

Todo App

Eine Lern-Anwendung für Softwarearchitektur

Vom Quick & Dirty zur Clean Architecture

Robert Bretz

6. Mai 2026

Inhaltsverzeichnis

1	Server-Absicherung (Ubuntu 24.04 auf Contabo VPS)	3
1.1	Schritt 1: SSH-Verbindung testen	3
1.2	Schritt 2: System-Updates	4
1.3	Schritt 3: SSH-Key-Authentifizierung	4
1.4	Schritt 4: SSH-Client-Konfiguration (Alias)	5
1.5	Schritt 5: SSH-Timeout auf 20 Minuten	6
1.6	Schritt 6: Fail2Ban (Bruteforce-Schutz)	6
1.7	Zusammenfassung	7

Das passiert, weil der Server einen neuen SSH-Fingerabdruck hat (durch die Neuinstallation). Dein PC erinnert sich an den alten Fingerabdruck und warnt dich vor einem möglichen Man-in-the-Middle-Angriff. Da du den Server selbst neu installiert hast, ist das harmlos.

Lösung: Den alten Eintrag löschen mit:

```
1 ssh-keygen -f '/home/computer/.ssh/known_hosts' -R '185.209.229.167'
```

Listing 3: Alten SSH-Fingerabdruck entfernen

Danach erneut verbinden und den neuen Fingerabdruck mit yes bestätigen.

1.2 Schritt 2: System-Updates

Nach dem ersten Login wird das System auf den neuesten Stand gebracht.

Ausgeführt auf dem Server (root@vmd147914):

```
1 apt update && apt upgrade -y
```

Listing 4: System-Updates ausführen

Erklärung des Befehls:

- apt – Advanced Package Tool, der Paketmanager von Ubuntu/Debian
- update – holt die neuesten Paketlisten von den Ubuntu-Servern
- && – führt den zweiten Befehl nur aus, wenn der erste erfolgreich war
- upgrade – installiert alle verfügbaren Aktualisierungen
- -y – beantwortet alle Rückfragen automatisch mit "Yes"

1.3 Schritt 3: SSH-Key-Authentifizierung

Ein SSH-Key ist sicherer als ein Passwort, da er nicht durch Ausprobieren (Bruteforce) erraten werden kann. Er besteht aus zwei Teilen:

- **Private Key** (id_ed25519) – bleibt auf deinem PC, niemals weitergeben!
- **Public Key** (id_ed25519.pub) – wird auf den Server kopiert

Das Verfahren nennt sich **asymmetrische Kryptographie**: Der Server schickt eine zufällige Nachricht, dein PC unterschreibt sie mit dem privaten Schlüssel, der Server prüft die Unterschrift mit dem öffentlichen Schlüssel. Stimmt sie überein, bist du eingeloggt – ohne Passwort.

Schritt 3a: Key-Paar erstellen – auf deinem lokalen PC:

```
1 ssh-keygen -t ed25519 -C "robert@local"
```

Listing 5: SSH-Key generieren

Erklärung des Befehls:

- ssh-keygen – Programm zum Erstellen von SSH-Schlüsselpaaren

- `-t ed25519` – verwendet den modernen Ed25519-Algorithmus (kurz, schnell, sicher)
- `-C "robert@local"` – Kommentar, damit du später erkennst, wofür der Key ist

Bei den Rückfragen einfach Enter drücken – der Key wird im Standardverzeichnis `~/.ssh/` gespeichert.

Schritt 3b: Public Key auf den Server kopieren – auf deinem lokalen PC:

```
1 ssh-copy-id root@185.209.229.167
```

Listing 6: Public Key auf den Server übertragen

Einmal das Server-Passwort eingeben. Der Befehl kopiert deinen Public Key in die Datei `~/.ssh/authorized_keys` auf dem Server.

Schritt 3c: Testen – auf deinem lokalen PC:

```
1 ssh root@185.209.229.167
```

Listing 7: Login ohne Passwort testen

Du wirst jetzt ohne Passwort-Abfrage eingeloggt.

1.4 Schritt 4: SSH-Client-Konfiguration (Alias)

Damit du nicht jedes Mal die IP-Adresse eintippen musst, wird ein Alias in der lokalen SSH-Konfiguration eingerichtet.

