Absicherung eines Raspberry Pi 3 mittels Hardening Techniken

A W M

an der Hochschule für Angewandte Wissenschaften Hof
Fakultät Informatik
Studiengang Mobile Computing
Modul AWM: Projektarbeit
Wintersemester 23/24

Sebastian Peschke (00706319)

Hof, 12. März 2024

Zusammenfassung

Die Studienarbeit im Rahmen des Wahlmoduls einer Projektarbeit bei einem Professor behandelt das Absichern eines Raspberry Pi 3 mithilfe von Linux Hardening-Techniken. Dabei wird auf allgemeine Konfigurationen nach der Installation des Betriebssystems eingegangen. Das Hardening von Geräten mit Internetzugang ist, aufgrund der Übertragung kritischer Daten, ein wichtiger Bestandteil von Infrastrukturen.

Inhaltsverzeichnis

Abkürzungsverzeichnis

API Application Programming Interface 1

BASH Bourne-again Shell 1

BSPW beispielsweise 5

CLI Command Line Interface 1

HTTP Hypertext Transfer Protocol 5

HTTPS Hypertext Transfer Protocol Secure 5

IoT Internet of Things 1

IP Internet Protocol 3, 6

IT Information Technology 4

LAN Local Area Network 3

OS Operating System 1, 2

SSH Secure Shell Protocol 1, 2, 3, 4, 5, 6, 7

SV Studierendenvertretung 1, 4

TCP Transmission Control Protocol 5

UFW Uncomplicated Firewall 5

Listings

1 Einführung

IoT (Internet of Things)Geräte finden in der heutigen Zeit immer mehr Verwendung im Alltag der Menschheit. Sei es die Smart-Fridge, welche einem ihre aktuelle Innentemperatur per App mitteilt oder die neue Philips hue smart light bulb, welche sich dem Verbrauchsverhalten des Nutzers anpasst bis hin zu einem Automatisierungssensor zum Kalkulieren der Auslastung eines Laufbandes in einem großen Unternehmen. Auch die SV (Studierendenvertretung) der Hochschule Hof hat sich diese Möglichkeit nicht entgehen lassen sich den Alltag eines Events einfacher zu gestalten. Zum komfortableren Bewältigen der Events, wie die BOOM-Party oder diverse Semester-Opening-Feiern gehört eine gute Koordination untereinander. Diese Koordination wurde durch die Organisationseinheit Technik der SV mittels einer Webapplikation, welche einen Echtzeit-Chat mithilfe der Telegram API (Application Programming Interface) überträgt, auf einem Raspberry Pi 3 umgesetzt. Das IoT-Gerät ist mit einem Raspbian OS (Operating System) eingerichtet und besitzt einen SSH (Secure Shell Protocol)-Server, welcher es Entwicklern und Maintainern der Web-App ermöglicht diese weiterzuentwickeln und zu warten. Um eine genügende Sicherheit des Geräts zu gewährleisten und sensible Daten der Hochschule und auch der SV nicht in Gefahr zu bringen, soll der Raspberry Pi 3 mit seinen installierten und verwendeten Technologien abgesichert werden.
Die folgenden Kapitel beschäftigen sich mit der Problematik ein IoT-Gerät ordnungsgemäß und angemessen abzusichern. Zuerst wird die initiale Konfiguration genauer angesehen. Sie beinhaltet das Konfigurieren eines Benutzers mit wenig Privilegien, welcher zum Login von außerhalb über SSH verfügbar sein soll. Ebenfalls beschäftigt sich das Kapitel mit dem Deaktivieren des default Nutzers. Als Nächstes wird das Einrichten von automatischen Sicherheitsupdates veranschaulicht. Darauf folgt die Konfiguration einer klein gehaltenen Firewall, welche es ermöglicht unerwarteten Netzwerkverkehr anhand des Whitelisting-Prinzips zu blockieren.
Der nächste Punkt geht auf bekannte und oft ausgenutzte Schwachstellen ein und wie man diese mittels weiterer Techniken mitegieren kann.
Die in diesem Dokument gezeigten Konfigurationen werden ausschließlich über das BASH (Bourne-again Shell)-CLI (Command Line Interface) veranschaulicht.

