BeOne stuttgart

DOCKER CONTAINER HARDENING

Container hinter Schloss und Riegel

Dr. Stefan Schlott (BeOne Stuttgart GmbH)

@_skyr skyr@chaos.social

ABOUT.TXT

Stefan Schlott, BeOne Stuttgart GmbH

Java-Entwickler, Scala-Enthusiast, Linux-Jünger

Seit jeher begeistert für Security und Privacy

DEFENSE IN DEPTH

Warum wir das überhaupt machen...

WAS WILL EIN ANGREIFER?

System war eigentliches Ziel? ☠

System war „Patient 0“ oder „Zwischenstop”?

→ Information Gathering

→ Lateral Movement

→ Persistenz

ABGRENZUNG

Worum es heute nicht geht:

Verwundbarkeit der eigenen Anwendung bzw. der verwendeten Bibliotheken

→ SpotBugs, OWASP Dependency Check, etc.

Veraltete/anfällige Software im Containerimage

→ clair, trivy, etc.

→ Container von DockerHub vs. Selberbauen

WAS SIND CONTAINER?

Ein technischer Blick

VERSCHIEDENE IMPLEMENTIERUNGEN

→ Docker ←

Podman

LXC

CRI-O

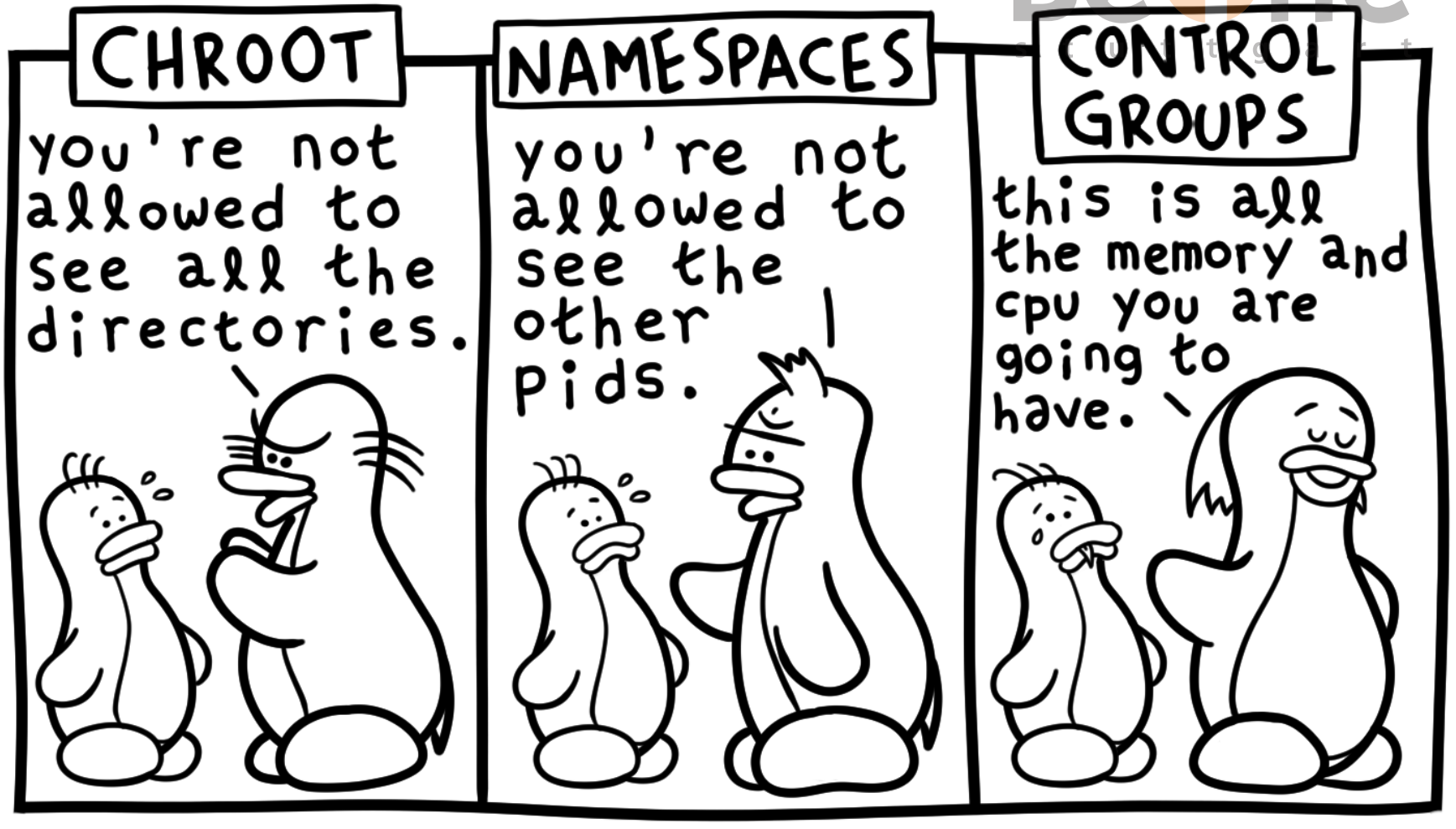
~~rkt~~

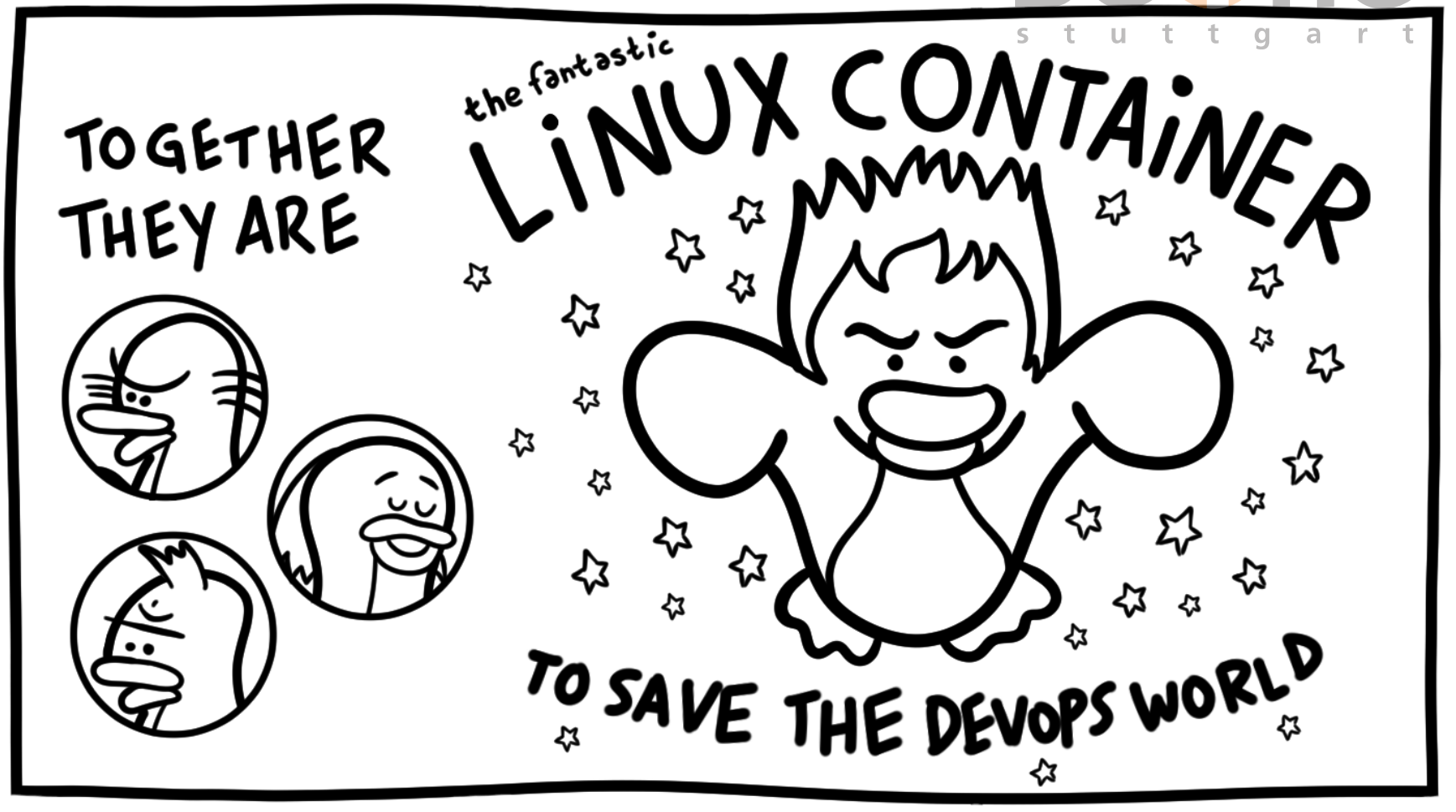
Be One
stuttgart

CONTAINER ≠ VM

Leben im selben Kernel

Durch verschiedene Techniken des OS in ihrer Sichtweise eingeschränkt





NAMESPACES

Nicht nur Prozessliste / PIDs (**pid / pid_for_children**)