Auf deinem lokalen PC:

```
1 nano ~/.ssh/config
```

Listing 8: SSH-Konfiguration bearbeiten

Folgenden Inhalt einfügen:

```
1 Host testserver
2     HostName 185.209.229.167
3     User root
4     IdentityFile ~/.ssh/id_ed25519
```

Listing 9: Inhalt von `~/.ssh/config`

Erklärung der Konfiguration:

- `Host testserver` – der Alias, unter dem du den Server ansprichst
- `HostName 185.209.229.167` – die tatsächliche Server-Adresse
- `User root` – Benutzername für die Verbindung
- `IdentityFile ~/.ssh/id_ed25519` – Pfad zum privaten Schlüssel

Testen:

```
1 ssh testserver
```

Listing 10: Mit Alias verbinden

Ab jetzt reicht dieser kurze Befehl.

1.5 Schritt 5: SSH-Timeout auf 20 Minuten

Standardmäßig trennt Ubuntu inaktive SSH-Verbindungen nach etwa 5 Minuten. Das wird nun auf 20 Minuten erhöht.

Auf dem Server (als root):

```
1 nano /etc/ssh/sshd_config
```

Listing 11: SSH-Server-Konfiguration bearbeiten

Folgende Zeilen suchen oder am Ende der Datei einfügen:

```
1 ClientAliveInterval 120
2 ClientAliveCountMax 10
```

Listing 12: Timeout-Konfiguration

Erklärung der Werte:

- `ClientAliveInterval 120` – Der Server sendet alle 120 Sekunden (2 Minuten) ein Signal an den Client
- `ClientAliveCountMax 10` – Nach 10 unbeantworteten Signalen wird die Verbindung getrennt

Die gesamte Timeout-Zeit berechnet sich: $120 \text{ Sekunden} \times 10 = 1200 \text{ Sekunden} = 20 \text{ Minuten}$. **SSH-Dienst neustarten:**

```
1 systemctl restart ssh
```

Listing 13: SSH-Dienst neustarten

Wichtig: Auf Ubuntu heißt der Dienst `ssh`, nicht `sshd` (im Gegensatz zu anderen Distributionen). Die aktuelle Verbindung bleibt beim Neustart bestehen. Die neue Einstellung gilt für alle zukünftigen Verbindungen.

1.6 Schritt 6: Fail2Ban (Bruteforce-Schutz)

Fail2Ban ist ein Dienst, der Logdateien überwacht und IP-Adressen automatisch sperrt, wenn zu viele fehlgeschlagene Login-Versuche erkannt werden.

Was ist Bruteforce? Ein Angreifer probiert tausende Passwörter durch, bis er das richtige findet. Fail2Ban unterbindet das, indem es die IP des Angreifers nach einer bestimmten Anzahl Fehlversuche temporär sperrt.

Standard-Konfiguration (ab Werk):

- 5 Fehlversuche in 10 Minuten
- Sperrdauer: 10 Minuten
- Überwacht wird der SSH-Dienst

Wo wird installiert? Die Programmdateien liegen unter `/usr/bin/`, die Konfiguration unter `/etc/fail2ban/`.

Wo kann ich es konfigurieren? Die Datei `/etc/fail2ban/jail.local` wird bei Updates nicht überschrieben und ist für eigene Anpassungen gedacht. Beispiel:

```
1 bantime = 600
2 findtime = 600
3 maxretry = 3
4
5 [sshd]
6 enabled = true
```

Listing 14: Beispiel: /etc/fail2ban/jail.local

Installation auf dem Server:

```
1 apt install -y fail2ban
```

Listing 15: Fail2Ban installieren

Automatischen Start aktivieren und sofort starten:

```
1 systemctl enable fail2ban && systemctl start fail2ban
```

Listing 16: Fail2Ban aktivieren und starten

Status prüfen:

```
1 systemctl status fail2ban
```

Listing 17: Fail2Ban-Status abfragen

Die Ausgabe sollte active (running) zeigen.

```
1 - fail2ban.service - Fail2Ban Service
2   Active: active (running)
3   ...
4   Server ready
```

Listing 18: Erfolgreiche Ausgabe

1.7 Zusammenfassung

Nach Abschluss dieses Schritts ist der Server grundlegend abgesichert:

- Passwort-Login funktioniert weiterhin (als Backup)
- SSH-Key-Login ist eingerichtet (bequem & sicher)
- Alias `ssh testserver` ist konfiguriert
- Verbindung trennt nach 20 Minuten Inaktivität
- Fail2Ban sperrt Angreifer nach 5 Fehlversuchen

Als nächstes folgt die Firewall-Konfiguration mit `ufw`.