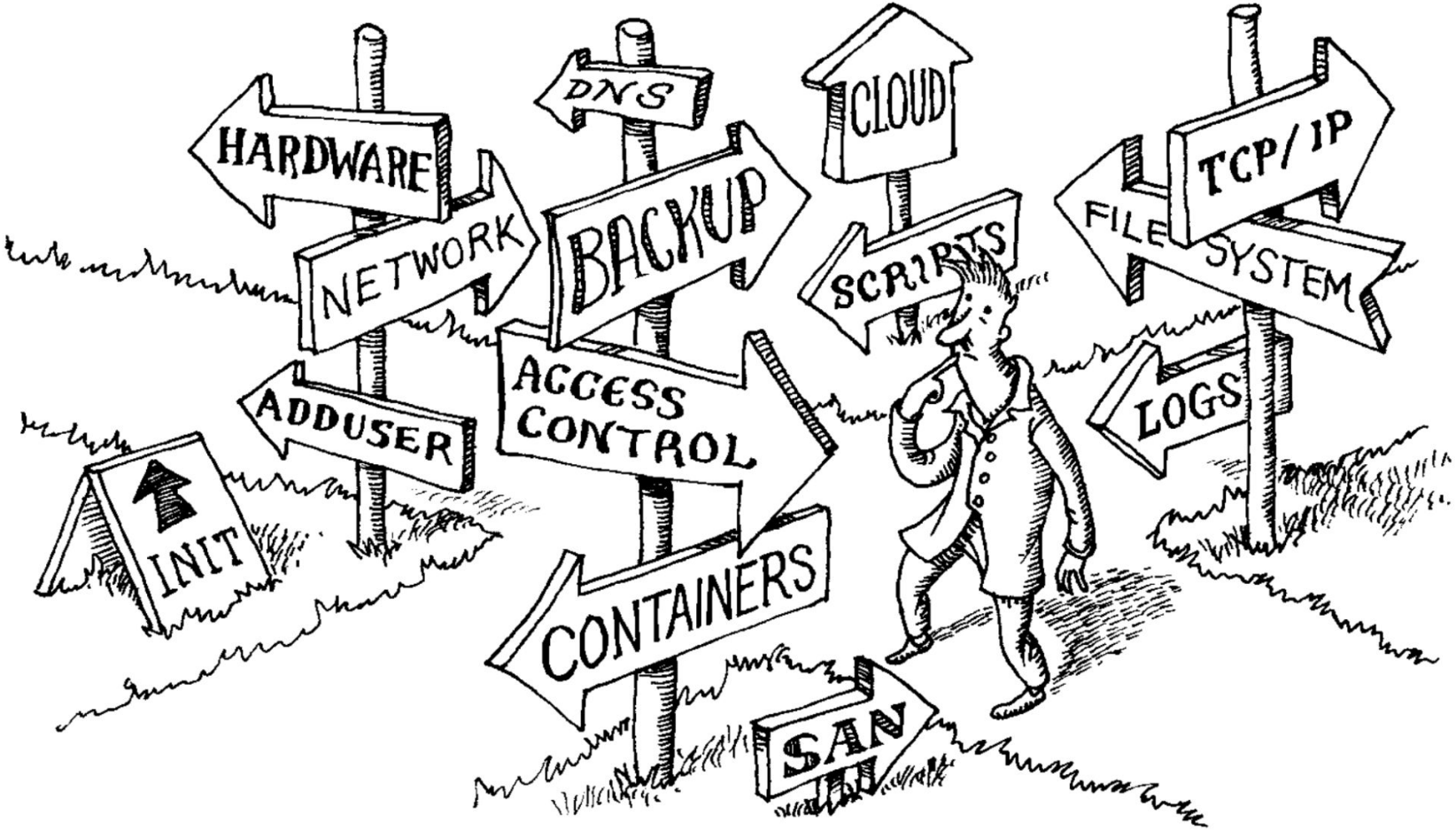
UNIX Systemadministration: Eine Einführung