Der grundlegenden Herangehensweise der Konfiguration wurden aus großen Teilen eines Blogs [1] entnommen.

2 Initiale Konfiguration

Dieses Kapitel beschäftigt sich mit der Konfiguration eines frisch installierten Systems. Die Installation des Raspbian OS wird dazu in dieser Arbeit nicht beschrieben.

2.1 Benutzerverwaltung

Zur initialen Konfiguration gehört das Verändern der default Nutzer, um den Einfallswinkel für einfache Angriffe zu minimieren. Um diesem Fehler vorzubeugen wird zunächst ein neuer Nutzer angelegt. Der Nutzer in diesem Beispiel trägt den Namen *svuser*.

Listing 1: bash Befehl zum erzeugen eines neuen Benutzers

```
$ sudo adduser svuser
```

Nun wird ein zweiter Nutzer *svadmin* angelegt. Dieser ist gerade für die Wartung und Weiterentwicklung des Raspberry Pis wichtig, um weitere Änderungen am Betriebssystem vornehmen zu können. Im Folgenden werden alle mit `sudo` ausgeführten Befehle über diesen Benutzer durchgeführt. Um dem Nutzer die entsprechenden Rechte zu geben, muss er zuerst zur `sudoers` gruppe hinzugefügt werden.

```
$ sudo usermod -aG sudo svadmin
```

Mit dem neu erzeugten Nutzer sollen weitere Änderungen vorgenommen werden. Dazu gehört das Abschalten des default Nutzers *pi* mit hohen Privilegien.

Listing 2: Abschalten des default Nutzers

```
$ sudo usermod --lock --expiredate 1 pi
```

Im Anschluss ist es eine gute Idee den `root` Benutzer zu konfigurieren. Hierzu wird der SSH-Zugriff von diesem Benutzer abgeschaltet. Um das zu bewerkstelligen, wird eine Benutzergruppe `ssh-users` angelegt und der aktuell angemeldete Benutzer dieser Gruppe hinzugefügt. Zum Inkrafttreten des Verbots, sich mit dem `root` Nutzer anmelden zu können, müssen folgende Änderungen in der SSH-Konfigurationsdatei in `/etc/ssh/sshd_config` vorgenommen werden:

- PermitRootLogin no
- AllowGroups ssh-users

Listing 3: Ändern des Passworts

```
$ sudo groupadd ssh-users
$ sudo usermod -a -G ssh-users svuser
```

Darauf folgt für das Inkrafttreten der Änderungen den SSH-Prozess neu zu starten.

Listing 4: Neustart des ssh-Prozesses

```
$ sudo systemctl restart ssh
```

2.2 Abschalten der Wireless Interfaces

Ein weiterer präemptiver Schritt zum Vorbeugen von weitverbreiteten Angriffsvektoren besteht daraus das *Wireless Interface* des Raspberry Pis zu deaktivieren. Diese Entscheidung beruht auf der Tatsache, dass der Pi3 mit einer statischen IP (Internet Protocol)-Adresse versehen wurde und nur über LAN (Local Area Network) erreichbar sein soll, um die Überwachung des Netzwerktraffic über den IT-Service einfacher zu gestalten.

Listing 5: Deaktivieren der Wireless Interfaces

```
$ sudo vim /boot/config.txt
....
# Disable wireless services
dtoverlay=pi3-disable-wifi
dtoverlay=pi3-disable-bt
```

Hierzu wurde auch das Bluetooth Interface deaktiviert, da es derzeit dieser Funktionalität nicht bedarf. Nach dem Anfügen der Konfigurationsdaten in der `config.txt` folgt ein Neustart des Systems, um überprüfen zu können, ob die Veränderungen übernommen wurden.

