Apr 28 14:35 tmp

drwxr-xr-x. 12 root root 144 Apr 28 14:13 usr

drwxr-xr-x. 20 root root 278 Apr 28 14:25 var

[lsauser@almalinux ~]$

We can note that there is a folder named **/root**. This is the home folder for the root user.

Let's change the folder. For example, go to the **/etc** folder. This is the place where most of the configuration files are stored. Then we can check if indeed we changed the folder:

[lsauser@almalinux ~]$ **cd /etc**

[lsauser@almalinux etc]$ **pwd**

/etc

[lsauser@almalinux etc]$

As we can see, there is no need to execute **pwd**. The prompt reflects or shows where in the file system tree we are currently.

When we want to address all files which name starts with something no matter what and how many symbols their name contains, we can use the **\*** symbol (we will discuss this technique in detail in further modules). For example, let's ask for all files starting with **os\***:

[lsauser@almalinux etc]$ **ls os\***

os-release

[lsauser@almalinux etc]$

In the coming modules we will get to know more about such constructions.

Okay, now that we know that this file exists, let's check what it contains (the actual output may be different):

[lsauser@almalinux etc]$ **cat os-release**

NAME="AlmaLinux"

VERSION="8.6 (Sky Tiger)"

ID="almalinux"

ID\_LIKE="rhel centos fedora"

VERSION\_ID="8.6"

PLATFORM\_ID="platform:el8"

PRETTY\_NAME="AlmaLinux 8.6 (Sky Tiger)"

ANSI\_COLOR="0;34"

CPE\_NAME="cpe:/o:almalinux:almalinux:8::baseos"

HOME\_URL="https://almalinux.org/"

DOCUMENTATION\_URL="https://wiki.almalinux.org/"

BUG\_REPORT\_URL="https://bugs.almalinux.org/"

ALMALINUX\_MANTISBT\_PROJECT="AlmaLinux-8"

ALMALINUX\_MANTISBT\_PROJECT\_VERSION="8.6"

[lsauser@almalinux etc]$

It appears that this file contains detailed information about our distribution.

Similar or additional information about the distribution we can get by executing:

[lsauser@almalinux etc]$ **uname -a**

Linux almalinux.lsa.lab 4.18.0-372.9.1.el8.x86\_64 #1 SMP Tue May 10 08:57:35 EDT 2022 x86\_64 x86\_64 x86\_64 GNU/Linux

[lsauser@almalinux etc]$

Beside the information about the kernel, we can extract information about the name of the host:

[lsauser@almalinux etc]$ **hostname**

almalinux.lsa.lab

[lsauser@almalinux etc]$

Now, let’s return to our home folder:

[lsauser@almalinux etc]$ **cd**

[lsauser@almalinux ~]$ **pwd**

/home/lsauser

[lsauser@almalinux ~]$

As we can see, if we execute the **cd** command without any arguments, the result is that we are "back home". There is also a special symbol that we can use – it is again the tilde symbol - **~**

Now, let's try different approach. In general, no matter what distribution we use, there is a common set of commands that is always available. For example, for all distributions that adopted **systemd** (don't worry about this now, we will elaborate on the topic in next modules) we can change the host's name with:

[lsauser@almalinux ~]$ **hostnamectl**

Static hostname: almalinux.lsa.lab

...

[lsauser@almalinux ~]$ **sudo hostnamectl hostname jupiter.lsa.lab**

[lsauser@almalinux ~]$

We can add also the so-called pretty name with:

[lsauser@almalinux ~]$ **sudo hostnamectl hostname --pretty 'Jupiter Server'**

[lsauser@almalinux ~]$

Our prompt stays the same. Let's check some of the related configuration files:

[lsauser@almalinux ~]$ **cat /etc/hostname**

jupiter.lsa.lab

[lsauser@almalinux ~]$ **cat /etc/machine-info**

PRETTY\_HOSTNAME="Jupiter Server"

[lsauser@almalinux ~]$

In order for the changes to be reflected in the prompt, we must close the session and open a new one. So, type **logout** and log back in.

