Linux Servers
Simulation and Virtualization
Hardware Virtualization – Virtual Machines
Operating System Virtualization - Containers

SCOMRED, January 2021

Linux

Linux belongs to an ancient wide family of operating systems known as **UNIX**. The **Linux Kernel** manages all low level operating system features like processes, memory, filesystems and networking.

There's only one Linux kernel, it's free, open source and under continuous development.

Yet, to obtain a fully working machine several additional software is required, including utilities to manage and interact with the kernel. A full package including the kernel and all necessary additional software is called a **distribution**.

There are several different Linux distributions, some may not be free because they may include non-free additional software.

Even though the kernel is the same, different distributions may use different commands and configuration files. In our laboratory classes we will be using Ubuntu, it belongs to a family of distributions called Debian. Among others, another important family of distributions is RedHat.

Ubuntu Server LTS

This Linux Ubuntu distribution is free, every two years a new LTS (Long Term Support) version is made available, for LTS versions the support is guaranteed during five years, this is most important when installing a server. Currently, the latest Ubuntu LTS version is 20.04.

As with other modern Linux distributions, Ubuntu is provided through a minimal installation CD, once the installation software is started it will use the internet connection to fetch updated packages, so once installation finishes an updated system is attained.

The installation itself is rather straightforward, no difficult questions are asked. Usually some confirmations about language, keyboard layout, time zone, the definition of user to later manage the system and how to use the disk where the operating system is going to be installed.

Once the system starts for the first time, new configurations and new packages can then be deployed.

Distributions called Ubuntu Server don’t include a graphical user interface (GUI), they must be managed on command line. The absence of a GUI means CPU and memory resources, that would be used for it, are available to improve the server’s performance.

Servers

Servers are machines dedicated to provide services, most of them through the network, thus one important configuration feature is the server's IP addresses being static and mapped in the DNS system.

A huge number of internet servers run the Linux operating system, this is because it's very stable, fast and comprises many features and services that can be added.

Linux takes the most of the underling hardware, but the fact is, nowadays, most servers (Linux or not) don't actually run over real hardware.

There are some issues on running servers over real hardware, among them:

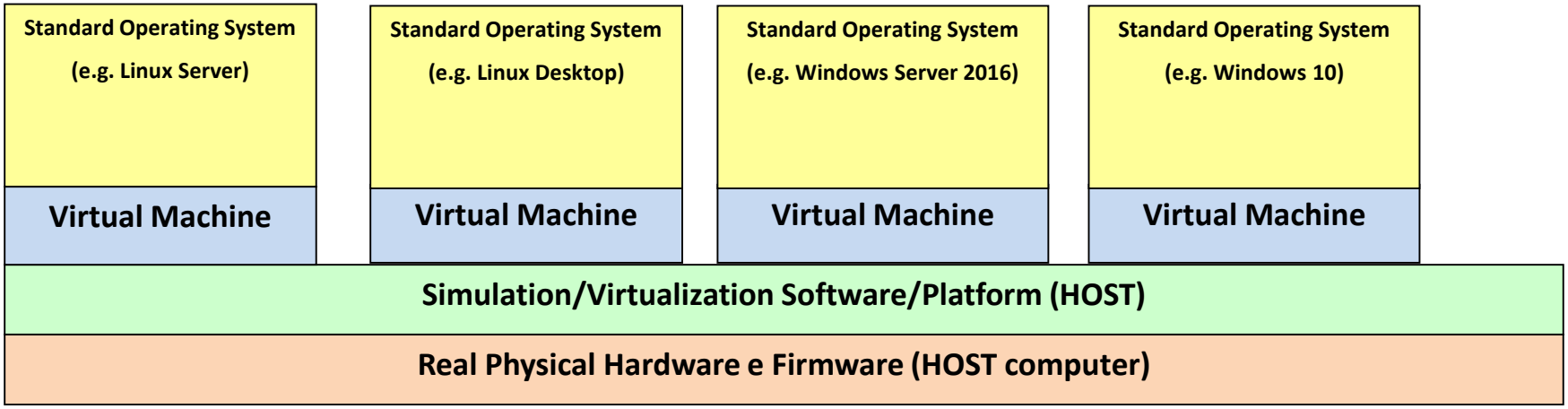
- Each hardware platform if fully and exclusively dedicated to a single server. If the server is lightly loaded then the hardware is underused.

