Chapter 0: Prerequisites – Virtualisation

Operating Systems – Lab 0C · Adapted by Antonio Clim, PhD

v2.0 – Ubuntu 24.04 LTS

0.1 Introduction to Virtualisation

The term virtualisation derives from the Latin virtus — a notion associated with potentiality, with something that exists merely as a possibility without materialising directly. In a technical context, virtualisation encompasses all mechanisms whereby systems and applications run on intermediary software platforms rather than interacting directly with physical hardware. Physical resources — processors, memory and storage — are abstracted and presented to guest operating systems as virtual equivalents, with a specialised software layer called a hypervisor managing this translation.

Consider, for instance, a computer running Windows 10 on which a Linux system must also be operated, without dual-booting being acceptable for flexibility reasons. The solution involves installing a virtual machine: an application (the hypervisor) takes a portion of the host's resources — a disc file of several gigabytes in place of a physical partition, a fraction of RAM and one or more processor cores — and presents them to the guest system as if they were real hardware. The net effect: multiple independent operating systems coexist on a single physical machine, each with its own isolated execution environment.

0.2 Virtual Machines and Containers

Virtualisation is achieved through two fundamental mechanisms. The first employs a Type 1 hypervisor (also known as bare-metal), installed directly on hardware without an intermediary host operating system; it mediates virtual machine access to resources. The second type — the Type 2 hypervisor (hosted) — is installed as an application within an existing operating system (the host), adding an additional abstraction layer.

Figure 1. Type 1 hypervisor (bare-metal)

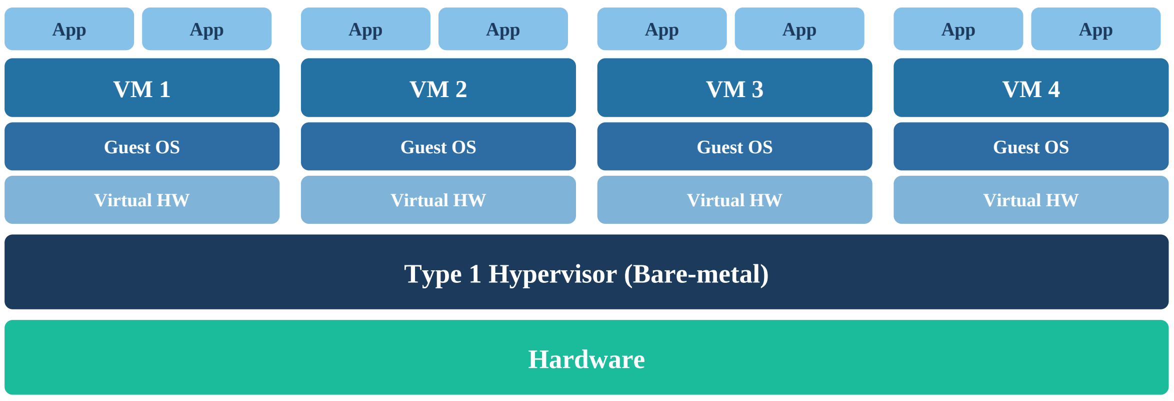
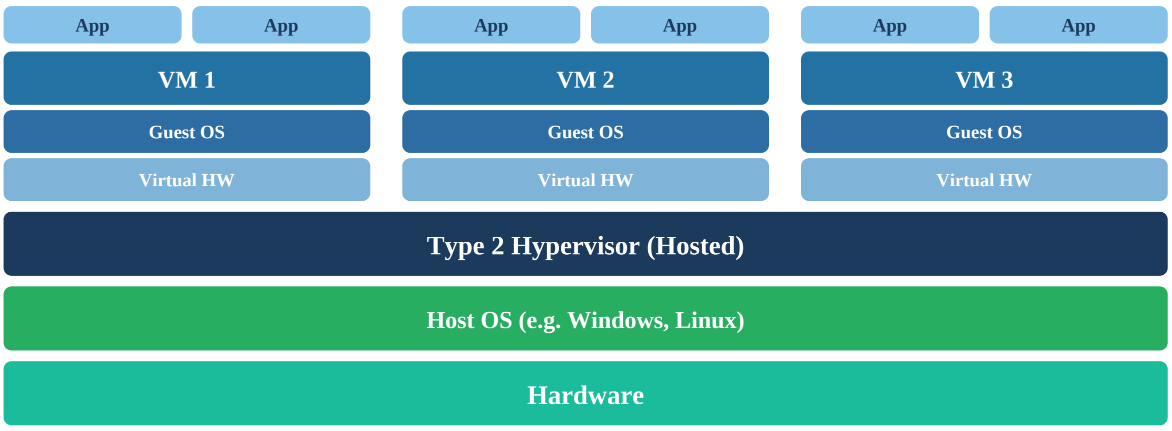