Control group (cgroup): Ressourcen-Beschränkung (CPU, disk I/O, network I/O)

Interprozess-Kommunikation (**ipc**): Signale, Shared Memory, etc. begrenzt

Mount (**mnt**): Individuelle Sicht aufs Dateisystem / Mountpoints

Network (**net**): „Virtueller“ Netzwerk-Stack

User ID (user): Abbildung von IDs innerhalb des Namespaces auf „echte” User-IDs

Host-/Domainname (**uts**)


LESSON #1
Don't run as root
(or at least drop privileges)
Kein su/sudo-Zugang zum root-User
USER . . . - Eintrag im Dockerfile
I AM ROOT

WEITERE EIGENSCHAFTEN

Capabilities: Feingranularere Einschränkungen als „root“ vs „nicht root”

AppArmor-Profil: Einschränkungen auf Verzeichnis-Teilbäume (lesen/schreiben/kein Zugriff) und andere Ressourcen

Seccomp-Profil: (Lowlevel)-Einschränkungen von syscalls

DIE DOCKER-SOCKET

The keys to the kingdom

BeOne stuttgart

DOCKER-SOCKET

Schnittstelle zum Docker Daemon

Genutzt u.a. vom Kommandozeilen-Tool

Freigabemöglichkeit fürs Netz

Gern in Container gereicht für z.B. CI-Server, Testcontainers, etc.

ZUGRIFF: JSON-API

`docker` Kommando nicht erforderlich - Ansteuerung über JSON-API:

```
docker images
```

ist äquivalent zu

```