- Many management operations require physical contact with the hardware, for instance adding or removing disks or memory.

Hardware simulation

A solution to overcome these issues is using a software platform that simulates the hardware, meaning it simulates a standard computer as operating systems expect it to be. These **software simulated** computers are known as **virtual machines**.

In a virtual machine hardware is simulated by software, for the operating system running within a virtual machine, usually called **guest operating system**, is business as usual as it can't tell the difference.



Hardware virtualization

Most hardware can be easily simulated, this is the case of external devices, including disk controllers and disks, keyboards, video displays and network interfaces, because the corresponding real devices are rather slow, software simulated versions are usually even faster than the real devices.

However, regarding the CPU and central memory (RAM), it's not that simple. Because this kind of hardware is very fast, simulating it by software is possible, but a very slow virtual machine will be the result.

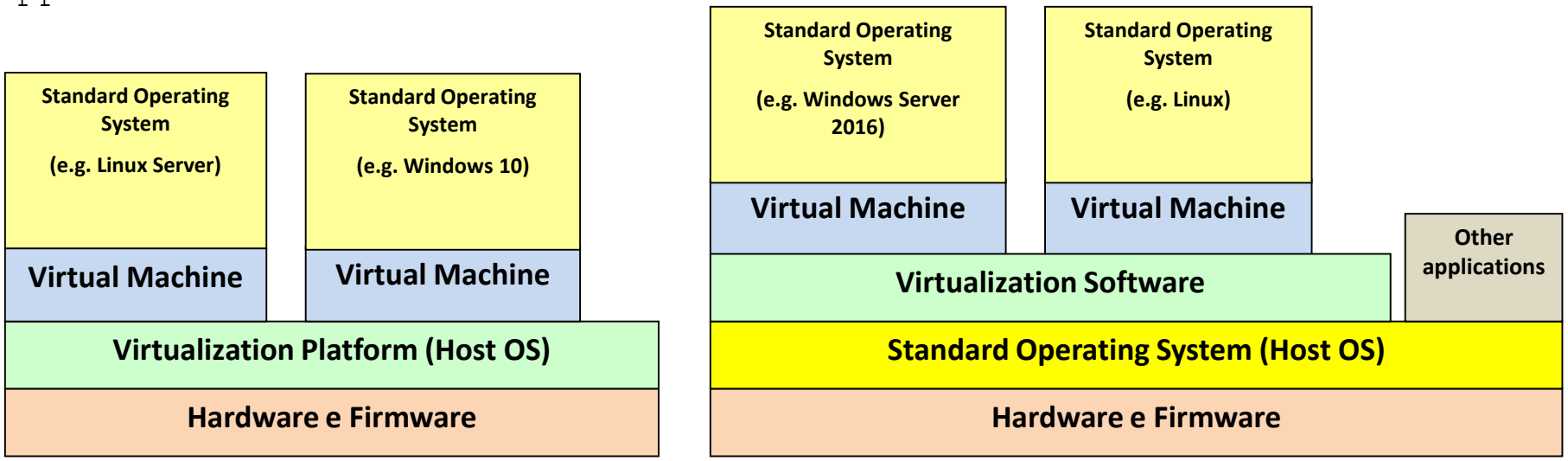
Modern CPUs support virtualization, it may have to be enabled on the hardware configuration, usually through a BIOS configuration option.

Virtualization allows a **safe** direct access of guest operating systems running in virtual machines to both the physical CPUs and the physical central memory (RAM). By using virtualization CPUs and memory are not simulated any more, the guest operating system now uses the real hardware, so a viable virtual machine is accomplished under performance point of view.

One drawback of virtualization is all virtual machines will have the kind of CPU identical to the physical CPU present.

Hardware Virtualization Platforms

The virtualization platform may be a specialized operating system, in that case the used hardware is entirely dedicated to virtualization or it may be an application running over a standard operating system. In the later, together with the virtualization software, other applications can be run.



In this course's laboratory classes we will be using **Oracle VM VirtualBox**, versions to run over several operating systems (host OS) are available, namely Linux, MS-Windows and OS X.