3 März 2024

about://me

- Seit einem ¾ Jahr Systemadministratorin bei INET
- Viel Legacy Infrastruktur übernommen
- Inzwischen sehr viel der Legacy Infrastruktur aufgeräumt
- Entsprechend viel gelernt
- Ziel: Möglichst viel Wissen weitergeben


HARDWARE
NETWORK
ADDUSER
INIT
DNS
BACKUP
ACCESS CONTROL
CONTAINERS
SAN
CLOUD
SCRIPTS
TCP/IP
FILE SYSTEM
LOGS

Systemadministration

- Umfasst sehr viele Tasks
  - Hardware
  - Software (Services)
  - Security
  - Backups
  - Monitoring
  - Automatisierung
  - Dokumentation
  - Tech-Support
  - …

Hardware

Server

- Beschreibt eine physische Maschine
- Sind immer an
- Halten lange
- Haben *viele* Nutzer
- Verschiedene Strategien zur Allokation
  - Riesen-Host: Alle Anwendungen auf einem Server
  - Unikate: Einzigartige Maschinen, welche unterschiedlich konfiguriert sind
  - VM-Cluster: Viele VMs, jede mit einer einzigen Aufgabe
  - Container-Hosts: n gleichartige Maschinen, auf die m Services verteilt sind

javaws viewer\(1\).j ~/Downloads

bowlxen-4-ipmi, PowerEdge R620, User: root,…

File View Macros Tools Power Next Boot Virtual Media Help

<<< Welcome to NixOS 23.11.20240213.01885a0 (x86_64) - tty1 >>>

Integrated Dell Remote Access Controller 7 | Enterprise

Support | Dell TechCenter | About | Logout

System
PowerEdge R620
root , Admin

- Overview
  - Server
  - Logs
  - Power / Thermal
  - Virtual Console
  - Alerts
  - Setup
  - Troubleshooting
  - Licenses
  - Intrusion
- iDRAC Settings
- Hardware
- Storage
- Host OS

Power Monitoring | Power Configuration | Voltages | Temperatures

Power Monitoring

Jump to: Power | Present Reading | Power Supply Unit Readings | Cumulative Reading | Historical Peaks | Historical Trends | System Headroom

Status

Power Control: Reset System (warm boot)

Health	OK
Server Status	ON
Present Reading	98 Watts (10.89% Capacity)
Power Supply Redundancy	Full
Active Power Cap Policy	No Power Cap Policy Set

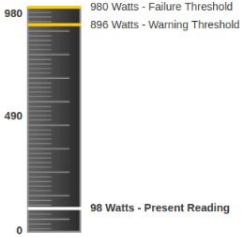
Apply

Power

▲ Back to Top

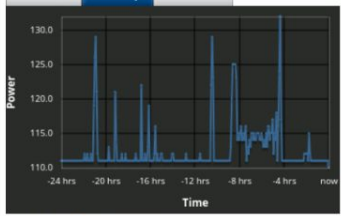
Present

Present Reading: 98 Watts (10.89% Capacity)



Past

Last Hour | Last Day | Last Week



Power Units (•) Watts ( ) Amps

Present Power Reading and Thresholds

▲ Back to Top

Status	Probe Name	Present Reading	Warning Threshold	Failure Threshold
✓	System Board Pwr Consumption	98 W 334 BTU/hr	896 W 3058 BTU/hr	980 W 3345 BTU/hr

Apply

Server Festplatten

- HDD vs. SSD
- HDDs sind billiger, SSDs sind performanter
- SSDs sind *viel* besser im random-access
  - Z.B. für Mail-Server sehr zu empfehlen
- SATA vs. SAS vs. NVME
  - 6Gb/s vs. 12 Gb/s vs. >32Gb/s
  - SAS Platten sind üblicherweise teurer, dafür performanter und zuverlässiger

Server-Dateisystem

- Was für Features könnte man wollen?
  - Crash-Recovery
  - Filesystem-RAID
  - Error detection & recovery
  - Copy-on-write
  - Snapshots
  - Compression
  - Deduplication