curl --unix-socket /var/run/docker.sock -X GET
http://localhost/v1.24/images/json
```

BeOne stuttgart

THE KEYS TO THE KINGDOM

Beliebige Mounts, Start priviligierter Container

Schreibzugriff auf Docker-Socket = root-Äquivalent!

Mitglieder der Usergruppe `docker` = root-Äquivalent!

(ohne weitere Schutzmaßnahmen wie User Mapping)

--PRIVILEGED

If it doesn't work, use more force...

DOCKER --PRIVILEGED

„Probier' mal mit `--privileged`“

Was sagt die Doku?

a “privileged” container is given access to all devices (...) as well as set some configuration in AppArmor or SELinux to allow the container nearly all the same access to the host as processes running outside containers on the host

MORE INFORMATION ON THE BLOG

Docker can now run within Docker

JÉRÔME PETAZZONI
Sep 05 2013

One of the (many!) features of Docker 0.6 is the new "privileged" mode for containers. It allows you to run some containers with (almost) all the capabilities of their host machine, regarding kernel features and device access.

Be1ne
stuttgart

THE KEYS TO THE KINGDOM (AGAIN)

Weitgehende Interaktion mit dem Kernel (z.B. auch Kernelmodule laden!)

`--privileged` = root-Äquivalent!