Among other popular virtualization platforms, we have VMware, Xen/XenServer, Hyper-V, and QEMU/KVM.

Virtual Machines' administration benefits

Virtual machines don't have physical hardware, it's all simulated by software, this has a significant impact on administration. All operations concerning hardware manipulation are now made by software, for instance: adding or removing RAM, adding or removing disks, adding and removing other devices like network adapters and video adapters.

Take the case of creating an exact copy of a disk, VM's disks are files on the host system, so it's just a matter of copying a file. Of course this should be done only if the VM is stopped.

Most virtualization systems provide a **clone** operation by which an exact copy of a VM may be created, usually the VM to be cloned must be stopped. Cloning can be used as a backup mechanism, even if the cloned VM is totally lost the clone can be used to restore it.

Another operation provided by most virtualization systems is snapshot, snapshots encompass the state of a running VM. By taking a snapshot, later the administrator will be able to role back the VM state to the time when the snapshot was taken. Nevertheless, snapshots can't be used for backup purposes, restoring a previous state through a snapshot will not work from VM with significant problems.

Virtual Machines efficiency benefits

On datacentres that maintain a large number of servers, efficiency is a key factor.

Of course, previously mentioned administration benefits, from using Virtual Servers instead of physical servers, have a direct role because one significant cost in a datacentre is administrators work hours, by making that work more productive there's a significant cost reduction there.

Yet, under hardware efficiency standpoint there're also very significant gains. In general, by using Virtual Servers, the same job can be accomplished with less physical hardware.

This happens because, with physical servers, hardware tends to be overused on some physical servers and underused on other physical servers. In a virtualization system, all existing hardware resources are assigned to virtual servers as need.

Reducing the amount of hardware in a datacentre, impacts on costs through the acquisition cost of the hardware itself, electric power consumption, refrigeration costs, and maintenance of the hardware.

Virtual Devices – Disks, CD/DVD and console

Virtual Machines' devices are simulated (virtual devices), the way they are implemented by the virtualization software encompasses in several cases interaction with users and administrators.

Disks and CD/DVD: often the virtual device is implemented by a file that is used to store the device's content, by copying one of these files we will be copying a disk or CD/DVD. ISO files usually burned into a real CD/DVD, can be directly used as a virtual CD/DVD.

The **console** is in fact a set of input and output devices (video display, keyboard and mouse) to be used as primary mean to interact with the Virtual Machine, namely during the boot and once the operating system starts.

Whoever has access to the console is able to perform low level operations like rebooting the Virtual Machine, installing a new operating system or start an existing operating system in recovery mode. Thus access to the virtual machine's console should be granted to the virtual machine's administrator only.

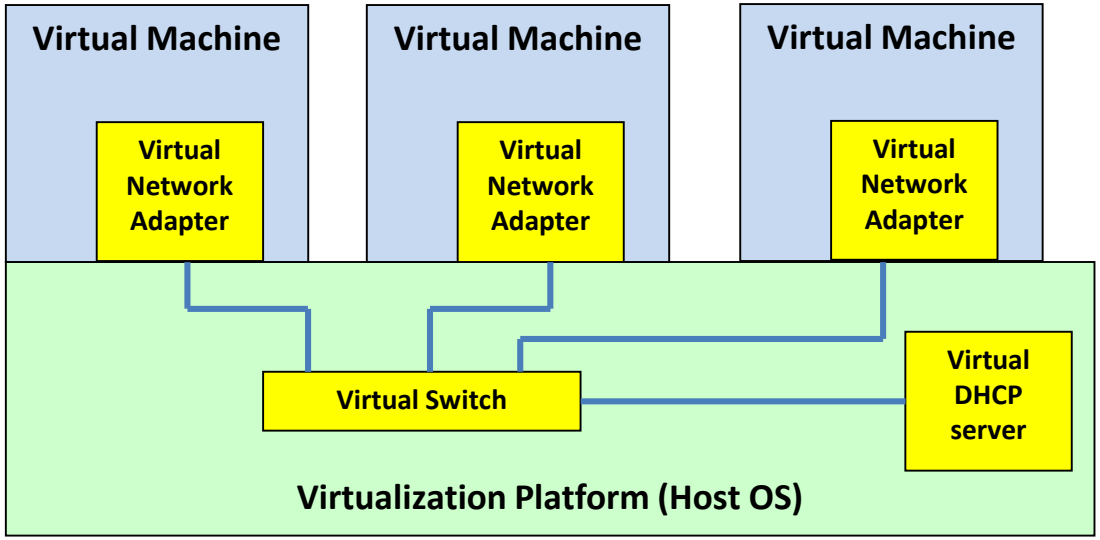
Usually, virtualization systems offer access to the console through specific software provided together with virtualization system.

