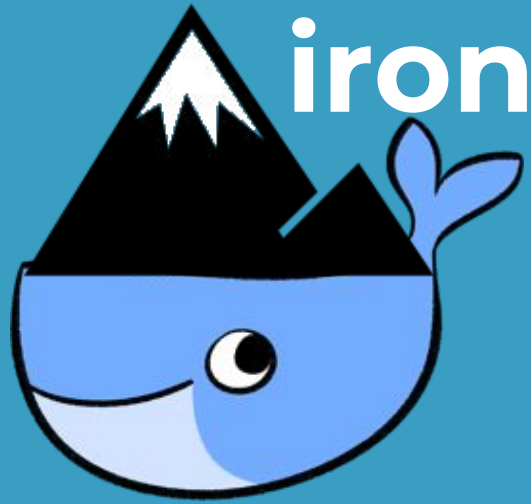


ironing out Docker

at ironPeak services

1. $ whoami

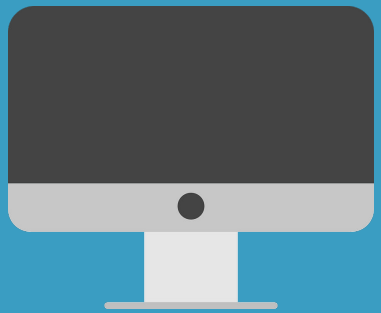
Niels Hofmans

role	Independent Cybersecurity Consultant
work	Code Security, App Security, Hardening, F5 BIG-IP
interest	Go, Docker, Cloud, Media
contact	hello@ironpeak.be
github	github.com/HazCod

2. $ tree

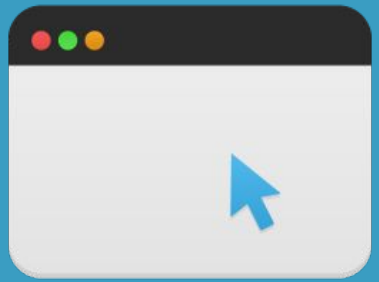
user

host



image

Runtime



3. $ client

The Client (you!)

- Hidden attack surface

- Several attack vectors
  - Phishing
  - Hardware
  - Software
  - Open-Source

- Social Networks

- Reused/shared Passwords

3. $ client

The Client (you!)

- Awareness
- Phishing
  - Common Sense
  - E-mail headers, content, DMARC
- Hardware
  - Disk encryption
  - Lock-down BIOS/SMC
  - Trustless with 2FA
  - Lock your session

3. $ client

The Client (you!)

- Software
  - OS Hardening
  - Non-privileged User
  - Firewall
  - Patching
  - Verify & Tag Open-Source

- Additional
  - Information leakage: e.g. LinkedIn, Github
  - Password manager with 2FA

4. $ host

Host hardening

- CIS Benchmarks
- Firewall

Daemon hardening

- CIS Benchmarks, docker-bench-security, kube-bench
- User Namespace Remapping
- Live Restore
- No experimental features
- Swarm autolock
- Kernel hardening: github.com/google/gvisor
- Enable SELinux/AppArmor + seccomp

4. $ host

Daemon Access

- UNIX Socket over SSH
- HTTP+TLS auth

Host Auditing

- Off-site log server over TLS/SSH
- Log forging / Denial of Service
- Audit tracing

  e.g. sysdig.org + falco.org, github.com/netdata/netdata

Private Registry

- client: DOCKER_CONTENT_TRUST=1
- daemon: content_trust: enforced

5. $ image

- **DIY** & **Commercial**
  - Base images: alpine (!), minideb, centos
    github.com/GoogleContainerTools/distroless
  - docker-slim
  - Image Signing

- **Leakage**
  - .dockerignore
  - docker secrets/vault

- Remove **defaults**
  - Network: bridge
  - Storage: AUFS

5. $ image

Dockerfile

- Linters; hadolint, …
- Pin os/package versions
- FROM & Multi-stage builds
- Least Privilege
  - $user & root without shells
  - tighten permissions
  - remove unnecessary tooling
- USER
- COPY --chown=x:x instead of ADD
- Scan for package vulnerabilities

5. $ image.findWally()

```
1  FROM golang:latest
2
3  ADD main.go /
4
5  RUN go build -o /app /main.go
6
7  CMD /main
```

5. $ image.findWally()

```
1  FROM golang:latest
2
3  ADD main.go /
4
5  RUN go build -o /app /main.go
6  USER?
7  CMD /main
```

5. $ image.getFixed()

```
1   FROM golang:1.12.1-alpine3.9 AS builder
2   RUN adduser -s /bin/true -u 1000 -D -h / app \
3       && sed -i -r "/^(app|root)/!d" /etc/group \
4       && sed -i -r "/^(app|root)/!d" /etc/passwd \
5       && sed -i -r 's#^(.*):[^:]*$#\1:/sbin/nologin#' /etc/passwd
6   COPY main.go /
7   RUN go build -ldflags '-w -s -extldflags "-static"' -o /app /main.go
8
9   FROM scratch
10  COPY --from=builder /etc/passwd /etc/shadow /etc/
11  COPY --from=builder /etc/ssl/certs/ca-certificates.crt /etc/ssl/certs/
12  COPY --from=builder /app /app
13  USER app
14  CMD ["/app"]
```

6. $ runtime: container

Container Runtime Properties

- Read-Only filesystem
- mounts: noexec, nodev, nosuid, mode, size, uid/gid
- pids-limit=1
- cgroup limits: cpu, memory/swap, network, size, disk i/o, ...
- restart: on-failure:5
- cap_drop: ALL
- security_opt:
    - no_new_privileges
    - SELinux/AppArmor + seccomp
- Environment variables vs. Secrets

6. $ runtime: app

Application Security

- OWASP ASVS: Level 1 - Level 3
  - web: github.com/OWASP/ASVS
  - mobile: github.com/OWASP/owasp-masvs
- Static Application Security Testing (SAST)
  - linters
  - OSS + commercial
- Dynamic Application Security Testing (DAST)
  - OpenVAS, OWASP ZAP, …

- Training & Awareness!

7. $ exit

https://ironpeak.be/slides/190319-ironing-out-docker.pdf