Aufgabenblatt 00 - ssh Security und fail2ban (v1.1)

Petar Cubela

Thu Sep 18 2025

Aufgabenblatt 00

In diesem Blatt lernst du wichtige Grundlagen, um die Sicherheit eines Linux Servers zu gewaehrleisten.

`ssh` (*secure shell*) ist eines der wichtigsten Tools die ein Linux Administrator beherrschen sollte. Dieses erlaubt es uns, eine Shell zu einem remote Linux (im allgemeinen Unix) System zu oeffnen, um dann administrative Taetigkeiten am Server durch zu fuehren. Die Authentifizierung passiert hier in der Regel mit Username und Passwort. Es gibt aber Situationen in denen es nicht ratsam ist, Usernamen und Passwort fuer die Authentifizierung zu nutzen. Zum Besipiel, wenn der `ssh` Port, 22, oeffentlich zuegaenglich ist, und damit jeder den *OpenSSH* Server des Linux Servers erreichen koennte. Ein Passwort koennte durch eine brute-force Attacke erraten werden oder durch andere Wege komprimittiert werden. Aus dem Grunde gibt es `ssh` Key Paare, welche es uns ermoeglichen uns ohne Passwort am Server zu authentifizieren.

Einer der keys ist hier ein **privater** Schluessel, den niemand haben sollte und der immer sicher aufbewahrt werden sollte. Der zweite key ist **public** und darf somit von anderen gesehen/gelesen werden. Der public Key wird am Linux Server in der Datei `/home/$USER/.ssh/authorized_keys` abgelegt, waehrend der private Key am eigenen Rechner hinterlegt wird, ebenfalls im Ordner `/home/$USER/.ssh/`. Beim Versuch sich mittels `ssh` am Ziel Server anzumelden, gleicht der Server seinen public Key mit dem private Key des `ssh`-Clients ab und wenn die beiden zu einander passen, wird Zugang gewaehrt.

Wir lernen, wie wir ssh key Paare erzeugen und am Ziel Server zu hinterlegen, welche es uns erlauben, uns ohne Passwort an einem OpenSSH server zu authentifizieren. Dies wird in Zukunft auch sehr nuetzlich sein, um Automatisierungstools gegen einen oder *mehrere* Server laufen zu lassen.

Zusaetzlich lernen wir `fail2ban`, welches ein einfaches *intrusion detection system(IPS)* ist, welches in der Lage ist Ports zu Diensten auf dem Server zu ueberwachen und bei missbraeulichem Verhalten IP Addressen zu blocken. Zum Beispiel kann eine IP Adresse fuer eine Stunde geblockt werden, wenn der entsprechende `ssh`-Client drei mal das Passwort falsch eingibt.

Um Datein zu bearbeiten benutze einen beliebigen Text Editor, wie zum Beispiel `nano` oder `vim`. Diese sind in der Regel auf den meisten Linux Betriebssystemen vorinstalliert.

Aufgabe 1 - ssh security and config

a. Erstelle ein `ssh` Key Paar auf deinem lokalen Rechner, lege den public key auf deinem remote Linux Server ab und melde dich ohne Passwort mit `ssh` am Server an. `ssh USER@EXAMPLE.COM`.

Tools, welche gentuzt werden sollen:

- `ssh`
- `ssh-keygen` (wenn dieser Befehl benutzt wird, setze keien Passwort fuer das Key Paar, sonst musst ihr Dieses jedes mal eingeben, wenn die Keys genutzt werden)
- `ssh-copy-id`

b. Passe die Konfiguration des *ssh daemons* `sshd`, unter dem Pfad `/etc/ssh/sshd_config`, so an, dass

- eine `root` Anmeldung nicht moeglich ist. (Der `root` User hat uneingeschraenkte Berechtigungen auf dem Server. Dieser sollte ueber `ssh` nie erreichbar sein.)
- die Passwort Authentifizierung nicht moeglich ist,

- und deiner Meinung nach die Sicherheit des `ssh`-Zugangs weiter erhoeht wird.

Nachdem der `ssh`-Dienst angepasst wurde, muss Dieser neugestartet werden, damit die Aenderugen geladen werden:

```
$ sudo systemctl restart ssh.service
$ sudo systemctl status ssh.service
```

Aufgabe 2 - fail2ban to protect ssh

Installiere `fail2ban` und konfiguriere es so, dass

- `ssh` ein aktives `jail` ist
- bei 3 maliger Passwortfalscheingabe, die IP Adresse des Clients fuer 2 Stunden gebannt wird

Die Konfigurationsdateien von `fail2ban` sind im Pfad `/etc/fail2ban/` hinterlegt.

Anleitung:

Gehe wie folgt vor:

1. Installation:

```
sudo apt install fail2ban
```

2. Pruefe den Status des `fail2ban`-Dienstes mittels `systemd`:

```
$ systemctl status fail2ban.service
```

3. Lese den Inhalt einer default Konfigurations:

```
$ cd /etc/fail2ban/
$ head -20 jail.conf
```

4. Erstelle wie empfohlen eine *local* jail Konfigurationsdatei:

```
$ cp jail.conf jail.local
```

5. Passe die `jail.local` Datei an, so dass die Vorgabe umegestzt ist.

```
$ sudo nano/vim jail.local
```

Recherchiere gegebenfalls im Internet, um dies umzusetzen. (Tipp: `ssh` speichert seine logs nicht mehr standard-maessig im Pfad `/var/log/`, sondern nutzt `systemd-journald`)

6. Teste deine Konfig von einem auf dem `pve.lab.softbox.net` verfuegbaren Linux Host.

Aufgabe 3 - nginx web server

Wir installieren hier einen der am meisten genutztne Web server auf der Welt `nginx` und machen damit den ersten Schritt zu einer web page.

Installiere `nginx`:

```
$ sudo apt install nginx
```

Pruefe den `nginx`-Dienst via `systemctl`:

```
$ sudo systemctl status nginx
```

Besuche die Seite http://student1.lab.softbox.net, welche dir die `nginx` welcome page zeigen sollte. Die Konfigurationsdateien von `nginx` unter dem Pfad `/etc/nginx/` zu finden. Die `html` (+ `css` + etc.) sind in der Regel unter dem Pfad `/var/www/` zu finden, wobei Dieser in `nginx`-Konfigurationsdateien beliebig waehlbar ist.

Pruefe auf welcher IP Adresse und welchem Port `nginx` *lauscht* (im englischen: *listen*

```
$ sudo ss -tulpn | grep nginx
```

Zusatz: Nutze `man ss`, um dich mit dem Befehl `ss` und den gewaehlten Optionen auseinanderzusetzen. Dies ist ein sehr praktischer Befehl, der sehr haeufig genutzt wird.