We can see that the new name is applied.

Now, we can check what date is today and what is the time now:

[lsauser@jupiter ~]$ **date**

Wed Apr 28 15:13:20 EEST 2021

[lsauser@jupiter ~]$

There is a way to modify the output of the **date** command:

[lsauser@jupiter ~]$ **date +%Y-%m-%d**

2021-04-28

[lsauser@jupiter ~]$

This way, we will receive the current date represented in **YYYY-MM-DD** format.

Should we need a calendar on the command line, we can have it easily with:

[lsauser@jupiter ~]$ **cal -3**

March 2021 April 2021 May 2021

Su Mo Tu We Th Fr Sa Su Mo Tu We Th Fr Sa Su Mo Tu We Th Fr Sa

1 2 3 4 5 6 1 2 3 1

7 8 9 10 11 12 13 4 5 6 7 8 9 10 2 3 4 5 6 7 8

14 15 16 17 18 19 20 11 12 13 14 15 16 17 9 10 11 12 13 14 15

21 22 23 24 25 26 27 18 19 20 21 22 23 24 16 17 18 19 20 21 22

28 29 30 31 25 26 27 **28** 29 30 23 24 25 26 27 28 29

30 31

[lsauser@jupiter ~]$

If we need to know since when or how long our system is operating, we can do it with:

[lsauser@jupiter ~]$ **uptime**

15:15:34 up 50 min, 1 user, load average: 0.00, 0.00, 0.00

[lsauser@jupiter ~]$

At any point we can ask for the history of executed commands:

[lsauser@jupiter ~]$ **history**

1 pwd

2 ls

3 ls -al

4 ls -a

5 ls -al /

6 cd /etc

7 pwd

8 ls os\*

9 cat os-release

10 uname -a

11 hostname

12 cd

13 pwd

14 hostnamectl

15 sudo hostnamectl hostname jupiter.lsa.lab

16 sudo hostnamectl hostname --pretty 'Jupiter Server'

17 cat /etc/hostname

18 cat /etc/machine-info

19 logout

20 date

21 date +%Y-%m-%d

22 cal -3

23 uptime

24 history

[lsauser@jupiter ~]$

If we want to end our session, we can do it with either **exit** or **logout**. Issuing any of these will close our session but will leave the machine up and running. Let's type **exit** and press the **Enter** key. Our session is closed now.

Let's log in back again and let's ask for all files (including hidden ones) in our home folder:

[lsauser@jupiter ~]$ **ls -a**

. .. .bash\_history .bash\_logout .bash\_profile .bashrc

[lsauser@jupiter ~]$

It appears that there is a special file that takes care of our history - **.bash\_history**. Let's check its contents:

[lsauser@jupiter ~]$ **cat .bash\_history**

pwd

ls

ls -al

ls -a

ls -al /

cd /etc

pwd

ls os\*

cat os-release

uname -a

hostname

cd

pwd

hostnamectl

sudo hostnamectl hostname jupiter.lsa.lab

sudo hostnamectl hostname --pretty 'Jupiter Server'

cat /etc/hostname

cat /etc/machine-info

logout

date

date +%Y-%m-%d

cal -3

uptime

history

exit

[lsauser@jupiter ~]$

As we can see the last few commands are not here and there is a perfect explanation for this. The reason is that they are kept in a buffer and are stored on the disk only when certain events occur, like session end.

### Download the History File to the Host

Now, using the theory in the **Transfer files between the host and a VM** sectionof this guide, let's download the actual history file locally.

Let's assume that we have **OpenSSH Client** installed. Then we must open a terminal and enter the following:

**scp -P 10022 lsauser@localhost:/home/lsauser/.bash\_history .**

Don't forget to adjust the port and the user according to your situation.

Now we can use a text editor of our choice to examine the history file we just copied.

