# Leseprobe

Zwei Auszüge als kostenfreie Vorschau: das Kapitel Basis-Härten und das Notfall-Szenario „Festplatte voll“ aus dem Notfall-Playbook. Das vollständige Handbuch (9 Kapitel mit Checklisten und Vorlagen, als PDF, Word und Markdown) erhalten Sie nach dem Kauf über Digistore24.

KAPITEL 2

# Basis-Härten: SSH, Firewall, Fail2ban

Ein Server, der aus dem Internet erreichbar ist, wird ab der ersten Minute automatisiert angegriffen — das ist kein Sonderfall, sondern Grundrauschen. Die gute Nachricht: Vier Maßnahmen wehren praktisch alle dieser Standardangriffe ab. Sie sind in 30 Minuten umgesetzt und gehören an den Anfang jedes Server-Lebens.

## SSH absichern: Schlüssel statt Passwort

Passwort-Logins sind das Haupteinfallstor. Stellen Sie auf SSH-Schlüssel um, bevor der Server produktiv wird:

1. Auf dem Arbeitsrechner ein Schlüsselpaar erzeugen: ssh-keygen -t ed25519 -C 'admin@‹servername›'
2. Öffentlichen Schlüssel auf den Server kopieren: ssh-copy-id ‹benutzer›@‹server-ip›
3. Login per Schlüssel testen — erst danach Passwort-Login abschalten.

Danach in /etc/ssh/sshd_config (oder als Drop-in unter /etc/ssh/sshd_config.d/99-haerten.conf) festziehen:

```
PermitRootLogin no
PasswordAuthentication no
KbdInteractiveAuthentication no
AllowUsers ‹benutzer›
```

Konfiguration prüfen und übernehmen: sshd -t && systemctl reload ssh

> **Wichtig:** Die bestehende SSH-Sitzung beim Umstellen geöffnet lassen und den Login parallel in einem zweiten Terminal testen. Wer sich aussperrt, braucht physischen Zugang oder die VPS-Konsole des Anbieters.

> **Hinweis:** Den SSH-Port zu verlegen (z. B. auf 2222) reduziert Log-Lärm, ist aber keine Sicherheitsmaßnahme — Portscanner finden ihn trotzdem. Schlüssel-Pflicht + Fail2ban wirken; der Port ist Geschmackssache.

## Firewall: Standardmäßig zu

Grundprinzip: Alles ist gesperrt, nur das Nötige ist offen. Unter Ubuntu ist ufw das einfachste Werkzeug:

```
ufw default deny incoming
ufw default allow outgoing
ufw allow OpenSSH
ufw allow 80/tcp    # HTTP  (Reverse Proxy, Kapitel 3)
```

```
ufw allow 443/tcp   # HTTPS (Reverse Proxy, Kapitel 3)
ufw enable
```

Mehr braucht ein typischer Home-Server nach außen nicht: Alle Web-Dienste laufen hinter dem Reverse Proxy über 443; interne Dienste bleiben im Container-Netz (Kapitel 4) oder werden per VPN erreicht.

> **Wichtig:** Docker umgeht ufw: Ein mit „ports: 8080:80“ veröffentlichter Container ist trotz Firewall aus dem Internet erreichbar, weil Docker eigene iptables-Regeln vor ufw setzt. Lösung in Kapitel 4: Ports nur an 127.0.0.1 binden und ausschließlich über den Reverse Proxy veröffentlichen.

## Fail2ban: Wiederholungstäter aussperren

Fail2ban liest Login-Fehlversuche aus den Logs und sperrt auffällige IP-Adressen automatisch für eine definierte Zeit. Installation und Minimal-Konfiguration (/etc/fail2ban/jail.local):

```
apt install fail2ban

# /etc/fail2ban/jail.local
[sshd]
enabled  = true
maxretry = 5
findtime = 10m
bantime  = 1h
```

Status prüfen: fail2ban-client status sshd — dort sehen Sie laufende Sperren und die Gesamtzahl der abgewehrten Versuche.

## Offene Ports inventarisieren

Was nicht läuft, kann nicht angegriffen werden. Verschaffen Sie sich einmal pro Monat den Überblick, was tatsächlich lauscht:

```
ss -tulpn          # alle lauschenden Ports + Prozesse
docker ps          # veröffentlichte Container-Ports
```

Tragen Sie das Ergebnis in das Dienste-Inventar ein (Vorlage im Anhang). Jeder Port ohne erklärbaren Zweck wird geschlossen oder der zugehörige Dienst deinstalliert.

| Port | Dienst | Erreichbar von | Zweck / Bemerkung |
|---|---|---|---|
| 22 | OpenSSH | Internet (nur Schlüssel) | Administration |
| 80 | Reverse Proxy | Internet | Nur Umleitung auf 443 |
| 443 | Reverse Proxy | Internet | Alle Web-Dienste, HTTPS |
| ‹…› | ‹…› | ‹nur intern / Internet› | ‹…› |

## Checkliste — direkt umsetzen

- ☐ SSH-Schlüssel eingerichtet, Passwort-Login und Root-Login deaktiviert
- ☐ ufw aktiv: deny incoming, nur SSH/80/443 offen
- ☐ Fail2ban installiert, sshd-Jail aktiv, Status geprüft
- ☐ Offene Ports mit ss -tulpn inventarisiert und im Anhang dokumentiert
- ☐ Aussperr-Test gemacht: Zweites Terminal, Login funktioniert noch

# Szenario A: Festplatte voll

Symptome: Dienste antworten mit Fehlern, Datenbanken verweigern Schreibzugriffe, im Log steht „no space left on device“. Häufigste Ursachen auf Home-Servern: Docker-Altlasten, ungebremste Logs, Backups auf derselben Platte.

1. Überblick: df -h — welche Partition ist voll?
2. Verursacher finden: du -xh --max-depth=2 / 2>/dev/null | sort -rh | head -20
3. Docker-Altlasten räumen (meist der größte Posten): docker system df, dann docker image prune -a und docker builder prune.
4. Journal eindampfen: journalctl --vacuum-size=200M
5. Container-Logs prüfen: Riesige *-json.log-Dateien unter /var/lib/docker/containers/ bedeuten fehlende Log-Rotation (Kapitel 5) — Rotation nachziehen, Container neu erstellen.
6. Danach Dienste prüfen: Insbesondere Datenbanken nach „Disk voll“ neu starten und Logs auf Korruptionshinweise prüfen.

> **Wichtig:** Niemals blind in /var/lib/docker löschen und keine Dateien entfernen, deren Zweck unklar ist. Wenn eine Datenbank betroffen war: Erst Dump ziehen (solange sie läuft), dann weiter aufräumen.

Namaru — Wissens-Archiv + async Consulting für deutsche KMU. Vollständiges Handbuch: https://namaru.de/produkte/home-server-betriebshandbuch/