Quelle: https://www.fam-moser.de/blog/computer/ubuntu/netzwerk/ubuntu-1404-laender-blockieren-mit-iptables.html

Ubuntu - Länder blockieren mit IPTables

Nachdem ich fail2ban installiert habe und eine Weile das Logging per Mail aktiv gelassen habe und der Schwall an fehlgeschlagenen Versuchen über den ssh Port in meine Server einzubrechen nicht geringer wurde, habe ich beschlossen die betreffenden Länder für den ssh Port komplett zu sperren. Dies galt im besonderen China und Russland, von wo aus die meisten Angriffe stammten.
Glücklicherweise gibt es einige Listen mit Subnetzlisten die man zum Filtern von Traffic aus verschiedenen Ländern verwenden kann. Um diesen Filter zu realisieren benötigt man das xtables-addons-common package sowie die xt_geoip Datenbank.

Außerdem müssen die Kernel Sourcen und die Build-Tools installiert sein, da die Module bei der Installation gebaut werden.

Die folgenden Befehle müssen alle mit root Berechtigungen durchgefürt werden.

Bash

```
apt-get install build-essential linux-headers-generic
```

Installation

Das Paket **xtables-addons-common** ist in den offiziellen Repositories zu finden. Zusätzlich benötigt man für das Parsen der GeoIP Datenbank das Paket **libtext-csv-xs-perl**.

Bash

```
apt-get install xtables-addons-common libtext-csv-xs-perl
```

Damit wird xtables aus den Quellen erzeugt und die entsprechenden Module erzeugt.

Ob die Installation erfolgreich war, kann man testen, indem man folgenden Befehl ausführt:

Bash

```
iptables -m geoip --help
```

Wenn der Befehl erfolgreich ausgeführt werden konnte und die Hilfeseite angezeigt wird, hat die Installation sehr warscheinlich funktioniert.

Geo-Daten importieren

Im nächsten Schritt laden wir die GeoIP CSV Datenbank herunter.

Update Januar 2019

Am 02.01.2019 wurde die Pflege der GeoLite Legacy Datenbank eingestellt. Beim Versuch die Datei GeoIPCountryCSV.zip herunterzuladen bekommt man einen **404 Not Found**.

Glücklicherweise gibt es die neue GeoLite2 Database, die man mit Hilfe des Projektes GeoLite2xtables zu der alten GeoLite Legacy Datenbank kovertieren kann. Dazu müssen die Pakete

- curl
- unzip
- per
- Perl Modul NetAddr::IP

installiert sein.

Bash

```
apt-get install curl unzip perl libnetaddr-ip-perl
```

Die neuen Scripte installiert man sich in dem man das Git Repository cloned:

Bash

```
cd /usr/local/src
git clone https://github.com/mschmitt/GeoLite2xtables.git
```

Dabei werden die Datenbanken nicht mehr mit **/usr/lib/xtables-addons/xt*geoip*dl** heruntergeladen sondern mit den Scripten **00*download*geolite2** und **10*download*countryinfo**. Die Konvertierung übernimmt das Script **20*convert*geolite2**.

Update Januar 2020

Seit dem 30.12.2019 gibt es mal wieder Änderungen was den Download der Dateien angeht. Die Dateien sind aufgrund einer Gesetzesänderung in den USA nicht mehr offen zugänglich. Glücklicherweise hat MaxMind einen Weg gefunden die Dateien trotzdem bereitzustellen. Dazu muss man folgende Dinge tun:

1. Einen kostenlosen Account bei MaxMind erstellen.
2. Den Link in der E-Mail anklicken und das Passwort setzen
3. Einen Lizenz-Key erstellen
4. Das Downloadscript 00*download*geolite2 aus dem GeoLite2xtables Git-Repository händisch anpassen

In der Datei *00downloadgeolite2* muss die Zeile

Bash

```
GEOLITEURL='https://geolite.maxmind.com/download/geoip/database/GeoLite2-Country-CSV.zip'
```

gegen diese ersetzt werden:

Bash

```
GEOLITEURL='https://download.maxmind.com/app/geoip_download?edition_id=GeoL
ite2-Country-CSV&license_key=[Lizenz-Key]&suffix=zip'
```

Der Marker [Lizenz-Key] muss mit dem gerade erzeugten Lizenz-Key von MaxMind ersetzt werden. Anschließend funktioniert das Script wieder wie vor den Änderungen von Dezember 2019.