Virtual Devices – Network adapters

For network adapters, virtualization platforms implement their own virtual networking infrastructures, usually in the form of **virtual switches**.

A virtual switch is a network switch simulated by software, virtual machines' network adapters are then connected to virtual switches within the virtualization platform.

Depending on the virtual switch settings, virtual machines with a network adapter connected to it may be able to access real networks connected to the virtualization system.

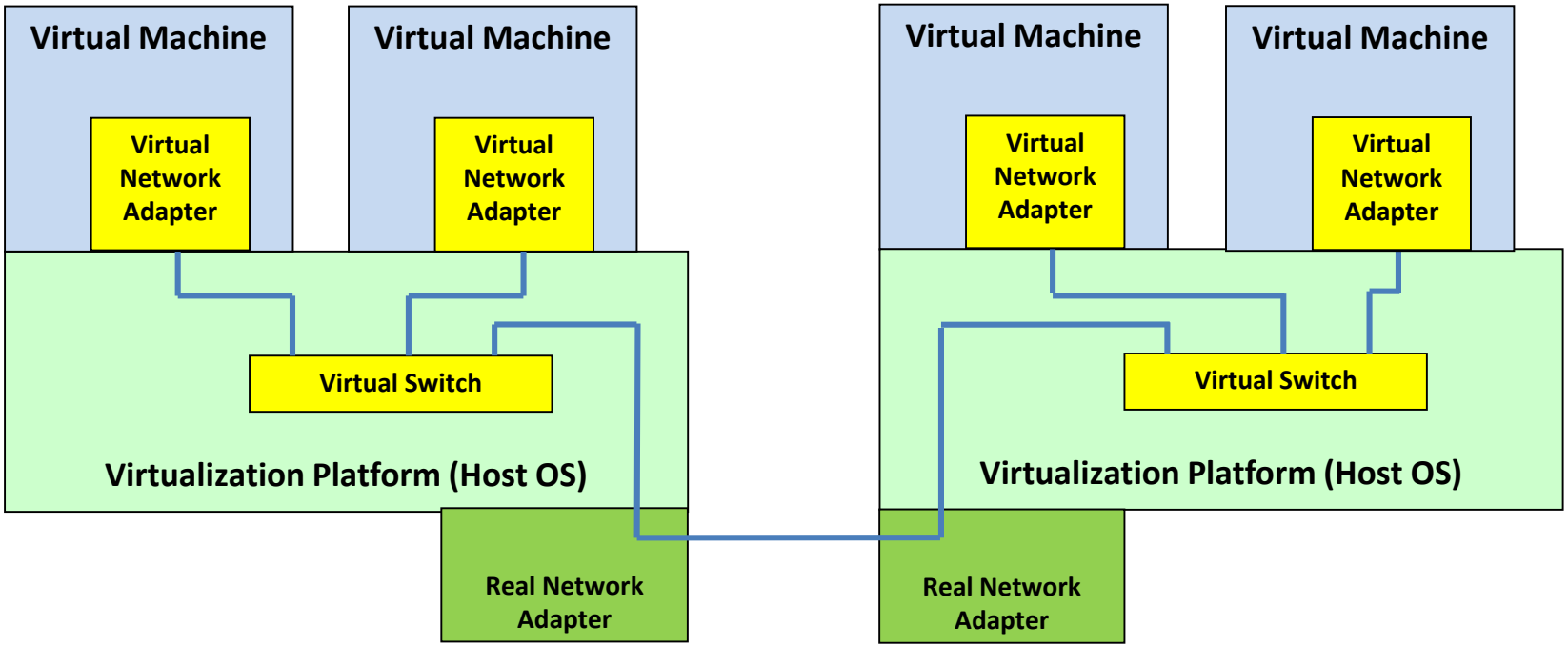


On the image the virtual switch isn't connected to any real network, so it will allow communications **only between virtual machines connected to it.**

Optionally, a virtual DHCP server may be connected to the virtual switch to provide automatic configuration.