2.3 optional: SSH Konfiguration

Zum weiteren Absichern des SSH-Zugangs wäre es möglich dem erstellten Nutzer unter `/home/svuser/.ssh/authorized_keys` einen Public-Key abzulegen. Dieser könnte dann zum passwortlosen Einloggen mittels des SSH-Keys verwendet werden.

Um keine Redundante Einloggmöglichkeiten zu ermöglichen, werden in der in 2.1 beschriebenen Konfigurationsdatei die Zeilen

- PubkeyAuthentication yes
- PasswordAuthentication no

hinzugefügt. Die Änderung tritt jedoch erst in Kraft, wenn der im Hintergrund laufende SSH Prozess neu gestartet 4 wird. Von dieser Konfiguration wurde in diesem Projekt aufgrund Inkonsistenz der Besetzung der IT (Information Technology)-Abteilung der SV abgesehen.

3 Automatische Security Updates

Aufgrund der sperrligen Besetzung der Technikabteilung der SV, sowie der dauerhaften Aufgabenstellung alle IT-Systeme instand zu halten ist es ratsam automatische Sicherheitsupdates auf Geräten zu konfigurieren. Für solche Zwecke gibt es bereits vorgefertigte Programme, die einem Administrator dabei helfen.
Die in diesem Projekt verwendeten Technologien beschränken sich hierbei auf die Folgenden:

- unattended-upgrades
- apt-listchanges
- apticron

Listing 6: Installation von Automatischen updates

```
$ sudo apt install unattended-upgrades apt-listchanges
  ↪ apticron
```

Zudem wurde sich bei der Konfiguration der automatischen Updates mithilfe einer bereits angefertigten Konfigurationsdatei [2] beholfen. Diese Konfiguration muss zunächst in das Verzeichnis `/etc/apt/apt.conf.d/` kopiert werden. Um zu verifizieren, ob die automatischen Updates funktionieren sollte der folgende Command eingegeben werden:

Listing 7: Verifikation der Konfiguration von automatischen Updates

```
$ sudo unattended-upgrade -d --dry-run
```

Zum Nachverfolgen der automatisierten Updates werden die Updates in `/car/log/unattended-upg` geloggt.

4 Simple Firewall Konfiguration mit UFW

Auf einem Raspberry Pi3 ist keine Firewall vorinstalliert. Somit muss zuerst eine Firewall installiert werden. Der simplizität und der lesbaren Dokumentation geschuldet wurde sich hierbei für die UFW (Uncomplicated Firewall) entschieden.

Listing 8: Installation der Firewall

```
$ sudo apt install ufw
```

Die Firewall besitzt eine Liste von Ad-Hoc-Kommandos [3], deren sich bedient wurde.

Listing 9: Konfiguration der UFW mit ad-hoc Kommandos

```
$ sudo ufw default deny incoming
$ sudo ufw default allow outgoing
$ sudo ufw allow http
$ sudo ufw allow https
$ sudo ufw allow ssh
$ sudo ufw logging on
$ sudo ufw enable
```

Die hier verwendeten Kommandos Konfigurieren die Firewall nach dem White-Listing Prinzip. Zuerst werden mit `ufw default deny incoming` alle Verbindungen, welche von außen auf den Raspberry zugreifen wollen, blockiert. Danach werden mit `ufw default allow outgoing` alle Verbindungen, die der Raspberry Pi nach außen, BSPW (beispielsweise) auf das Internet mit dem Kommando `ping` oder einem Browser, erlaubt. Die weiteren Konfigurationen mit dem Schlüsselwort `allow` ermöglichen es Verbindungen, die von außen auf den Raspberry Pi zukommen, zu erlauben. Die erlaubten Verbindungen sind jedoch nur auf HTTP (Hypertext Transfer Protocol) (TCP (Transmission Control Protocol) Port 80), HTTPS (Hypertext Transfer Protocol Secure) (TCP Port 443) und SSH (TCP Port 22) beschränkt.
Das Kommando `ufw logging on` bezweckt, dass die von der Firewall blockierten Verbindungen unter `/var/log/ufw.log` aufgezeichnet werden.
Zum schlussendlichen Aktivieren und Anwenden der Regeln der Firewall wird diese jetzt mit dem `enable` Schlüsselwort aktiviert.