Server-Dateisystem Vergleich

	ext4	XFS	Btrfs	ZFS	NTFS	APFS
Crash Recovery	Journaling	Journaling	Ja	Ja	Journaling	Ja
Filesystem-RAID	Nein	Nein	RAID0 & RAID1	Ja	Nein	Nein
Error recovery	Nein	Nein	Ja	Ja	Nein	Nein
Copy-on-write	Nein	Ja	Ja	Ja	Nein	Ja
Snapshots	Nein	Nein	Ja	Ja	Nein	Ja
Compression	Nein	Nein	Zlib, LZO, Zstd	gzip, lz4, Zstd, ...	LZ77, LZNT1	Ja
Deduplication	Nein	Ja	Ja	Ja	Ja (Server)	Ja

Basierend auf https://en.wikipedia.org/wiki/Comparison_of_file_systems

Security

Access Control (Filesystem)

- Findet in einer FS-unabhängigen Schicht (VFS) statt
- Jede Datei hat einen Owner und eine Gruppe
- Permissions sind durch 9 Bits konfiguriert (Read, Write, Execute)

```
-rw-rw-rw- 1 emily emily  26 Jan  1 19:22 all.txt
-rw-rw---- 1 emily admins 40 Jan  1 19:32 group.txt
-rw------- 1 emily emily  22 Jan  1 19:32 owner.txt
```

- Das VFS kann als sicher angenommen werden
- Aber lieber mehrere Layer an Security! (z.B. Container)

Security: SSH

- Wie greife ich auf die Kommandozeile meines Servers zu?
  - **Secure** Shell, standardmäßig auf Port 22
- Authentifizierung über z.B. Passwort oder Public-Key
  - Besser nur Public-Key erlauben
- Keyformat: ed25519 für kürzere Keys
- Keys sollten mit Passwort geschützt sein
  - Sonst kann jedes Programm mit execute Rechten deine Keys lesen! (z.B. pip install)
- ssh-audit zur Inspektion von Schwachstellen in der sshd_config

Security: Passwörter

- Es gibt viele Admin/Service Accounts
- Für diese ist es wichtig sichere Passwörter zu haben
  - Möglichst Lang, zufällig, einzigartig
- Passwort Manager sind zu empfehlen (KeePassXC, Pass)
  - Nur noch ein passwort merken!
- Wie lerne ich komplizierte Passwörter?
  - Passwort von root setzen
  - sudo mit targetpw: Jedes Mal das neue Passwort eingeben
- Mehr Details gibt es in einem späteren TechTalk!

Backups

Warum machen wir Backups?

- Wenn was passiert ist das Geschrei groß
- Warum fallen Systeme aus?
  - Hardware Failure
  - Bit-Rot
  - Ransomware
- Welche Daten sichern?
  - Alles, was State ist
  - **Achtung**: z.B. bei Datenbanken (Postgresql) ist kein Hot-Backup möglich
- “Kein Backup, kein Mitleid”
- Am Besten tägliche Backups

Mögliche Backup-Medien

- HDD
  - Klassisch, Gut, Billig
  - Muss mit RAID gesichert werden
- Tape Storage
  - Große Einmalinvestition
  - Auf viel Speicher gerechnet deutlich billiger
  - Read-Only Schalter
  - Schlechte Random-Access performance
- Cloud
  - Kostet deutlich mehr
  - Verantwortung ist outgesourced
  - Benötigt guten Up- und Downlink
  - GitHub, Google Drive, DropBox, …

Redundanz

- Wie Backupe ich meinen Backup Server?
- Mehr Backup Server!
  - 3,2,1 Regel: 3× Daten auf 2 Medien mit 1 off-site Kopie
- Testen! Kann ich die Backups wiederherstellen?

Wie erstelle ich Backups?