DOCKER BENCH FOR SECURITY

Der Baseline-Check

PRÜFT STANDARD-OPTIONEN

Docker Bench for Security ausführen, wenn Container gestartet sind

Details zu Tests: Shellskripte im Verzeichnis `test`

Tests: Teils eher heuristisch

Hinweise zur Systeminstallation (Auditing, etc.): Für heute out of scope

Hinweise zu Security-Options, Privilegien, Ressourcen-Limits (DoS-Schutz)

CAPABILITIES REDUZIEREN

STANDARD-CAPABILITIES

Capability Key	Capability Description
SETGID	Make arbitrary manipulations of process GIDs and supplementary GID list.
SETUID	Make arbitrary manipulations of process UIDs.
SETPCAP	Modify process capabilities.
NET_BIND_SERVICE	Bind a socket to internet domain privileged ports (port numbers less than 1024).
NET_RAW	Use RAW and PACKET sockets.
DAC_OVERRIDE	Bypass file read, write, and execute permission checks.
FOWNER	Bypass permission checks on operations that normally require the file system UID of the process to match the UID of the file.
SETFCAP	Set file capabilities.
KILL	Bypass permission checks for sending signals.
AUDIT_WRITE	Write records to kernel auditing log.
CHOWN	Make arbitrary changes to file UIDs and GIDs (see chown(2)).
FSETID	Don’t clear set-user-ID and set-group-ID permission bits when a file is modified.
SYS_CHROOT	Use chroot(2), change root directory.
MKNOD	Create special files using mknod(2).

Be1ne
stuttgart

FAST IMMER UNNÖTIG!

Start mit `--cap-drop all`

Anschließend - falls nötig - mit `--cap-add ...` selektiv hinzufügen

Be1ne
stuttgart

NO_NEW_PRIVS

Start mit `--security-opt=no-new-privileges`

Bonus (anderer Mechanismus, aber seelenverwandt): Prozess verbieten, seine Privilegien zu erhöhen

Wie kommt ein Prozess zu sowas!?

suid-Binaries!

READ-ONLY CONTAINER

Be One stuttgart

ANGREIFER-PERSISTENZ ERSCHWEREN

Start mit `--read-only`: Filesystem des Containers ist schreibgeschützt

Mit `--tmpfs pfad`: Schreiben in Ramdisk erlauben (keine Executables erlaubt!)

Schreiben nur in Volumes

VOLUMES WORKAROUND

Volume bind mounts: Keine Option für `noexec`

Workarounds:

Volumes in eigener Partition, die im Host mit noexec mounten

Bind Mount im System:

```
mount -o bind,noexec /../vol /../vol-noexec
```

...schützt vor direkter Ausführung - Skriptinterpreter können natürlich nach wie vor Skripte starten

NETZ-ISOLATION

Nicht jeder muss mit jedem reden

REICHWEITEN-BEGRENZUNG

Häufige Konstellation: Mehrere Container mit Einzelfunktion (Datenbank, Message Queue, etc.), aber nur einer mit „Außensichtbarkeit“ (Web Application)

Keine Verbindung nach außen!

Keine Verbindung zu allen anderen Containern!

