**CMD vs. ENTRYPOINT, EXPOSE, COPY/ADD, USER & WORKDIR**

**1. Introduction**

* **Goal:** Complete Dockerfile concepts to prepare for Kubernetes.
* **Today’s Topics:**
  + COPY vs. ADD
  + CMD vs. ENTRYPOINT (deep dive)
  + EXPOSE (myth vs. reality)
  + USER & WORKDIR (security best practices)

**2. Key Concepts**

**A.**CMD**vs.**ENTRYPOINT

| **Feature** | **CMD** | **ENTRYPOINT** |
| --- | --- | --- |
| **Overridable?** | Yes (docker run <image> <new-cmd>) | No (by default, but can be forced with --entrypoint) |
| **Use Case** | Default arguments for the container | Fixed commands (e.g., nginx -g "daemon off;") |
| **Best Practice** | Use for default runtime args | Use for immutable commands |

**Demo:**

1. **Overriding**CMD**:**

dockerfile

CMD ["ping", "-c", "4", "google.com"]

bash

docker run my-image ping -c 10 youtube.com # *Overrides CMD*

1. ENTRYPOINT**Lockdown:**

dockerfile

ENTRYPOINT ["ping", "-c", "4"]

CMD ["google.com"]

bash

docker run my-image youtube.com # *CMD is appended to ENTRYPOINT*

**B.**EXPOSE**– Documentation, Not Port Forwarding**

* **Myth:** EXPOSE 80 publishes the port.
* **Reality:** It’s metadata for developers.

dockerfile

EXPOSE 80 # Just a hint: "This container listens on port 80"

* **Actual Port Publishing:**

bash

docker run -p 8080:80 my-image *# Host:8080 → Container:80*

**C.**COPY**vs.**ADD

| **Instruction** | **COPY** | **ADD** |
| --- | --- | --- |
| **Function** | Copies local files | Copies + extracts URLs/tarballs |
| **Use Case** | Static files (e.g., index.html) | Remote resources (e.g., ADD https://example.com/file.tar.gz /app) |

**Demo:**

dockerfile

COPY index.html /var/www/html/ # Local file

ADD https://example.com/terraform.zip /tmp # Remote download

**D.**USER**&**WORKDIR**– Security & Isolation**

1. USER**:** Avoid running as root.

dockerfile

RUN useradd -r -u 1001 appuser

USER appuser # Container runs as non-root

1. WORKDIR**:** Sets the working directory.

dockerfile

WORKDIR /app # All subsequent commands run in /app

**Security Demo:**

Bash

docker exec -it my-container bash

whoami *# Shows "appuser" (not root)*

cd /root *# Permission denied!*

**3. Step-by-Step Demo**

**Step 1: Build a Dockerfile with All Instructions**

dockerfile

FROM ubuntu:latest

WORKDIR /app

RUN apt update && apt install -y nginx

COPY index.html /var/www/html/

ADD https://example.com/terraform.zip /tmp

EXPOSE 80

ENTRYPOINT ["nginx", "-g", "daemon off;"]

**Step 2: Build & Run**

Bash

docker build -t my-app .

docker run -d -p 8080:80 --name web my-app

**Step 3: Verify Security**

Bash

docker exec -it web whoami *# Should NOT be root*

**4. Best Practices**

1. **Always:**
   * Use USER to avoid root.
   * Prefer COPY over ADD unless fetching remote files.
2. **Never:**
   * Rely on EXPOSE for networking—use -p at runtime.
   * Override ENTRYPOINT in production.

**5. Key Takeaways**

* CMD**vs.**ENTRYPOINT**:** Default args vs. immutable commands.
* EXPOSE**≠ Port Forwarding:** It’s just documentation.
* **Security:** Always drop root privileges with USER.

**Structured Notes Format:**

* **Problem/Solution pairs** (e.g., "Myth: EXPOSE publishes ports" → "Reality: It’s metadata").
* **Comparison tables** (CMD vs. ENTRYPOINT, COPY vs. ADD).
* **Code blocks** for commands/Dockerfile snippets.
* **Emphasis on security** (USER, non-root containers).