5 Verbreitete Schwachstellen und ihre Mitigationen

Zur weiteren Verstärkung der Credentials für den SSH-Nutzer `svuser` und, um einer Brute-Force-Attacke vorzubeugen [4], wird in dieser Sektion die Installation und Einrichtung von fail2ban beschrieben. Fail2ban [5] ist eine Open-Source-Software, welche Logdateien scant und anhand der darin enthaltenen IP-Adresse diese für einen konfigurierbaren Zeitraum bannt.
Installiert werden kann diese Software entweder über das offizielle Github Repository oder für auf Debian basierten Systemen (oder ähnliche) über den im Betriebssystem verfügbaren Paket-Manager.

Listing 10: Installation von fail2ban

```
$ sudo apt install fail2ban
```

Die Installation beinhaltet das Einrichten eines Hintergrundservice, welcher jedoch initial nicht aktiv ist. In der bereits vorkonfigurierten Datei `/etc/fail2ban/jail.conf` existieren Konfigurationen, wie eine Maximalanzahl für Versuche zum Login über SSH oder Einstellungen für zahlreiche andere Webframeworks. Für bessere Wartbarkeit der Software wird hier eine `jail.local` Datei angelegt, da bei jedem Update von fail2ban die Konfigurationsdatei automatisch mit einer neueren Version überschrieben wird.

Listing 11: Anlegen einer lokalen jail.conf Datei

```
$ sudo cp /etc/fail2ban/jail.conf /etc/fail2ban/jail.
  ↪ local
```

In der Konfigurationsdatei gibt es einen Eintrag mit dem Namen `bantime`. Für viele Brute-Force-Attacken spielt Zeit eine große Rolle. Um einem Angreifer in diesem Szenario *"den Spaß zu verderben"* wird mit dem Setzen der bantime, nach mehrmaligem Fehlversuch, auf 10 Minuten mit `bantime=10m` bereits viel geholfen. Weitere Konfigurationen können in folgender Form (siehe Listing 12) in der **jail.local** Datei vorgenommen werden.

Listing 12: Konfiguration des ssh jails von fail2ban

```
$ sudo nano /etc/fail2ban/jail.local
   [sshd]
   enabled = true
   port = ssh
   filter = sshd
   logpath = /var/log/auth.log
   maxretry = 3
   bantime = 10m
```

Mithilfe dieser Konfiguration ist es nun nicht mehr möglich einen Brute-Force-Angriff auf den SSH-Port durchzuführen. Zum Inkrafttreten der Änderungen und zur Aktivierung des SSH-jails muss der fail2ban Service gestartet werden.

Listing 13: Starten von fail2ban

```
$ sudo systemctl enable fail2ban
$ sudo systemctl start fail2ban
```

Literatur

[1] *Raspberry Pi Hardening Guide*. URL: https://chrisapproved.com/blog/raspberry-pi-hardening.html (besucht am 12. 03. 2023).

[2] Chris Goff. *Konfigurationsdatei für Autoupdates*. URL: https://gitlab.com/cgoff/raspberry-pi-hardening/blob/master/unattended-upgrades-config/50unattended-upgrades (besucht am 12. 03. 2023).

[3] manpages. *Manpage für UFW*. URL: https://manpages.ubuntu.com/manpages/bionic/en/man8/ufw.8.html (besucht am 12. 03. 2023).

[4] Jasmine Carson. *Exploring the Vulnerabilities and Prevention of Raspberry Pi System*. URL: https://emerging-researchers.org/projects/12496/ (besucht am 12. 03. 2023).

[5] fail2ban. *fail2ban*. URL: https://github.com/fail2ban/fail2ban (besucht am 12. 03. 2023).