Figure 2. Type 2 hypervisor (hosted)



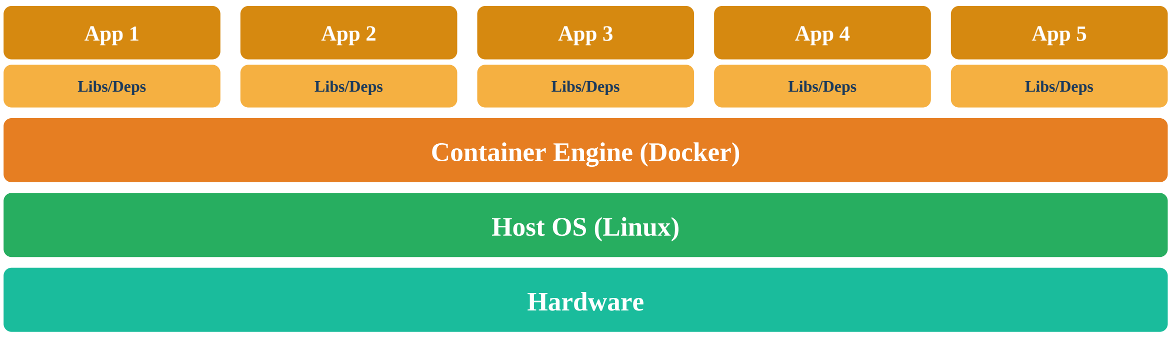
The operating system running inside a virtual machine is called the guest OS. It is typically installed from a preconfigured image, although classic installations from optical media, via PXE or other network boot mechanisms are also possible. Subsequent configuration — security patches, network settings and permissions — falls to the system administrator.

A major advantage of virtualisation lies in efficient resource utilisation. In a traditional scenario, a high-performance physical server hosts a single operating system running multiple applications; should the system fail, all become unavailable. Through virtualisation, each application receives its own isolated environment with specific settings, and a VM crash affects only that service — the others continue operating, thus achieving high availability.

Containerisation

A significant evolution of the virtualisation paradigm is containerisation. Unlike virtual machines, a container does not include a complete guest operating system; instead, the containerised application relies on a container engine (e.g. Docker Engine) that communicates directly with the host OS kernel without hypervisor mediation. The developer packages the application together with all required dependencies into the container, achieving portability and reproducibility.

Figure 3. Container architecture (Docker)



Comparison between virtual machines and containers

Criterion	Virtual Machines	Containers
Isolation	Complete (separate OS)	Process-level (shared kernel)
Resource usage	High (full OS per VM)	Low (no additional OS)
Start-up time	Minutes	Seconds
Portability	Limited by hypervisor	High (standardised images)
Security	Strong isolation	Requires careful configuration

Scalability	Moderate	High (K8s orchestration)

0.3 Virtualisation Solutions

Choosing a virtualisation solution requires careful consideration, as compatibility between different products is often problematic. Virtual machines are deployed from an image containing the guest OS, applications and configurations made by the administrator.

VMware — One of the oldest and most stable virtualisation platforms, currently owned by Broadcom (acquired in 2023). The vSphere and ESXi products are valued for their reliability and enterprise support.

KVM (Kernel-based Virtual Machine) — An open-source hypervisor integrated directly into the Linux kernel. It offers superior scalability over proprietary solutions with no licensing costs. Not to be confused with a KVM switch (keyboard-video-mouse).

