# Useful CLI tools

* Nslookup – it can check corresponding IP address for a domain name and vice versa
* Iptables – it is a tool used for setting up for example firewall rules.
* Ping – test network connectivity
* traceroute / tracert - See the path traffic takes to a destination
* netstat / ss - View open ports and network connections
* tcpdump / wireshark - Capture and analyze network packets

# Resolving domain names

We can assign to servers domain (DNS) names which will be used in communication instead of IP addresses.

Resolving domain names means finding an IP address matching given DNS name and vice vers.

DNS servers are used for resolving domain names.

# DNS

DNS is a server used for translating domain (DNS) names into IP addresses and vice versa.

/etc/resolv.conf file contains information about IP addresses of the DNS servers which are used for translating DNS names.

In the resolv.conf file there might be written:

* search domain-name-1 domain-name-2 …
* Options ndots:x

That means there need to be at least x dots in the domain name in order to be treated as FQDN.

So if host tries to resolve a domain name which has less than x dots, then it will try to append domain names from the ‘search’ field to the domain name which it is trying to resolve.

There might be a situation that host will try to use only the first domain name from the ‘search’ field and if it fails it will not try other names.

# Load balancer

Load balancer distributes network traffic across multiple servers. It can redirect traffic to the least busy server.

# Reverse proxy

It forwards client’s requests to application servers. It is a gateway between users and application servers.

# Subnet

Logically separated section of a network.

# Gateway

A device that routes traffic from one network to another.

# SSL/TLS

Protocols that secure data transfer over internet (used in HTTPS)

# VPN

Encrypts a traffic over network and routes it through a secure server.

# Protocols

## SSH

A protocol for accessing remote machines.

## TCP

## HTTPS

# Packets

Data is split into packets before being transmitted over a network.

# LAN

LAN stands for Local Area Network. It is a network of devices which communicate with each other directly, without going through the internet.

For example a laptop and phone connected to the same wi-fi or router are part of the same LAN, or laptop and a printer connected by a cable can also be in the same LAN.

# Network interface

Network interface specifies how computer, container, VM or Kubernetes Pod connects to a network.

It can be physical (like a network card or wi-fi adapter) or virtual (used by VMs, containers or Pods).

It acts as a gateway that allows a device to send and receive data over a network.

# Bridge

A bridge connects multiple network interfaces at Layer 2 of the OSI model so they can communicate as if they are part of the same LAN (without using the internet).

For example it can enable communication between computers, containers, VMs or Kubernetes Pods as if they are on the same local network.

# Netfilter

It is a framework built into the Linux kernel that allows to inspect, modify and control network traffic.

It can:

* Decide whether to allow or block packets
* Modify packets
* Log traffic

We can configure the Netfilter rules using for example iptables CLI tool.

It can be used to set up firewall rules.

# Firewall

It is a set of rules regarding what traffic is allowed (on which ports, from which IPs, using which protocols etc).

It can be set up using Netfilter.

# Ethernet

Ethernet is a protocol used to send data between devices in a LAN.

It works at Layer 2 of the OSI model (Data Link Layer).

For example when we plug in a network cable into a computer, Ethernet is used to send data over that cable.

# OSI model

It is a conceptual framework used to understand how network systems communicate in layers.

It has the following layers:

1. Physical
   1. The actual hardware (cables, switches, electrical signals)
2. Data Link
   1. Transfers frames between devices on the same network (e.g., Ethernet, MAC addresses)
3. Network
   1. Routing data across networks (e.g., IP addresses)
4. Transport
   1. Reliable delivery (e.g., TCP/UDP)
5. Session
   1. Manages sessions between applications
6. Presentation
   1. Translates data (e.g., encryption, compression)
7. Application
   1. Interfaces with apps (e.g., browsers, APIs)

## Data Link Layer

It defines how devices on the same physical or virtual network (like LAN) talk to each other.

It uses MAC addresses to identify devices.

It packages data into frames to be sent over the network.

Examples of Data Link Layer protocols:

* Ethernet
* Wi-Fi

## Network Layer

It handles routing and forwarding of data packets between devices across different networks.

Key functions of this layer:

* Addressing
  + It uses IP addresses to identify source and destination devices on the network.
* Routing
  + Determines the best path for data to travel from source to destination
* Packet Forwarding
  + Transfers packets from one network segment to another using routers

Common protocols in this layer:

* IP (IPv4, IPv6 – for IP addresses)

## Transport layer

Key functions:

* **Segmentation and reassemly**
  + Breaks big packets of data into smaller pieces at a sender and reassemblies them at a receiver.
* **Connection Management**
  + **Connection-oriented communication** (e.g., TCP): Establishes, maintains, and terminates a connection.
  + **Connectionless communication** (e.g., UDP): No session is maintained between devices

Popular protocols in this layer:

* TCP
* UDP

## Session Layer

It is responsible for creating, managing and terminating sessions between applications.

When two applications / devices want to talk to each other, a session is created and used for exchanging information. When the conversation ends the session is done.