Docker Container Hardening Guide

1. Use Minimal Base Images

Use lightweight, secure base images like alpine or distroless. Avoid unnecessary tools/utilities inside the image.

2. Run as Non-Root

Never run containers as root. Create and use a non-root user in Dockerfile.

3. Limit Container Capabilities

Drop all Linux capabilities and add only what's needed using --cap-drop and --cap-add flags.

4. Use Read-Only Filesystems

Prevent writing to container filesystem using --read-only.

5. Use Seccomp, AppArmor, SELinux

Apply security profiles like seccomp, AppArmor, or SELinux for runtime protection.

6. Use Multi-Stage Builds

Separate build and runtime stages to exclude unnecessary files and reduce image size.

7. Scan for Vulnerabilities

Use tools like Trivy, Docker Scout, or Grype to scan images regularly.

8. Keep Images Up to Date

Regularly rebuild images to include the latest patches and avoid using :latest in production.

9. Limit Network Exposure

Use isolated Docker networks and avoid using host networking mode.

Docker Container Hardening Guide

10. Avoid Storing Secrets in Images

Never hardcode credentials in Dockerfile. Use a secrets manager.

11. Set Resource Limits

Set memory and CPU limits to prevent resource exhaustion attacks.

12. Audit Docker Daemon and Hosts

Harden the host OS, secure the Docker socket, and disable remote API without TLS.

Recommended Tools

- Trivy: Vulnerability scanner for Docker images.
- Docker Scout: Official Docker image scanner.
- Grype: CLI vulnerability scanner.
- Dockle: Linter for Dockerfile security best practices.
- Clair: Static vulnerability analysis.
- Sysdig Falco: Real-time security monitoring and threat detection.