Virtual Switches

If not connected to real networks adapters, virtual switches provide network connectivity only between virtual machines connected to it. Nevertheless, it may be possible to interconnect several virtual switches like this, even if they are hosted in different virtualization platforms.

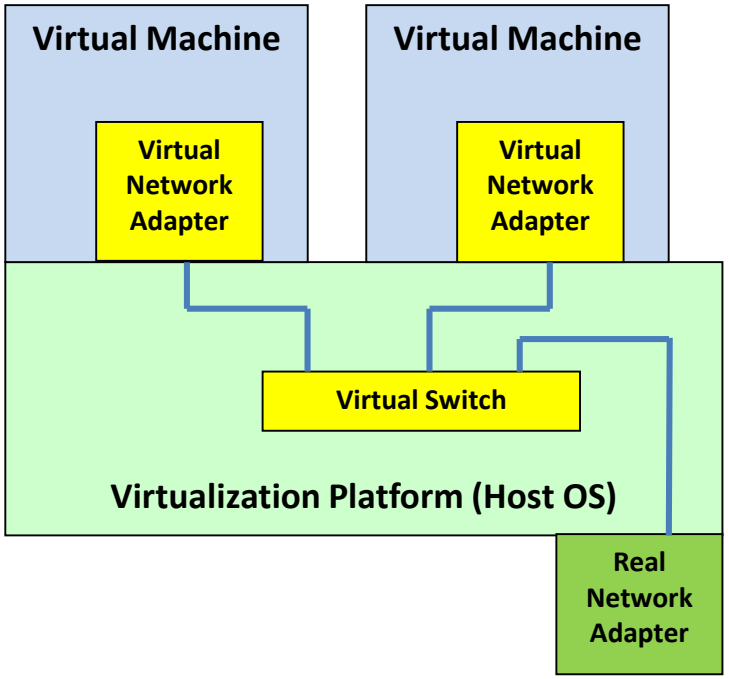


In this setup, virtual machines still aren't able to communicate with the real network, but now virtual machines hosted in different virtualization platforms are able to communicate because the two virtual switches now become a single **distributed virtual switch.**

Virtual Switches – bridge connection

One way to create a virtual switch providing real network access is called **bridge mode**, in bridge mode, layer two frames are directly transferred between the virtual switch and the real network.

In this setup, **the real network is extended into the virtual switch**. Thus, virtual machines connected to such a virtual switch operate exactly as if connected to the real network.



In this case, the virtual machines must configure the network adapter with an appropriate address for the real network. If DHCP is to be used, then in the real network there should be a real DHCP server operating. It will be that real DHCP server assigning configuration data to the guest operating system running in the virtual machine.

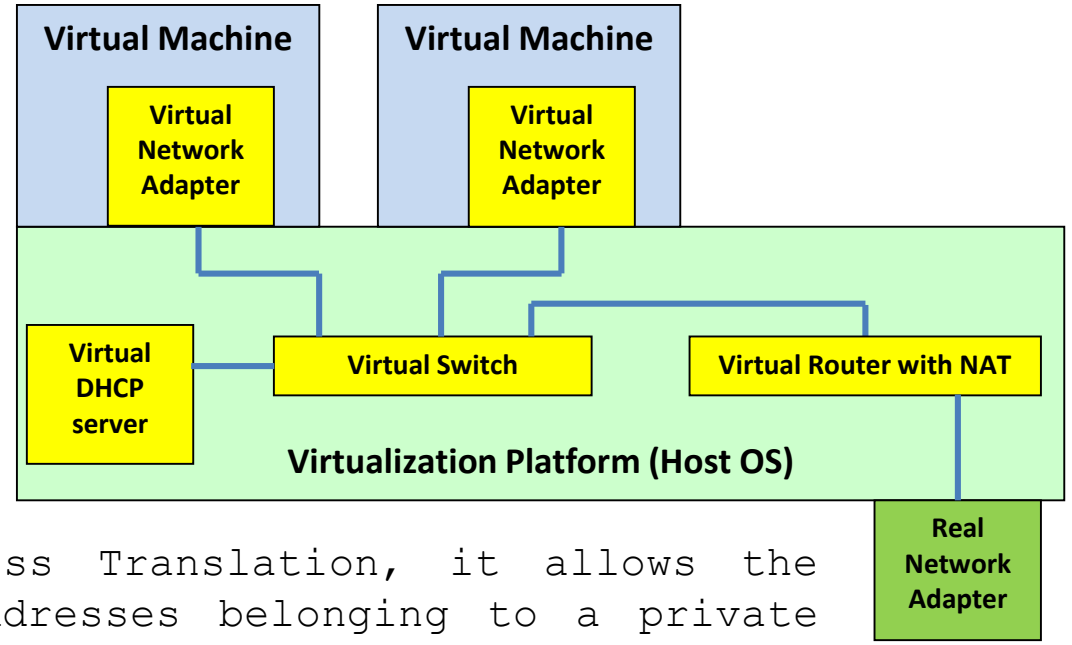
Using virtual switches connected to a real network in bridge mode is typically an option for virtual servers managed by systems administrators because it requires knowledge about the real network settings.