Client (Push)

- Der Client initiiert das Backup

Vorteile

- Einfach zu implementieren
- Viel gute Software
- SSH-Keys liegen auf den Clients
- Getrennte Accounts für verschiedene Clients
- NAT Traversal

Nachteile

- Lastspitzen
- Clients kontrollieren den Backup-Schedule
- Jeder Client muss konfiguriert werden
- Möglicherweise inkonsistente Configs

Server (Pull)

- Der Server initiiert das Backup

Vorteile

- Zentralisierte Verwaltung
- Skalierbar durch wenig Konfiguration
- Einfaches Monitoring: Server weiß alles
- Übersichtlichkeit: Du siehst alles

Nachteile

- Es gibt einen “master” SSH-Key
- Wenig Software mit allen Features
- Single Point of Failure
- Die Clients müssen immer online sein
- Die Clients brauchen eine statische IP

Client Backup Software

- Filesystem Snapshots
  - Btrfs, zfs
  - Hilft nicht gegen Plattenausfall
- Rsync
  - Gut geeignet für viele kleine Dateien
  - Delta Algorithmus
  - Keine Encryption
  - Backup-Skript selbst schreiben
- Tar
  - Gute Auslastung des Netzwerks
  - Encryption, Compression
  - Möglichkeit für inkrementelle Backups
  - Es dauert das Archiv zu erstellen
  - Nicht so effizient wie rsync
- Borg
  - Speed, Encryption, SSH
  - Deduplication, Encryption, Compression
  - Schnell
  - Nur CLI
- Duplicati
  - Cloud: Google Drive, OneDrive, WebDAV
  - Web UI
  - Funktioniert mit OAuth
  - Daten sind nicht “easily” accessible

Persönliche Empfehlung: Kopia

- Eine Policy fürs gesamte System (jeden Ordner)
  - Wie oft Backups gemacht werden, wie lange sie behalten werden, etc.
- Viele Möglichkeiten für Server
  - SFTP (ssh), WebDAV, Google Cloud, Amazon S3, …
- Many-to-Many Architektur
- Verschlüsselt (AES-256)
- Kompression: pgzip, lz4, zstd, …
- Error Correction gegen Hardware Ausfall
- *Schnell*: 250k Files (23GB) in 12s
- Streaming von Backups mit ~30mbps
- Alternativ ist Borg auch sehr solide

Server Backup Software

- Selbstgeschriebene Rsync Skripte
  - Viel manuelle Skripterei
  - Funktion nicht garantiert
- Rsnapshot
  - Rsync über deklarative Config
  - Staggered File Versioning
  - Durch `backup_script` auch Datenbank Backup möglich
  - Keine Web UI
- BackupPC
  - Frontend für rsync
  - Speichert die Checksummen
  - Mails, Multi-User, Monitoring
  - Kein Staggered File Versioning
- UrBackup
  - Benötigt eigene Client-Software
  - Benötigt eigene Ports
  - Repliziert das Dateisystem des Servers mit Hardlinks
    - Dateien sind Verfügbar
  - Duplizierte Dateien von verschiedenen Clients werden nur 1× gespeichert
- Bacula
  - Sehr enterprisig
  - Hat (in gekauft) alle features
  - Sehr kompliziert
  - Hat 5 verschiedene Docker Container

Backup: Monitoring

Last backup

client_name	last_backup	last_backup_since
admin	2024-03-09 20:14:45	22.1 hour
authentication	2024-03-09 20:22:05	22.0 hour
en-observertory	2024-03-09 20:22:16	22.0 hour
nixie	2024-03-09 21:06:09	21.2 hour

Last backup status

client_name	backup_status
admin	OK
authentication	OK
en-observertory	OK
nixie	OK

Client status

client_name	client_status
admin	Online
authentication	Online
en-observertory	Online
nixie	Online

Total backup size

client_name	total_backup_size
nixie	237 GiB
admin	2.23 GiB
en-observertory	522 MiB
authentication	12.2 MiB

Total backup count

client_name	total_backup_count
admin	16
authentication	16
en-observertory	16
nixie	16

Client details

client_name	client_os_version	client_version
en-observertory	NixOS 23.11 (Tapir); Kernel 6.1.7...	2.5.24
admin	NixOS 23.11 (Tapir); Kernel 6.1.7...	2.5.24
authentication	NixOS 23.11 (Tapir); Kernel 6.1.7...	2.5.24
nixie	NixOS 23.11 (Tapir); Kernel 6.1.7...	2.5.24

Total backup size

Total backup count

Client status (online)

Moderne Systemadministration mit NixOS und Docker

Warum NixOS?

