# INCEPTION

## Docker Fundamental Concepts

Concepts Guide for the Project

42 School - System Administration

# TABLE OF CONTENTS

# 1. INTRODUCTION TO DOCKER

## What is Docker?

> Docker is an open-source platform that automates the deployment of applications within software containers. A container packages an application together with all its dependencies (libraries, configuration files, etc.) into a standard unit that can run consistently across any environment.

## Why use Docker?

**Portability:** Containers work the same way in development, testing, and production.

**Isolation:** Each container is isolated from other containers and the host system.

**Efficiency:** Containers share the operating system kernel, making them lighter than virtual machines.

**Reproducibility:** The same container produces the same results anywhere.

# 2. CONTAINERS VS VIRTUAL MACHINES

## Virtual Machine (VM)

A virtual machine is a complete operating system running on top of a hypervisor. Each VM includes a full copy of the operating system, applications, necessary libraries, and several gigabytes of disk space. This makes them heavy and slow to start.

## Docker Container

A container shares the host operating system kernel and isolates only the application and its dependencies. It doesn't need a complete operating system, making it much lighter (typically megabytes instead of gigabytes) and faster to start (seconds instead of minutes).

## Key Differences

**Size:** VMs (several GB) vs Containers (several MB)

**Speed:** VMs (minutes to boot) vs Containers (seconds)

**Resources:** VMs (reserve resources) vs Containers (share resources dynamically)

**Isolation:** VMs (complete hardware isolation) vs Containers (process-level isolation)

# 3. DOCKER IMAGES

## What is a Docker Image?

A Docker image is a read-only template containing a set of instructions for creating a container. It's like a 'photo' or 'snapshot' of a system with everything needed to run an application: the base operating system, software, libraries, dependencies, and application code.

## Dockerfile

A Dockerfile is a text file containing the instructions needed to build a Docker image. It defines which base operating system to use, what software to install, which files to copy, and what command to execute when the container starts.

## Layers

Docker images are composed of layers. Each instruction in the Dockerfile creates a new layer. Layers are read-only and stacked on top of each other. Docker reuses common layers between different images, saving disk space.

## Base Images

In the Inception project, all images are built from Debian Bookworm as the base image. Debian is a stable and well-documented Linux distribution. Bookworm is the code name for Debian 12, the current stable version.

# 4. DOCKER COMPOSE

## What is Docker Compose?

Docker Compose is a tool for defining and running multi-container Docker applications. It uses a YAML file to configure all application services, their networks, and volumes. With a single command, you can create and start all defined services.

## docker-compose.yml

This is the configuration file where the complete infrastructure is defined. It specifies: which containers to create, from which images, which ports to expose, which volumes to mount, which networks to use, and which environment variables to configure.

## Services

A service is the definition of a container in Docker Compose. Each service has a name (like nginx, wordpress, mariadb) and an associated configuration. Services can communicate with each other using their names as hostnames thanks to Docker's internal DNS.

## Orchestration

Docker Compose orchestrates the startup of containers in the correct order, respecting dependencies. If WordPress depends on MariaDB, Compose ensures MariaDB is ready before starting WordPress.

# 5. DOCKER NETWORKS

## Docker Network

A Docker network allows containers to communicate with each other. Docker provides network isolation by default, so containers on different networks cannot communicate directly.

## Bridge Network

This is the most common type of network in Docker. It creates a private internal network on the host where connected containers can communicate with each other. It's like having a virtual switch to which all containers connect. In Inception, all services are on a bridge network called 'inception'.

## Internal DNS

Docker provides an automatic internal DNS server. Each container can communicate with others using the service name as hostname. For example, WordPress can connect to MariaDB simply using 'mariadb:3306' as the database server address.

## Isolation

Containers on the same network can communicate, but they are isolated from the host and other networks. Only explicitly exposed ports (like NGINX's 443) are accessible from outside the Docker network.

# 6. VOLUMES AND PERSISTENCE

## What is a Volume?

> A Docker volume is a mechanism to persist data generated and used by containers. Containers are ephemeral by nature: when deleted, all their data is lost. Volumes allow data to survive the container's lifecycle.

## Bind Mount

> A bind mount links a directory from the host system to a directory inside the container. In Inception, WordPress and MariaDB data is stored in /home/user/data/ on the host and mounted inside the containers. This allows direct access to data from the host and guarantees its persistence.