Virtual Switches – NAT connection

Another way to create a virtual switch providing real network access is called **NAT mode**. This encompasses a virtual switch with a virtual DHCP server and a virtual router performing NAT.

The virtual DHCP server provides IP configuration data to virtual machines within an internal private IP network, including the default gateway pointing out to the virtual router.

The virtual router implements the NAT function.

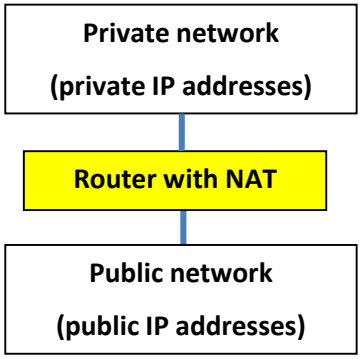


NAT stands for Network Address Translation, it allows the hiding of whole set of IP addresses belonging to a private network behind a single public IP address, in this case the public address of the real interface in the real network.

Under the real network's point of view all IP packets coming from the private network (virtual switch) will appear be coming from the real network adapter, this is because all packets will have as source address the public address of the real interface in the real network.

NAT (Network Address Translation)

NAT is a technique by which IP packets' source and destination addresses are changed with several purposes. In our scenario the purpose is hiding a private network behind a single public IP address.



In this case, for every IP packet being transferred from the private network to the public network, the NAT function will change the packet's source address to the public address of the router, so any packet outgoing to the public network will have that public source address.

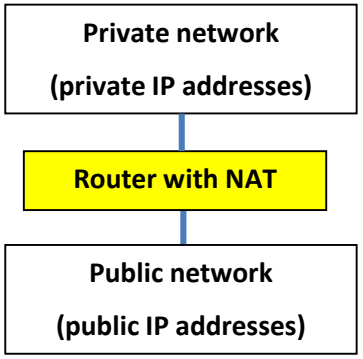
When this happens, the router stores in a NAT table the mapping between that outgoing packet and the original private address it had. Of course, later when a response to that packet arrives, it will have as destination address the router's public address. Thanks to the NAT table the router is able to match that response and will then change the destination address to be the initial source address within the private network.

By using NAT, nodes within a private network are able to communicate with public networks, including the internet.

Still there´s a catch, nodes on the private network are not reachable from public networks. Namely, servers won't be accessible.

NAT and services forwarding

Servers in a private network are not reachable from public networks, this is true because, with NAT, packets are transferred from the public network to the private network only if they match entries in the NAT table.



This happens if packets match entries created when packets were previously transferred from the private network to the public network.

However, the administrator can manually add static entries to the NAT table, misleading the NAT function.

For instance if there's a web server on the private network we could add an entry to the NAT table matching any web traffic arriving to router's public address and stating the corresponding private address is the web server's address.

Now, any web traffic directed to the router's public address will be redirected to the web server's private address. Notice the web server's private address is still not reachable from public networks, clients using the server must send requests to the router's public address.

Services forwarding schemas are supported in most virtualization platforms using virtual switches connected by NAT to a real network.