→ Erschwert Daten-Exfiltration

→ Erschwert Lateral Movement

DOCKER NETWORKS

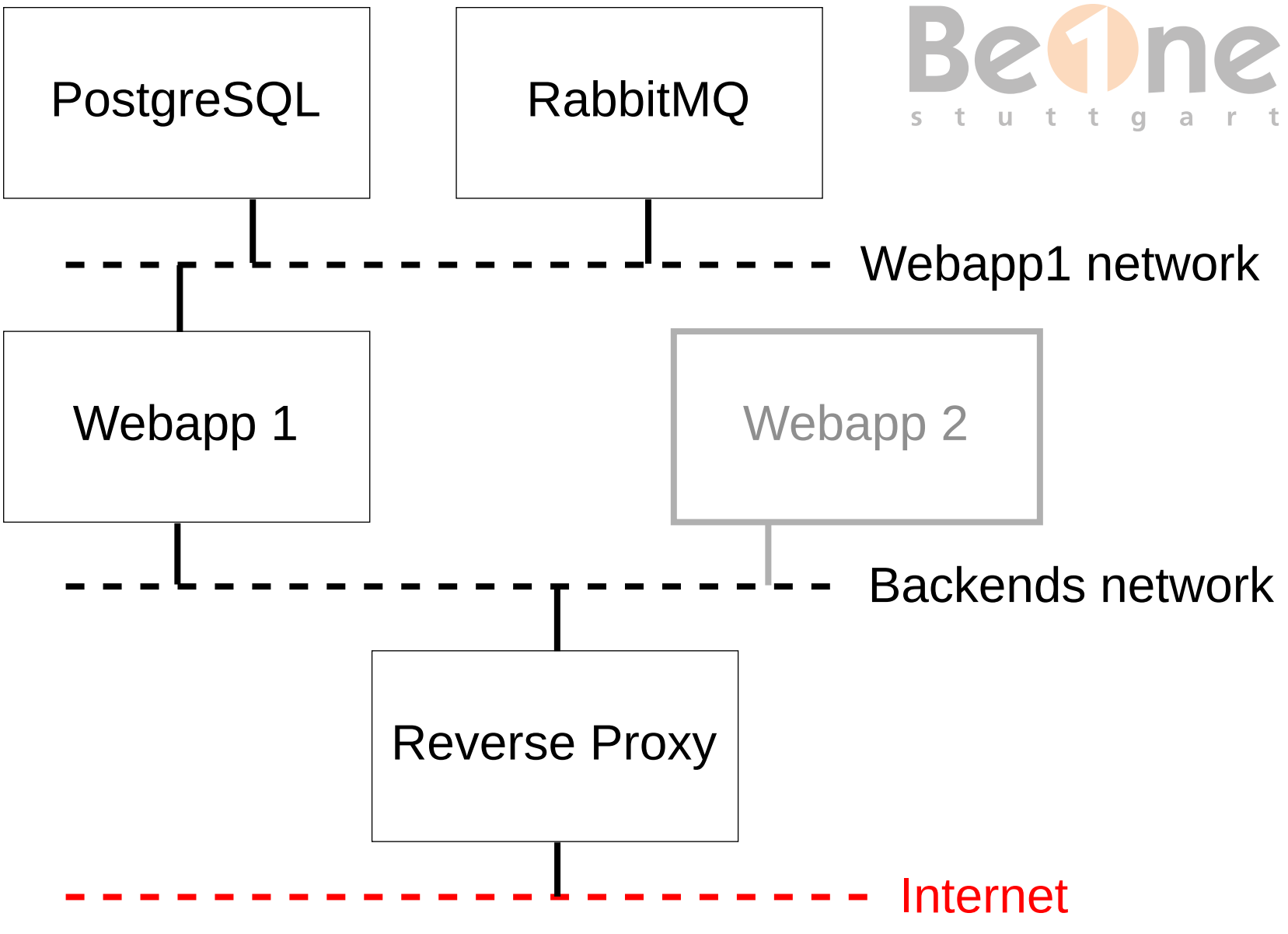
Leider keine „Firewall-Funktionalität”

Container können mehreren Netzen angehören. Typen:

none - kein Netzwerkzugriff

bridge - Virtuelle Bridges
Option „internal“: Kein Zugriff nach außen

⚠ host ⚠ - Direkter Zugriff auf die Netzwerk-Interfaces


PostgreSQL
RabbitMQ
Webapp1 network
Webapp 1
Webapp 2
Backends network
Reverse Proxy
Internet

BEISPIEL MIT DOCKER-COMPOSE


```yaml
version: '3'
services:
  webapp:
    image: petclinic
    networks:
      - backends
    labels: # traefik-autoconfig
  proxy:
    image: "traefik:v2.1"
    networks:
      - backends
      - internet
    ports:
      - "80:80"
    volumes:
      - "/var/run/docker.sock:/var/run/docker.sock:ro"
      - "traefik-config:/etc/traefik"
networks:
  backends:
    driver: bridge
    internal: true
  internet:
    driver: bridge
```

DOCKER-IMAGES MINIMIEREN

Look Ma, no shell!