- nix: Funktionaler Paketmanager ohne Seiteneffekte
- Eine Config, die das gesamte System deklarativ beschreibt
  - Versioniert durch git, für die Ewigkeit
- Reproduzierbarkeit: Mit der gleichen Konfiguration wird *gleiche* System gebaut
- Alle Dateien sind unter /nix/store/{sha256-hash}-name-version/...
  - Mehrere Versionen des gleichen Pakets können koexistieren → keine kollisionen
  - Alle benötigten Dateien werden gesymlinked
- Atomische Upgrades & Rollbacks
  - nixos-rebuild switch --rollback
  - Vorherige Konfigurationen können über GRUB gebooted werden
- ISO-Datei aus Config erstellen (zum installieren)

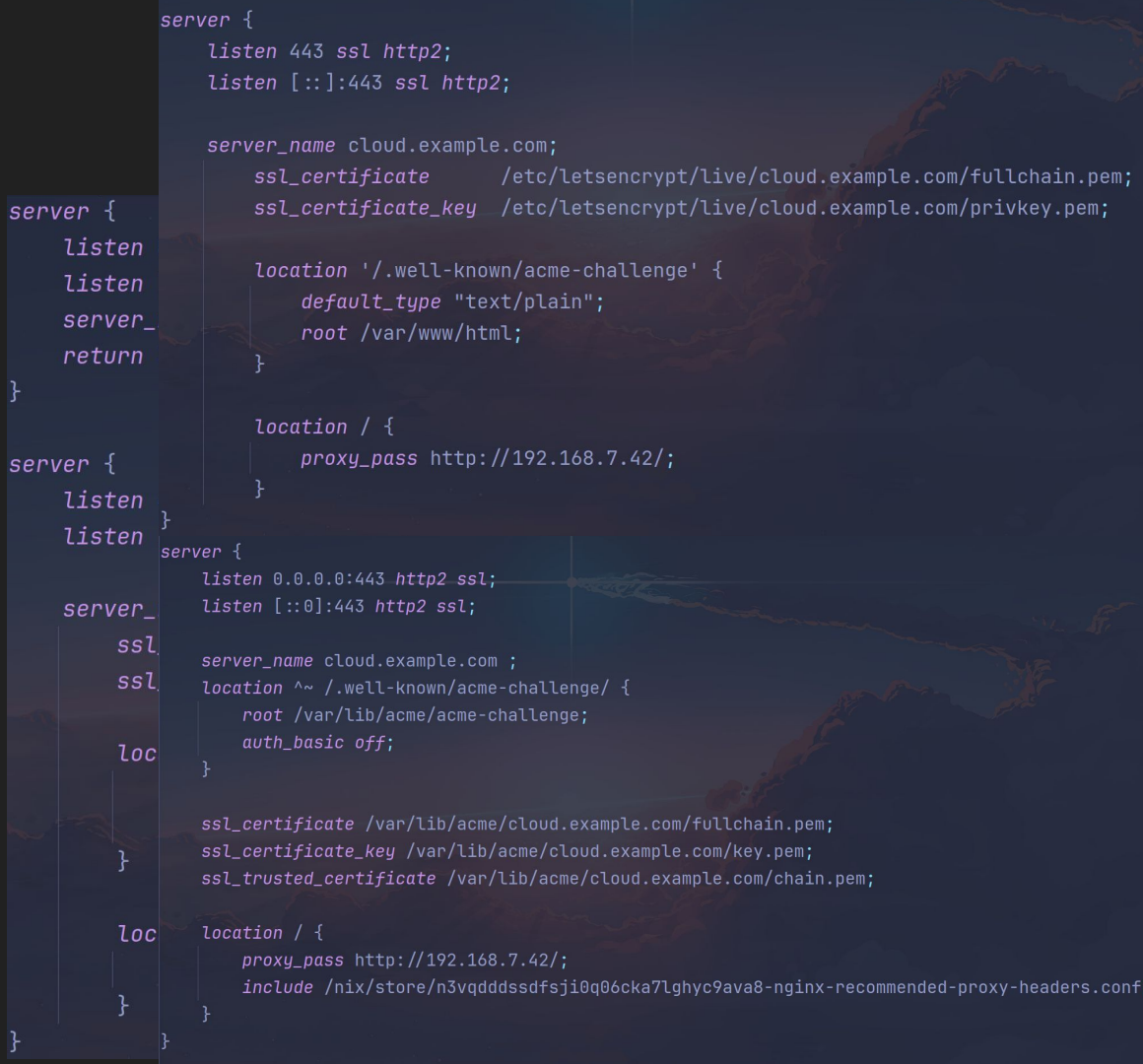
NixOS: Deklarative Config

```
cfg = {
  services.wiki-js = {
    enable = true;

    settings.db = {
      db = "wiki-js";
      host = "/run/postgresql";
      user = "wiki-js";
    };
  };
};
```

```
cfg.services.hedgedoc = {
  enable = true;
  environmentFile = config.age.secrets.HedgeDoc_EnvironmentFile.path;

  settings = {
    domain = "${SUBDOMAIN}.${config.domainName}";
    allowOrigin = [ "localhost" "${SUBDOMAIN}.${config.domainName}" ];
    host = "0.0.0.0";
    protocolUseSSL = true;

    db = {
      dialect = "postgres";
      host = "/run/postgresql";
    };

    # Users and Permissions
    email = false;
    allowAnonymous = false;
    allowEmailRegister = false;
    allowFreeURL = true;
    requireFreeURLAuthentication = true;
    defaultPermission = "limited";
```