Import automatisieren

Um die Datenbank regelmäig zu aktualiseren fasst man die benötigten Befehle zu einem Script zusammen:

/usr/local/bin/update*xt*geoip.sh

Bash

```
#!/bin/bash
set -euxo pipefail

TMP_PATH="/tmp/xt_geoip"
SCRIPT_HOME="/usr/local/src/GeoLite2xtables"

$SCRIPT_HOME/00_download_geolite2
$SCRIPT_HOME/10_download_countryinfo

mkdir -p "$TMP_PATH"
mkdir -p "/usr/share/xt_geoip"

cat /tmp/GeoLite2-Country-Blocks-IPv{4,6}.csv | \
  "$SCRIPT_HOME"/20_convert_geolite2 /tmp/CountryInfo.txt > \
  "$TMP_PATH/GeoIP-legacy.csv"

/usr/lib/xtables-addons/xt_geoip_build -D /usr/share/xt_geoip
"$TMP_PATH/GeoIP-legacy.csv"
```

und lässt dieses über einen Cronjob regelmäßig laufen:

/etc/cron.d/cron*xtgeoip*update

cron

```
0 4 * * 1 root /usr/local/bin/update_xt_geoip.sh > /dev/null
```

Alte Vorgehensweise

Im folgenden ist die alte Vorgehensweise zu finden, diese ist nur der Vollständikeit halber drin, wird aber nicht mehr funktionieren.

Bash

```
sudo /usr/lib/xtables-addons/xt_geoip_dl
```

Damit werden zwei Dateien in das aktuelle Verzeichnis heruntergeladen, eine für IPv4 und eine für IPv6. Auf einem meiner Server wurde die IPv4 Datei nicht automatisch entpackt. Sollte das passieren, muss diese noch manuell entpackt werden:

Bash

```
unzip GeoIPCountryCSV.zip
```

Im letzten Schritt werden diese Daten in xtables importiert damit man diese mit iptables verwenden kann. In diesem Schritt benötigen wir übrigens das Paket libtext-csv-xs-perl, welches wir oben installiert haben.

Bash

```
sudo mkdir /usr/share/xt_geoip
sudo /usr/lib/xtables-addons/xt_geoip_build -D /usr/share/xt_geoip *.csv
```

Hier sollte man darauf achten, ob sowohl bei IPv6 also auch bei IPv4 mehr als 0 ranges angezeigt werden. Sind z.B. alle IPv4 ranges 0, dann hat man ggf. vergessen die zip Datei zu entpacken.

Zum Schluss können wir die CSV/ZIP Dateien wieder löschen, da die Daten jetzt importiert sind.

Bash

```
rm GeoIP*
```

Verwendung

Damit man sich nun nicht versehentlich aussperrt kann man eine Rule anlegen die einem auf jeden Fall den Zugriff ermöglicht.

Bash

```
iptables -A INPUT -s 12.34.56.78 -j ACCEPT
```

Zum Testen benötigt man dann aber ein weiteres System mit einer anderen externen IP-Adresse.

Um nun alle IP-Adressen bestimmter Länder für einen Port wie z.B. ssh zu blockieren führt man folgendes aus:

Bash

```
iptables -A INPUT -p tcp --dport 22 -m geoip \
--src-cc KR,CN,IN,RU,SA,TR,VN,UA,BR,VE,PK,JP -j DROP
```

Damit werden jetzt Verbindungsversuche über das tcp Protokoll auf den Port 22 (ssh) aus den genannten Ländern verworfen.