Xen — A Type 1 open-source hypervisor and a collaborative project of the Linux Foundation. Its microkernel design yields high performance with low resource consumption.

Hyper-V — Microsoft's virtualisation solution, integrated into Windows Server and Windows 10/11 Pro. It enables running Linux distributions in Windows environments.

VirtualBox — An Oracle open-source product, easy to install and configure on Linux, macOS or Windows. Recommended for desktop use, less suitable for large-scale production environments.

Docker — A container engine that enables the creation of containerised applications — autonomous components that do not require a dedicated host OS. Containerd facilitates code and dependency packaging. A leading project of the Cloud Native Computing Foundation.

Kubernetes — A container orchestration platform, originally developed by Google and open-sourced in 2014. It organises resources into clusters, nodes and pods, providing automated management, load balancing and horizontal scaling.

0.4 IaaS — Infrastructure as a Service

The concepts presented above form the foundation of the IaaS (Infrastructure as a Service) model, a form of cloud computing in which computing resources — virtual instances, block storage and networking — are delivered on demand over the internet. Computing systems reside in external data centres, managed by hypervisors and interconnected via software-defined networks, communicating through the APIs of orchestration platforms such as OpenStack, Apache CloudStack and OpenNebula.

0.5 VM and Container Configuration

When a new virtual machine is created, it must be configured to communicate with the external network: a unique hostname, IP address, subnet mask, DNS and default route. The base image is stored as a set of files ready to run, but it lacks essential information — network settings, permissions, database routes and SSH keys — which are injected through initialisation scripts.

Utilities such as cloud-init simplify this process: they call APIs, read metadata from a dedicated server and configure instances automatically. In Kubernetes, Init Containers run before the main application starts.

SSH keys are used for secure administration of containers and VMs; they are generated automatically at installation or manually with `ssh-keygen`. Another critical element is the UUID, stored in the D-Bus machine ID file (`/etc/machine-id`) — a 128-bit number with negligible collision probability.

Virtualisation Extensions

Efficient hypervisor operation requires hardware support: Intel VT-x or AMD-V extensions, activated in BIOS/UEFI. Verification is performed by inspecting `/proc/cpuinfo`: the presence of the `vmx` (Intel) or `svm` (AMD) flag confirms activation, whilst `lm` indicates a 64-bit architecture.

Verifying virtualisation extensions

```
$ grep -cE "vmx|svm" /proc/cpuinfo
8
$ lscpu | grep Architecture
Architecture:    x86_64
```

Glossary

Application container — A packaged application that accesses computing resources directly through a container engine, without a complete guest OS.

D-Bus machine ID — A configuration file containing the system's unique UUID, generated at installation or first boot.

Guest drivers / Guest agents — Software components installed in the guest OS that add functionality such as mouse support, time synchronisation and folder sharing between host and guest.

SSH host keys — Randomly generated cryptographic key pairs used for secure authentication between systems.

Virtual Machine (VM) — A virtualised computing system providing access to a complete OS installed on virtual hardware furnished by a hypervisor.

Practical Section: Installing Ubuntu 24.04 LTS

A. Preparing VirtualBox

Regardless of the chosen distribution, instance configuration in VirtualBox follows the same steps. Whilst configuring, you may download the Ubuntu Server 24.04 LTS image from https://ubuntu.com/download/server.

Step 1. Open VirtualBox and click New. In the creation wizard:

- Name: `srv1033` (e.g. srv + study group number)
- Type: Linux
- Version: Ubuntu (64-bit)
- Memory: minimum 1024 MB (2048 MB recommended)
- Hard disc: VDI, dynamically allocated, 15–20 GB

Step 2. In Settings → Storage, attach the downloaded `.iso` file to the IDE controller (empty disc icon).

Step 3. In Settings → Network, for Adapter 1 select Bridged Adapter instead of NAT, so the VM receives an IP address from the local network directly from the DHCP server.