## Data Persistence

Thanks to volumes, important data persists even if containers are stopped, restarted, or deleted. In the project, the WordPress blog and MariaDB database remain safe on the host, allowing containers to be rebuilt without losing any data.

# 7. NGINX

## What is NGINX?

NGINX is a high-performance web server and reverse proxy. It's known for its efficiency, stability, and low resource consumption. It can handle thousands of simultaneous connections thanks to its event-driven architecture.

## Web Server

As a web server, NGINX can serve static files directly: HTML, CSS, JavaScript, images, etc. It's very efficient at this task because it uses advanced file transfer techniques at the operating system kernel level.

## Event-Driven Architecture

Unlike other servers that create a process or thread per connection, NGINX uses an asynchronous event-driven model. It has a master process and multiple worker processes that can handle thousands of connections each without blocking.

## TLS/SSL

NGINX handles TLS (Transport Layer Security) encryption, the modern version of SSL. In the Inception project, NGINX is the entry point that accepts HTTPS connections on port 443, using TLS 1.2 or TLS 1.3 to encrypt all communication.

# 8. PHP-FPM

## What is PHP-FPM?

PHP-FPM (FastCGI Process Manager) is an alternative implementation of PHP FastCGI. It's a PHP process manager optimized for high-traffic websites. It separates PHP processing from the web server, allowing for a more scalable and efficient architecture.

## Process Management

PHP-FPM maintains a pool of PHP worker processes always ready to process requests. It can dynamically manage these processes: create more when there's high load and remove some when demand decreases.

## Socket Communication

PHP-FPM can communicate through Unix sockets or TCP ports. In the Inception project, WordPress uses PHP-FPM listening on port 9000, and Adminer on port 9001. NGINX connects to these ports to send PHP requests that need processing.

## Advantages over mod_php

Unlike mod_php which runs inside the web server, PHP-FPM is a separate process. This allows: better isolation, ability to run different PHP versions, restarting PHP without restarting the web server, and better resource usage on servers with multiple sites.

# 9. FASTCGI

## What is FastCGI?

FastCGI is a protocol for interconnecting external programs with a web server. It's an evolution of CGI (Common Gateway Interface) designed to improve performance by maintaining persistent processes instead of creating a new process for each request.

## CGI vs FastCGI

Traditional CGI creates a new process for each request, which consumes many resources. FastCGI maintains persistent processes that can handle multiple requests. This drastically reduces the overhead of creating and destroying processes.

## NGINX-PHP Communication

When NGINX receives a request for a PHP file, it cannot execute it directly. Instead, it uses the FastCGI protocol to communicate with PHP-FPM. NGINX sends the request through FastCGI, PHP-FPM processes the PHP script and returns the result to NGINX, which finally sends it to the client.

## FastCGI Parameters

FastCGI transmits information about the request through parameters. These include: which script to execute (SCRIPT_FILENAME), client information (REMOTE_ADDR), HTTP request data (REQUEST_METHOD, QUERY_STRING), and more. These parameters allow the PHP script to have all the necessary context to generate the correct response.

# 10. MARIADB

## What is MariaDB?

MariaDB is an open-source relational database management system (RDBMS). It's a fork of MySQL created by MySQL's original developers after its acquisition by Oracle. It's fully compatible with MySQL in terms of commands, APIs, and protocols.

## Relational Database

A relational database organizes information into tables with rows and columns. Tables can be related to each other through keys. MariaDB uses SQL (Structured Query Language) to query and manipulate this data. It's perfect for WordPress, which needs to store posts, users, comments, settings, etc.

## ACID

MariaDB guarantees ACID properties: Atomicity (transactions are all or nothing), Consistency (data always follows rules), Isolation (transactions don't interfere with each other), and Durability (committed data persists even in case of failures). This ensures blog data integrity.

## Port 3306

MariaDB listens by default on port 3306. In the Inception project, this port is NOT exposed to the outside, it's only accessible from the internal Docker network. WordPress connects to 'mariadb:3306' to access the database securely.

# 11. WORDPRESS

## What is WordPress?

WordPress is an open-source content management system (CMS) written in PHP. Originally designed for blogs, it now powers over 40% of all websites on the Internet. It allows creating and managing websites without needing to code.