Leider gibt es eine Einschränkung bei iptables. Man kann nicht mehr als 10 Länder angeben. Daher muss man ggf. die Regel invertieren und Länder angeben aus denen der Zugriff erlaubt sein soll.

/etc/iptables/rules.v4

```
iptables -A INPUT -p tcp --dport 22 -m geoip ! --source-country DE,CH,AT -j
DROP
```

iptables-persistent

Um die Regel persistent zu machen, kann man das Paket *iptables-persistent* benutzen. Vor der Installation sollte man aber *fail2ban* stopppen, da man bei der Installation die Möglichkeit hat aktuell bestehende Filterregeln dauerhaft zu sichern.
Da *fail2ban* diese dynamisch anlegt, macht es Sinn dies vorher zu deaktivieren.

Für die Installation führt man dann folgenden Befehl aus:

Bash

```
apt-get install iptables-persistent
```

Das Abspeichern der aktuell verwendeten Regeln kann man mittels iptables-save jederzeit erneut durchführen:

Bash

```
sudo service fail2ban stop
sudo iptables-save > /etc/iptables/rules.v4
sudo service fail2ban start
```

Aus der Datei entfernt man dann alles was man nicht braucht und speichert diese wieder ab.

Bash

```
sudo nano /etc/iptables/rules.v4
```

Die Datei könnte im einfachsten Fall so aussehen:

/etc/iptables/rules.v4

```
# Generated by iptables-save v1.4.21 on Thu Feb 12 19:06:25 2015
*filter
:INPUT ACCEPT [7:905]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [1:88]
-A INPUT -m geoip ! --source-country DE,CH,AT -j DROP
COMMIT
# Completed on Thu Feb 12 19:06:25 2015
```

Das Paket *iptables-persistent* kümmert sich jetzt darum dass diese Regeln bei jedem Neustart wieder geladen werden.

Ufw

Benutzt man Ufw, dann kann man die Regel z.B. in die Datei /etc/ufw/before.rules eintragen:

/etc/ufw/before.rules

```
# Block specific countries from ssh port
-A ufw-before-input -p tcp --dport 22 -m geoip ! --source-country DE,AT,CH
-j DROP
```

Damit sollte die Regel bei jedem Systemstart geladen werden.

Welches Land welchen Ländercode hat, sieht man übrigens wenn man xt*geoip*build ausführt. Außerdem kann man diese in der CSV Datei einsehen. Alternativ kann man diese auch auf https://www.iso.org/obp/ui/#search suchen.
Wenn man *fail2ban* so einstellt, dass man bei jedem Block eine Mail mit den whois Daten des Angreifers bekommt, kann man die Liste mit den Länderkürzeln so sukzessive erweitern bis man endlich wieder beruhigt schlafen kann.

Troubleshooting

Auf meiner JiffiyBox wollte die Installation nicht so richtig klappen. Das xtables Paket lies sich ohne Probleme installieren und auch die Hilfeseite von iptables zu geoip wurde korrekt angezeigt. Wenn ich aber versucht habe eine Regel anzulegen welche die Option **--source-country** verwendet hatte kam es immer zu der Fehlermeldung:

```
No chain/target/match by that name.
```

Es hat sich herausgestellt dass die notwendigen Kernel-Module, darunter auch das xt_geoip Modul, nicht erzeugt werden konnten, weil die JiffyBox einen speziell angepassten Kernel verwendet für den DomainFactory keine Sourcen bereitstellt.

Die Lösung war den Server auf den Distributionskernel umzustellen, die Kernel header herunterzuladen, das Paket xtables-addons-common zu deinstallieren und wieder neu zu installieren.

Bash

```
apt-get remove xtables-addons-common --purge
apt-get install linux-headers-generic
apt-get install xtables-addons-common
```

Bei dieser Installation dann wurden die Module gebaut und das GeoIP Modul lies sich erfolgreich laden mit:

Bash

```
modprobe xt_geoip
```