BeOne stuttgart

DOCKER SLIM

Opensource-Tool: Größe minimieren (Löschen nicht benutzer Files), Erstellen von SecComp- und AppArmor-Profilen

Startet Container, „klicken“ auf Webanwendung

Parallel: Tracing von Zugriffen und benötigten Capabilities/Rechten

```
state=image.inspection.start
image id=sha256:7e... size.bytes=226244354 size.human=226 MB
image.users exe='petclinic' all='petclinic'
image.stack index=0 name='alpine:3.11' id='sha256:e7...'
image.stack index=1 name='alpine-base:latest' id='sha256:dd...'
image.stack index=2 name='alpine-jre11:latest' id='sha256:ad...'
image.stack index=3 name='petclinic:latest' id='sha256:7e...'
image.exposed_ports list='8080'
(...)
state=completed
results status='MINIFIED BY 3.19X [226244354 (226 MB) => 70980544 (71 MB)]'
results  image.name=petclinic.slim image.size='71 MB' data=true
results  artifacts.location='/home/user/.../artifacts'
results  artifacts.report=creport.json
results  artifacts.dockerfile.original=Dockerfile.fat
results  artifacts.dockerfile.new=Dockerfile
results  artifacts.seccomp=petclinic-seccomp.json
results  artifacts.apparmor=petclinic-apparmor-profile
```

```
profile petclinic-apparmor-profile
    flags=(attach_disconnected,mediate_deleted) {
  network,
  /usr/lib/jvm/java-11-openjdk/bin/java rix,
  /etc/passwd r,
  /lib/ld-musl-x86_64.so.1 r,
  /lib/libc.musl-x86_64.so.1 r,
  /lib/libz.so.1 r,
  /lib/libz.so.1.2.11 r,
  /srv/petclinic.jar r,
  /usr/bin/java r,
  /usr/lib/jvm/default-jvm r,
  /usr/lib/jvm/java-11-openjdk/jre r,
  /usr/lib/jvm/java-11-openjdk/lib/jli/libjli.so r,
  /usr/lib/jvm/java-11-openjdk/lib/jvm.cfg r,
  /usr/lib/jvm/java-11-openjdk/lib/libjava.so r,
  /usr/lib/jvm/java-11-openjdk/lib/libverify.so r,
  /usr/lib/jvm/java-11-openjdk/lib/server/libjvm.so r,
}
```

→ trotz Alpine Linux: ~140 MB reduziert, Security-Profile automatisch generiert

Heuristik - aber eine Vorlage für Handkonfiguration

Be One stuttgart

LOOK MA, NO SHELL!

Dockerfile: CMD vs ENTRYPOINT

ENTRYPOINT: Kommando, das beim Start ausgeführt wird

CMD: Dessen Aufruf-Parameter

ENTRYPOINT default: `/bin/sh`

→ Wenn sonst keine Shellskripte verwendet werden: Hier Shell „wegoptimieren“ und Shell aus Container löschen

Be One
stuttgart

DISTROLESS

Google-Projekt: Sprachspezifische, minimale Container

Ausschließlich Binaries/Libraries für den Sprachstart

Kein Paketmanager, keine Shell, etc.

→ minimale Größe, minimale Angriffsfläche

→ Frage nach Security Patches / Selberbauen

APPARMOR-PROFILE

Be One stuttgart

MANDATORY ACCESS CONTROL

Capabilities

Netzwerk (Protokolltypen)

Filesystem (feingranular - lesen, schreiben, ausführen, Hardlinks)

Beim Docker-Start: `--security-opt="apparmor:..."`

Be1ne
stuttgart

PROFIL ERZEUGEN

Handarbeit

AppArmor-Tools: „Training“ mit `aa-genprof`

Vereinfachte Konfiguration mit bane

APPARMOR - FAT JAR

```
#include <tunables/global>

profile fat-jar flags=(attach_disconnected,mediate_deleted) {
  #include <abstractions/base>
  #include <abstractions/user-tmp>

  network inet tcp,
  deny network raw,
  deny network packet,

  /tmp/** rw,
  /etc/** r,
  /srv/** r,
  /usr/bin/java rmix,
  /usr/lib/jvm/** rmix,
}
```

FAZIT

FAZIT

Kein root im Container!

Whitelisting / Ausdünnen:
Capabilities, Containerinhalt, Schreibrechte

Netzwerk: Außenkommunikation durch „stapeln” limitieren

stefan.schlott@beone-stuttgart.de
@_skyr skyr@chaos.social