NixOS: Nginx

- Nginx setup mit LetsEncrypt + ProxyPass an einen Container:
- Stattdessen im Imperativen System:
- Generierte Config von NixOS:



NixOS: Automatisierung

- Beispiel: Mein eigenes privates Repo

Language	files	blank	comment	code
Nix	49	228	64	1541
Bourne Shell	14	145	70	491
SUM:	63	373	134	2032

- Automatisiertes Aufsetzen eines Servers mit ZFS, LUKS Encryption, NixOS, Services wie Nextcloud, Wiki.js, SSO, Monitoring, …
- Ein Shell Skript: `./bin/install.sh` aufgesetzt innerhalb von Minuten

NixOS: Caveats

- Dokumentation
  - Manchmal veraltet oder überhaupt nicht vorhanden
  - Source Code lesen ist meist einzige Option für Dokumentation
- Imperatives System oft schneller aufsetzbar (“quick and dirty”)
  - Über lange Zeit (und viele Installationen) gleicht sich das aus
- Es gibt nicht *jede* Software
  - Docker Container können Abhilfe schaffen
  - Insbesondere bei kommerzieller / proprietärer Software problematisch
- Annahmen über klassische Linux (z.B. Debian) Systeme gelten nicht
  - Es gibt kein /bin/bash und *nur* /usr/bin/env und /bin/sh
- Rebuilden von Configs ist langsam (vor allem auf alten Maschinen)
  - Ein Rebuild auf meinem Server dauert 1-2 min
- Fehlermeldungen sind teilweise sehr obskur
- Bei Upgrades *können* Dinge immernoch kaputt gehen

(Docker) Container

- Früher: Viele VMs mit z.B. Debian und einem einzigen Service
- Ziel: Gleiches Level an Isolierung beibehalten, aber performanter
  - Zusätzlicher Vorteil: Es braucht kein großes Cluster + Hypervisor
- Container teilen sich den Kernel, VMs nicht
- Ein Container ist eine isolierte Gruppe an Prozessen
  - Isolierung durch Namespaces, CGroups & privatem rootfs
- Für gewöhnlich über Environment-Variablen konfiguriert
- Definierte Volumes, welche in den Container gemounted werden
- Bekannt geworden durch Docker, besitzen aber standardisiertes Format
- Für gewöhnlich *sehr* einfach aufzusetzen