## CMS (Content Management System)

A CMS is software that allows creating, editing, organizing, and publishing digital content. WordPress provides a user-friendly web interface where users can write posts, upload images, manage comments, change site design, and much more, without touching code.

## WordPress Architecture

WordPress is written in PHP and requires a MySQL or MariaDB database. WordPress PHP code queries the database to retrieve posts, users, settings, etc., and generates dynamic HTML pages. WordPress files include: PHP code (core), themes (visual appearance), plugins (extra functionalities), and uploads (images and files).

## Users and Roles

WordPress has a role system: Administrator (full control), Editor (manages content), Author (publishes their own posts), Contributor (writes but doesn't publish), and Subscriber (only reads). In the Inception project, two users are created: an administrator and a regular user.

# 12. SSL/TLS

## What is SSL/TLS?

SSL (Secure Sockets Layer) and TLS (Transport Layer Security) are cryptographic protocols that provide secure communications over a network. TLS is the modern and more secure version of SSL. They encrypt communication between the user's browser and the web server, protecting data privacy and integrity.

## Encryption

Encryption transforms data into an unreadable format during transmission. Only the recipient with the correct key can decrypt it. This protects sensitive information like passwords, personal data, or payment information from being intercepted by attackers.

## Certificates

An SSL/TLS certificate is a file containing a public key and information about the server's identity. In production, these certificates are issued by trusted Certificate Authorities (CA). In the Inception project, self-signed certificates generated with OpenSSL are used, valid for development but generating browser warnings.

## TLS 1.2 and 1.3

TLS 1.2 (2008) and TLS 1.3 (2018) are the current protocol versions. TLS 1.0 and 1.1 are obsolete and have known vulnerabilities. The Inception project configures NGINX to accept ONLY TLS 1.2 and 1.3, rejecting old insecure versions. TLS 1.3 is faster and more secure, eliminating weak encryption algorithms.

## HTTPS

HTTPS (HTTP Secure) is HTTP over TLS. When a site uses HTTPS, all communication is encrypted. The padlock in the browser's address bar indicates a secure HTTPS connection. The standard HTTPS port is 443, while HTTP uses port 80.

# 13. WP-CLI

## What is WP-CLI?

WP-CLI (WordPress Command Line Interface) is a command-line tool for managing WordPress. It allows performing virtually any action that can be done from the web admin panel, but in an automated way through commands.

## Automation

WP-CLI is ideal for automated installation and configuration scripts. Instead of following the WordPress web installation wizard manually, WP-CLI can be used to download WordPress, create configuration, install the database, and create users automatically through commands.

## Use in Inception

In the Inception project, WP-CLI is used in the WordPress container initialization script. It allows downloading WordPress core, configuring database connection, installing WordPress with title and administrator, creating the second user, all without manual intervention. This makes the container reproducible and automated.

# 14. PROCESSES AND PID 1

## What is a Process?

A process is a program in execution. Each process has a unique identifier called PID (Process ID). In a Linux system, all processes form a hierarchical tree, where each process has a parent process (except the root process).

## PID 1 in Linux

In a normal Linux system, PID 1 is the init process (or systemd in modern distributions). It's the first process to start and the ancestor of all other processes. If PID 1 dies, the system shuts down.

## PID 1 in Docker

In a Docker container, the process specified in CMD or ENTRYPOINT becomes PID 1. Docker monitors this process: if PID 1 terminates, Docker considers the container has finished and stops it. Therefore, it's crucial that the container's main process runs in the foreground and doesn't terminate prematurely.

## Daemon vs Foreground

A daemon is a process that runs in the background. On traditional servers, services like NGINX run as daemons to free the terminal. However, in Docker, if a service daemonizes, the command that launched it terminates immediately. Docker sees that PID 1 terminated and kills the container. That's why in Docker, services must run in the foreground using special options.

## Examples in Inception

NGINX uses 'daemon off' to force foreground execution. PHP-FPM uses the '-F' flag (foreground). MariaDB uses 'mariadbd' which runs in foreground by default. These configurations ensure the service is PID 1 and the container remains active while the service is running.

# 15. REVERSE PROXY

## What is a Proxy?

A proxy is an intermediary between clients and servers. There are two main types: forward proxy which represents clients, and reverse proxy which represents servers.