Operating systems virtualization

The idea behind hardware virtualization can be deployed at operating systems level.

Hardware virtualization allows a virtual machine to access real CPUs and memory in a controlled and closed environment in such a way it can't affect the system's integrity.

When implemented at the operating system level, this same idea results in the **container** concept. A container is a virtual closed environment where an independent instance of the operating system runs.

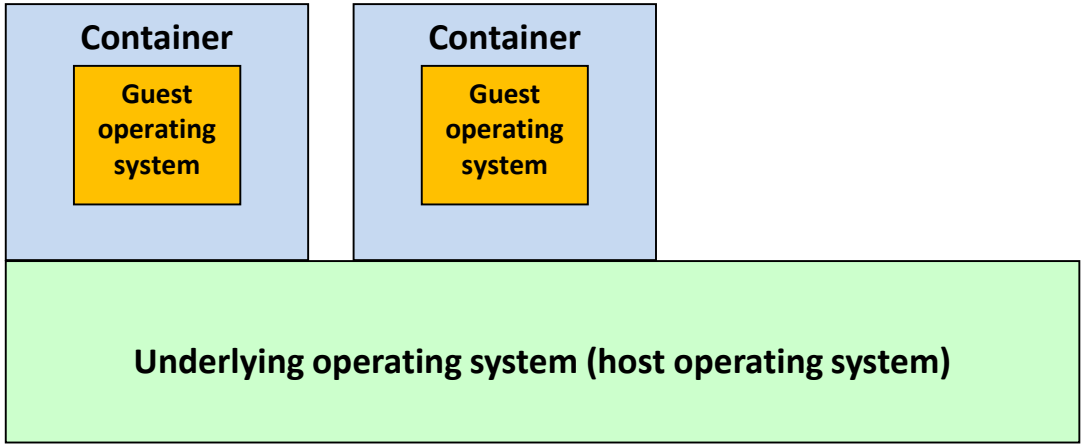
The same way a virtual machine is allowed to use the underling real hardware, the container is allowed to use the underling operating system in a controlled and closed environment in such a way it can't affect the underlying operating system's integrity.

Yet, because the real operating system being used is the underlying operating system, the operating system running inside the container must be the same.

Although the container uses the underlying operating system, it does so in a closed independent environment, it will not have access to resources managed by the underlying operating system like filesystems, and devices. **The container will have its own filesystem and its own simulated devices.**

Containers

The container will look pretty much like a virtual machine, but it's much lighter and uses considerably less resources.



The container makes use of direct system-calls to the underling operating system, but is able to use resources only within the container.

The container's operating system has its own filesystem, its own processes and memory and uses its own devices that are simulated by the container.

For instance, regarding network adapters **containers also use virtual switches** the very same way virtual machines do.

Containers and image files

A container is created from an image file, a container's image file stores the filesystem required for the operating system to start and operate, it's a minimal filesystem with only the necessary files for a specific purpose.

Public repositories make available images for several purposes, some may be just a minimal generic operating system, other may implement network services like web servers or database servers.

Once created, the container can be managed and modified to meet even more specific requirements. A container can be copied and it's also possible to create an image from a container.

The container concept points to the principle of avoiding having many services running in a single container, the idea is having a container for each service. This may be enforced by some containers implementations like Dockers.

Dockers containers are particularly light and usually run a single application inside them, they are often used to test applications under development in a closed controlled environment.

Linux containers (LXC) are to be used in laboratory classes of this course, inside an LXC container things will look like a standard Linux operating system with several processes and services running at the name time.

Containers

Remember containers use operating system virtualization, this means the operating system inside the container is the same as the underling operating system. You can't run Windows applications inside a container over Linux, likewise you can't run Linux inside a container over Windows. Actually, Docker implementations for Linux are based on LXC.

From the underlying operating system, where the container is running, several interactions with the operating system and applications inside the container are possible. They encompass for instance executing commands, opening a command line session and copying files to and from the filesystem within the container.

When the underling operating system is shutdown, all running containers are stopped, but their state is saved. Later when the underling operating system boots those stopped containers are restarted to the state they had when they were stopped.

Some hardware virtualization platforms may also support some of these features for Virtual Machines, but that usually requires adding specific drivers to the guest operating system, running "inside" the Virtual Machine.