B. Installing Ubuntu Server 24.04 LTS

Ubuntu, derived from Debian, is one of the most widely deployed Linux distributions. Version 24.04 LTS (Noble Numbat) benefits from five-year extended support (until April 2029). The modern installer, Subiquity, replaces the legacy debian-installer with an intuitive TUI interface.

Step 1. Start the VM. At the boot screen, select Try or Install Ubuntu Server.

Step 2. Select the language (English recommended for server environments) and keyboard layout.

Step 3. For installation type, choose Ubuntu Server (without desktop environment).

Step 4. Network configuration: if the adapter is set to Bridged, the interface will automatically receive an IP address via DHCP. Verify and confirm.

Step 5. Proxy configuration: leave the field empty (unless you are behind a proxy server).

Step 6. Ubuntu mirror: keep the default value (`http://archive.ubuntu.com/ubuntu`).

Step 7. Disc configuration: select Use an entire disk. Accept the proposed partitioning scheme (LVM by default).

Step 8. Confirm writing to disc (this action is irreversible for the VM contents).

Step 9. Create the user account:

```
Example credentials

Your name:   Student SOP
Server name: srv1033
Username:    1033_climantonio
Password:    student
```

Step 10. At the SSH screen, tick Install OpenSSH server. Do not select additional packages from Featured Server Snaps.

Step 11. Wait for installation to complete, then select Reboot Now. When prompted, remove the ISO image from Storage and press Enter.

C. Post-Installation Configuration

C.1. First Login and Basic Commands

After reboot, you will be greeted by the login screen (tty/getty). Enter the credentials created during installation.

Initial verification commands

```
$ whoami
1033_climantonio
$ hostname
srv1033
$ date
Wed Feb 12 10:30:00 EET 2025
$ ip addr show | grep inet
    inet 127.0.0.1/8 scope host lo
    inet 192.168.1.105/24 brd 192.168.1.255 scope global dynamic enp0s3
```

C.2. Connecting via SSH

To facilitate work, connect to the VM via SSH from the host system. On Linux/macOS, use the terminal directly; on Windows, use `ssh` from PowerShell/CMD or the PuTTY application.

SSH connection and privilege escalation

```
# SSH connection from the host system
$ ssh 1033_climantonio@192.168.1.105
1033_climantonio@192.168.1.105's password: ********
Welcome to Ubuntu 24.04 LTS (GNU/Linux 6.8.0-xx-generic x86_64)

# Obtaining root privileges
$ sudo -i
[sudo] password for 1033_climantonio: ********
root@srv1033:~#
```

Important note: In Ubuntu, the root account is disabled by default. Administrative privileges are obtained through the `sudo` *command, which prompts for the current user's password (not root's).*

C.3. Setting the Hostname

The hostname is configured during installation but can be modified afterwards with `hostnamectl`:

Hostname configuration

```
$ hostnamectl
  Static hostname: srv1033
  ...
  Operating System: Ubuntu 24.04 LTS
  Kernel: Linux 6.8.0-xx-generic

# Changing hostname
$ sudo hostnamectl set-hostname srv1033.clim.local
$ cat /etc/hostname
srv1033.clim.local
```

C.4. Adding a Static IP Address

Ubuntu 24.04 uses Netplan for network configuration. Configuration files reside in `/etc/netplan/`.

Static IP configuration with Netplan

```
# View current configuration
$ cat /etc/netplan/50-cloud-init.yaml

# Example static IP configuration
$ sudo nano /etc/netplan/01-static.yaml
network:
  version: 2
  ethernets:
    enp0s3:
      addresses:
        - 192.168.1.105/24
      routes:
        - to: default
          via: 192.168.1.1
      nameservers:
        addresses: [8.8.8.8, 8.8.4.4]

$ sudo netplan apply
```

C.5. Package Management

Ubuntu uses the APT (Advanced Package Tool) package manager. Regular system updates are essential for security.

Package management with APT

```
# Update package lists
$ sudo apt update

# Apply upgrades
$ sudo apt upgrade -y

# Install useful packages
$ sudo apt install -y vim tmux mc htop net-tools
```