## Reverse Proxy

A reverse proxy sits in front of one or more backend servers and receives all client requests. Clients believe they're communicating directly with the server, but they're actually communicating with the proxy, which decides which backend server to forward each request to.

## NGINX as Reverse Proxy

In the Inception project, NGINX acts as a reverse proxy. It's the only entry point accessible from the Internet (port 443). It receives all HTTPS requests and, depending on the requested path, forwards the request to the appropriate service: WordPress for general paths, Adminer for /adminer/, and static-site for /portfolio.

## Advantages of Reverse Proxy

**Security:** Backend servers are not directly exposed to the Internet.

**Load balancing:** Can distribute requests among multiple backend servers.

**SSL/TLS Termination:** The proxy handles encryption, freeing backends from this task.

**Caching:** Can cache responses to improve performance.

**Compression:** Can compress responses before sending them to the client.

# 16. SECURITY CONCEPTS

## Principle of Least Privilege

This principle establishes that a process or user should only have the minimum permissions necessary to perform its function. In Docker, services run with the www-data user (not root) whenever possible. If an attacker compromises the service, they have limited system access.

## Environment Variables

Environment variables allow configuring applications without hardcoding sensitive values in the source code. In Inception, passwords and configurations are stored in a .env file that is NOT uploaded to the Git repository. Docker injects these variables into containers at runtime.

## Network Isolation

Containers in Docker are isolated by default. In Inception, MariaDB does NOT expose its port 3306 to the outside, it's only accessible within the internal Docker network. WordPress can connect to MariaDB, but the Internet cannot directly access the database.

## Secrets Management

Secrets (passwords, API keys, certificates) should never be stored in source code or Docker images. The project subject emphasizes using the .env file for credentials and suggests Docker secrets for confidential information. This prevents accidental exposure of credentials in public repositories.

## Updates and Patches

Keeping software updated is crucial for security. Using official base images like Debian and updating packages during image build helps protect against known vulnerabilities. In production, it's important to rebuild images periodically to incorporate security patches.

# GLOSSARY OF TERMS

| Term | Definition |
|---|---|
| API | Application Programming Interface - Interface for programs to communicate |
| Backend | Part of the system that functions on the server, not visible to the user |
| Bridge | Type of Docker network that connects containers in a private network |
| CMS | Content Management System |
| Container | Standard unit of software that packages code and dependencies |
| Daemon | Process that runs in the background |
| DNS | Domain Name System - System that translates names to IP addresses |
| Dockerfile | File with instructions to build a Docker image |
| FastCGI | Protocol for communication between web server and applications |
| Fork | Copy of a software project that develops independently |
| Frontend | Part of the system visible and interactive for the user |
| HTTPS | HTTP Secure - HTTP encrypted with TLS |
| Image | Read-only template for creating Docker containers |
| Kernel | Operating system core that manages hardware and processes |
| Layer | Layer of a Docker image, result of a Dockerfile instruction |
| LEMP | Linux, NGINX, MySQL/MariaDB, PHP - Web technology stack |
| Mount | Link a filesystem or directory to a point in the directory tree |
| PHP | PHP: Hypertext Preprocessor - Language for web development |
| PID | Process ID - Unique process identifier |
| Port | Number identifying a communication point in a network |
| RDBMS | Relational Database Management System |
| SQL | Structured Query Language - Language for managing databases |
| TLS | Transport Layer Security - Protocol for secure communication |
| Volume | Mechanism to persist data in Docker |
| Worker | Process that performs tasks or handles requests |
| YAML | Yet Another Markup Language - Format for configuration files |

# REFERENCES AND RESOURCES

## Official Documentation

Docker Documentation: docs.docker.com

NGINX Documentation: nginx.org/en/docs

MariaDB Knowledge Base: mariadb.com/kb/en

WordPress Codex: codex.wordpress.org

## Tools

Docker Compose: docs.docker.com/compose

WP-CLI: wp-cli.org

OpenSSL: openssl.org

## Learning Resources

Docker Get Started: docs.docker.com/get-started

NGINX Beginner's Guide: nginx.org/en/docs/beginners_guide.html

PHP-FPM Documentation: php.net/manual/en/install.fpm.php