Docker: Security

- Docker-Container sind für gewöhnlich secure
  - Root Zugriff im Container sollte nicht zu Root auf Host führen
  - Sobald ein Container kompromittiert ist, kann dieser mit allen reden
- `dockerd` läuft mit root privileges, falls der exploited wird: problematisch
  - Gab schonmal privilege escalation CVE-2019-5736
  - Gibt aber auch inzwischen rootless Docker
  - Selbst arbitrary Commands können Problematisch sein: `/` unter `/host` im Container
- Docker Daemon Socket = root Zugriff

MailServer.nix × MailMan.nix

```
virtualisation.oci-containers.containers.mail = {
  image = "mailserver/docker-mailserver:latest";

  ports = [
    "25:25"
    "465:465"
    "587:587"
    "993:993"
    "4190:4190"
    "11334:11334"
  ];

  volumes =
  [
    "/data/Mail/mail-data:/var/mail"
    "/data/Mail/mail-state:/var/mail-state"
    "/data/Mail/mail-logs:/var/log/mail"
    "/data/Mail/config:/tmp/docker-mailserver"
    "/etc/localtime:/etc/localtime"

    # DNS
    "/etc/resolv.conf:/etc/resolv.conf:ro"

    # Overrides
    "/etc/postfix/postfix-main.cf:/tmp/docker-mailse
    "/etc/dovecot/dovecot.cf:/tmp/docker-mailserver/

    # MailMan
    "/data/MailMan/mailman-core/var/data/:/var/lib/m

    # Certificates
    "${config.age.secrets.Mail_EnvironmentFile.path}
    "/var/lib/acme/new-mail.inet.tu-berlin.de/:/var/
  ];
```

MailServer.nix ×

```
    "${config.age.secrets.Mail_EnvironmentFile.path}:${
    "/var/lib/acme/new-mail.inet.tu-berlin.de/:/var/lib
  ];

  environmentFiles = [ config.age.secrets.Mail_Environm

  environment = {
    OVERRIDE_HOSTNAME = "mail.${config.host.networking.
    LOG_LEVEL = "trace";
    TZ = "Europe/Berlin";
    ONE_DIR = "1";

    SSL_TYPE = "manual";
    SSL_CERT_PATH = "/var/lib/acme/new-mail.inet.tu-ber
    SSL_KEY_PATH = "/var/lib/acme/new-mail.inet.tu-berl

    ENABLE_QUOTAS = "0";
    ENABLE_MANAGESIEVE = "1";

    # Spam protection with Rspamd
    ENABLE_RSPAMD = "1";
    ENABLE_OPENDKIM = "0";
    ENABLE_OPENDMARC = "0";
    ENABLE_POLICYD_SPF = "0";

    # Virus protection
    ENABLE_AMAVIS = "0";  # Amavis is done by the DFN i
    ENABLE_CLAMAV = "0";

  };
};

environment.etc."postfix/postfix-main.cf".text = ''
  message_size_limit = 26214400000
  mailbox_size_limit = 26214400000
```

MailServer.nix ×

```
environment.etc."postfix/postfix-main.cf".text = ''
  message_size_limit = 26214400000
  mailbox_size_limit = 26214400000

  # MailMan Config
  recipient_delimiter = +
  unknown_local_recipient_reject_code = 550
  owner_request_special = no

  transport_maps = regexp:/var/lib/mailman/postfix_lmt
  local_recipient_maps = regexp:/var/lib/mailman/post
  relay_domains = regexp:/var/lib/mailman/postfix_doma

  mynetworks = 10.88.0.0/16
'';

environment.etc."dovecot/dovecot.cf".text = ''
  submission_max_mail_size = 262144000
  imap_max_line_length = 67108864
  mail_cache_max_size=200 MB
'';

systemd.tmpfiles.rules = [
  "d /data/Mail/mail-data/ 0755 5000 5000"
  "d /data/Mail/mail-state/ 0755 5000 5000"
  "d /data/Mail/mail-logs/ 0755 5000 5000"
  "d /data/Mail/config/ 0755 5000 5000"
];

}
```

Quellen / Literaturempfehlung

- UNIX and Linux System Administration Handbook (ISBN: 978-0134277554)
- The Practice of System and Network Administration (ISBN: 978-0321919168)