SSH

Banner-Warnseite anlegen

Will man eine Warnseite bei einem SSH Zugang anzeigen, hilft diese Anleitung

Remote Script über SSH ausführen

Installieresshpass, ein Tool zur nicht-interaktiven ssh-Passwort-Authentifizierung.
So führst du den Befehl sudo ohne Passwort aus:

SSH hat deinen Job abgebrochen und beendet? Hier ist die Lösung

Du hast einen langwierigen rsync-Job oder ein Python-Skript auf einem Remote-Server laufen lassen, nur um mit anzusehen, wie es in dem Moment abstürzt, in dem deine SSH-Sitzung unterbrochen wird, und jetzt musst du dich mit nohup, screen, tmux und systemd auseinandersetzen, damit das nie wieder passiert.

Nähere Infos findest du auf Tecmint

SSH-Absicherung: fail2ban mit ipset-Integration

Ausgangslage

Massive Brute-Force-/Scan-Angriffe auf den SSH-Dienst, u. a. von einzelnen IPs mit mehreren zehntausend Login-Versuchen. SSH ist bereits über Key-Auth, Non-Standard-Port und fail2ban abgesichert. Ziel dieser Maßnahme: Reduktion von Logspam, Ressourcenverbrauch und Angriffsfläche durch effizientere Blockierung auf Firewall-Ebene mittels **ipset** statt einzelner iptables-Regeln pro IP.

Voraussetzungen

* Root- bzw. sudo-Zugriff auf den Server
* fail2ban bereits installiert und aktiv
* iptables (legacy) im Einsatz

1. ipset installieren

```
\\
sudo apt update\\
sudo apt install ipset\\
ipset --version\\
```

2. Manuelle Blocklist für bekannte Angreifer-IPs

Für IPs/Ranges, die dauerhaft und unabhängig von fail2ban gesperrt werden sollen (z. B. bekannte bösartige Hoster-Ranges):

```
sudo ipset create blocklist hash:\ip
sudo ipset add blocklist 61.138.113.\187
sudo iptables -I INPUT -m set --match-set blocklist src -j \DROP
```

3. Persistenz der manuellen Blocklist

ipset-Sets sind standardmäßig **nicht** über Reboots persistent.

3.1 Einmalig sichern

```
sudo ipset save > /etc/ipset.\conf
```

Kontrolle:

```
cat /etc/ipset.\conf
```

3.2 Restore beim Boot (systemd)

Datei: `/etc/systemd/system/ipset-persistent.service`

```
[Unit]
Description=Restore ipset \rules
Before=netfilter-persistent.\service
After=network-pre.target

[Service]
Type=\oneshot
ExecStart=/sbin/ipset restore -! -file /etc/ipset.conf

[Install]
WantedBy=multi-user.\target
```

```
sudo systemctl enable ipset-\persistent
```

3.3 Automatisches Sichern bei Änderungen

Da neue IPs nicht automatisch in `/etc/ipset.conf` landen, empfiehlt sich ein Save-Service beim Shutdown.

Datei: `/etc/systemd/system/ipset-save.service`

```
[Unit]
Description=Save ipset rules on \shutdown
Before=shutdown.target reboot.target halt.target

[Service]
Type=\oneshot
ExecStart=/sbin/ipset save -file /etc/ipset.\conf
RemainAfterExit=yes

[Install]
WantedBy=halt.target reboot.target shutdown.\target
```

```
sudo systemctl enable ipset-save.\service
```

> **Hinweis:** Dieser Save/Restore-Mechanismus betrifft ausschließlich das manuell gepflegte Set `blocklist`. Das fail2ban-eigene Set (siehe Punkt 4) wird davon bewusst **nicht** erfasst.

4. fail2ban-Integration mit ipset

Anstatt dass fail2ban für jede gebannte IP eine eigene iptables-Regel anlegt, wird die mitgelieferte ipset-Action genutzt. Diese erstellt und verwaltet das Set (`f2b-<jailname>`) sowie die zugehörige iptables-Regel **automatisch** beim Start/Stop des Jails.

4.1 Verfügbare Actions prüfen

```
ls /etc/fail2ban/action.d/ | grep \ipset
```

Relevant: `iptables-ipset-proto6.conf`

4.2 jail.local konfigurieren

Datei: `/etc/fail2ban/jail.local`

```
[sshd]
enabled = \true
port = <NICHT-STANDARD-PORT>
```

```
maxretry = \3
findtime = \600
bantime = \86400
bantime.increment = \true
bantime.factor = \24
bantime.maxtime = \604800
banaction = iptables-ipset-\proto6
```

* **bantime.increment**: Bannzeit verlängert sich bei wiederholten Verstößen exponentiell (Faktor 24, Cap bei 7 Tagen) * **banaction**: nutzt ipset statt Einzelregeln

4.3 fail2ban neu starten

```
sudo systemctl restart \fail2ban
sudo fail2ban-client status \sshd
```

4.4 Kontrolle: Set und Regel automatisch angelegt

```
sudo ipset list f2b-\sshd
sudo iptables -L INPUT -n --line-numbers | grep f2b-\sshd
```

4.5 Funktionstest

```
sudo fail2ban-client set sshd banip 1.2.3.\4
sudo ipset list f2b-\sshd
sudo fail2ban-client set sshd unbanip 1.2.3.\4
```

5. Ergänzende Härtungsmaßnahmen

5.1 sshd_config

```
PermitRootLogin \no
AllowUsers <username>
MaxStartups 3:50:\10
LoginGraceTime \15
```

5.2 Connection-Rate-Limiting (zusätzlich zu fail2ban)

```
sudo iptables -A INPUT -p tcp --dport <PORT> -m conntrack --ctstate NEW -m
recent --set --name \SSH
sudo iptables -A INPUT -p tcp --dport <PORT> -m conntrack --ctstate NEW -m
recent --update --seconds 60 --hitcount 4 --name SSH -j \DROP
```

Begrenzt neue Verbindungsversuche auf 4 pro Minute je IP, unabhängig von fail2ban.

5.3 Optionale Erweiterungen (nicht umgesetzt / zu prüfen)

* GeoIP-Blocking (xtables-addons / nftables-Sets) für Länder ohne legitimen \Zugriffsbedarf * CrowdSec als Ergänzung/Ersatz zu fail2ban (Community-Blocklists) * Automatisches Reporting gebannter IPs an AbuseIPDB via fail2ban-Plugin

6. Architekturübersicht

Set	Verwaltet durch	Persistenz	Zweck
blocklist	manuell / Admin	ipset-persistent.service (Boot)	dauerhafte Sperre bekannter Angreifer
f2b-sshd	fail2ban (automatisch)	keine (Neuaufbau bei Jail-Start)	dynamische Bans aus Login-Versuchen

Offene Punkte / To-Do

* Migration bereits bestehender, einzeln gesetzter iptables-Bans in das neue ipset-Schema
* Entscheidung zu GeoIP-Blocking treffen
* Monitoring/Alerting bei ungewöhnlich hoher Ban-Rate einrichten

Changelog

Datum	Änderung	Autor
date	Ersterstellung – ipset + fail2ban-Integration	Franz

From:
http://wiki.waldhofer.at/ - **Wiki von Franz**

Permanent link:
http://wiki.waldhofer.at/doku.php?id=ubuntu:ssh

Last update: **2026/09/11 09:06**