### Reboot and Power Off the VM

Return to the console session established earlier. Now, let's reboot the system with this command:

[lsauser@jupiter ~]$ **reboot**

Or with this one:

[lsauser@jupiter ~]$ **sudo shutdown -r now**

**NOTE (3.2.1):** *Depending on the distribution and its version we may be forced to add* ***sudo*** *before commands like* ***reboot, halt, poweroff*** *and* ***shutdown*** *in order to be able to make them work. Please note, that this behavior can change between the versions of a distribution. For example, in* ***Ubuntu 18.10*** *there is no need to use* ***sudo*** *with* ***reboot*** *and* ***poweroff****, but in* ***Ubuntu 19.04*** *you must use it.*

**NOTE (3.2.2):***At this point it is enough to know that by using* ***sudo*** *our user receives temporary* ***super-powers*** *and it can act as if it is the* ***root*** *user.*

**NOTE (3.2.3):** *Another interesting thing is whose password we must enter when using* ***sudo****.*

*In distributions that are part of the* ***Red Hat*** *and* ***Debian*** *families, we must enter* ***our*** *password:*

*[lsauser@jupiter ~]$* ***sudo reboot***

*[sudo] password for lsauser:*

*On* ***openSUSE*** *we will be asked for the* ***root*** *password (this could be changed):*

*lsauser@jupiter:~>* ***sudo reboot***

*[sudo] password for root:*

*Execution of a command can be interrupted with the key combination of* ***Ctrl+c***

Once the machine is up again let's log in.

Open second SSH session. Then return to the first one.

Let's test the **wall** command and send a message to all sessions:

[lsauser@jupiter ~]$ **wall 'Attention! This module is ending soon :)'**

[lsauser@jupiter ~]$

Broadcast message from lsauser@jupiter.lsa.lab (pts/0) (Wed Apr 28 21:25:53 2021):

Check that the message is seen in the second session as well. Close the session and return to the first one.

If we want to power off the machine, let's say **in 8 minutes**, we can execute this command:

We can set up system shutdown in the future, this way giving the logged in users an option to finish their work and log out:

[lsauser@jupiter ~]$ **sudo shutdown -P +8**

Shutdown scheduled for Wed 2021-04-28 21:34:06 EEST, use 'shutdown -c' to cancel.

In the other session, we will see something like:

[anotheruser@jupiter ~]$

Broadcast message from root@jupiter.lsa.lab (Wed 2021-04-28 21:26:07 EEST):

The system is going down for power-off at Wed 2021-04-28 21:34:06 EEST!

...

In order to cancel the process, we must execute:

[lsauser@jupiter ~]$ **sudo shutdown -c**

Everyone will receive a notification:

[anotheruser@jupiter ~]$

Broadcast message from root@jupiter.lsa.lab (Wed 2021-04-28 21:26:37 EEST):

The system shutdown has been cancelled

...

Let's schedule the shutdown in 10 minutes:

[lsauser@jupiter ~]$ **sudo shutdown -P +10**

And try to login remotely. For this, initiate new SSH session (use the port you defined in the forwarding rule):

[HOST-OS] **ssh -p 20022 lsauser@localhost**

lsauser@localhost's password:

System is going down. Unprivileged users are not permitted to log in anymore. For technical details, see pam\_nologin(8).

Connection closed by 127.0.0.1 port 20022

[HOST-OS]

We won't be allowed to log in. And the reason is that once there are **5 minutes or less left until the shutdown**, a special file is created:

[lsauser@jupiter ~]$ **ls -al /run/nologin**

-rw-r--r--. 1 root root 121 Apr 28 21:30 /run/nologin

[lsauser@jupiter ~]$

Again, we can cancel the shutdown procedure:

[lsauser@jupiter ~]$ **sudo shutdown -c**

Or force the shutdown with either:

[lsauser@jupiter ~]$ **sudo poweroff**

Or this one:

[lsauser@jupiter ~]$ **sudo shutdown -P now**