Installation Ubuntu 20.04 Server

Installationsmedium

> https://releases.ubuntu.com/20.04/

Während der Installation setze ich

- die Locale auf de_de
- das Keyboard-Layout auf „German QWERTZ“

Handwerkszeug installieren

VIMnox

```
# aptitude install vim-nox
```

Midnight Commander

```
# aptitude install mc
```

Net-Tools (ifconfig, etc.)

```
# aptitude install net-tools
```

Timezone

Aktuell eingestellte Zeitzone:

```
# timedatectl
               Local time: Sun 2020-10-11 11:00:01 UTC
           Universal time: Sun 2020-10-11 11:00:01 UTC
                 RTC time: Sun 2020-10-11 11:00:02
                Time zone: Etc/UTC (UTC, +0000)
System clock synchronized: yes
              NTP service: active
          RTC in local TZ: no
```

Zeitzone umstellen:

```
# timedatectl list-timezones|grep Berlin
Europe/Berlin
# timedatectl set-timezone Europe/Berlin
# timedatectl
```

```
               Local time: Sun 2020-10-11 13:02:31 CEST
           Universal time: Sun 2020-10-11 11:02:31 UTC
                 RTC time: Sun 2020-10-11 11:02:32
                Time zone: Europe/Berlin (CEST, +0200)
System clock synchronized: yes
              NTP service: active
          RTC in local TZ: no
```

Reaktivierung von ifupdown

Um netplan.io zu deaktivieren, muss lediglich das Paket ifupdown installiert werden. **Die Deinstallation von netplan.io ist nicht empfehlenswert**, insbesondere dann nicht, wenn die Deaktivierung via SSH vorgenommen wird. Nach der Deinstallation ist ein Zugriff via IP nicht mehr möglich. Es muss auf die Konsole ausgewichen werden!

```
aptitude install ifupdown
```

Im Bootloader muss ebenfalls das Laden von netplan unterdrückt werden:

/etc/default/grub

```
[...]
GRUB_CMDLINE_LINUX="netcfg/do_not_use_netplan=true"
```

```
update-grub
```

Um das klassische Verhalten von ifupdown wiederherzustellen, muss ebenfalls systemd-networkd ausgeschaltet werden. Dies geschieht folgendermaßen:

```
systemctl disable systemd-networkd.service
systemctl mask systemd-networkd.service
systemctl stop systemd-networkd.service
```

Die Netzwerkkonfiguration sollte nun komplett aus der interfaces-Datei übernommen werden. Eine Ausnahme stellen die DNS-Server dar. Damit diese ebenfalls aus interfaces übernommen werden, muss systemd-resolved ausgeschaltet und resolvconf aktiviert werden!

```
aptitude install resolvconf
```

```
systemctl disable systemd-resolved.service
systemctl stop systemd-resolved.service
systemctl mask systemd-resolved.service
```

```
systemctl disable systemd-networkd-wait-online.service
systemctl stop systemd-networkd-wait-online.service
systemctl mask systemd-networkd-wait-online.service
```

```
reboot
```

Firewall

Installation

Die Pakete „iptables-persistent“ und „netfilter-persistent“ stehen in direkter Abhängigkeit und müssen daher beide installiert werden.

```
apt-get update
aptitude install iptables-persistent netfilter-persistent
```

Ubuntu kommt von Hause aus mit dem Paket ufw, ebenfalls eine auf iptables-basierende Firewall. Den Job übernimmt nun netfilter-persistent, daher deinstalliere ich es:

```
aptitude purge ufw
```

Konfiguration / Regelwerk

Um ein Regelwerk zu kreieren, empfehle ich, ein Bash-Skript mit iptables-Befehlen zu schreiben. Sobald dieses ausgeführt worden ist, muss das Regelwerk abgespeichert werden. Dies geschieht mit folgendem Befehl:

```
netfilter-persistent save
```

Netfilter erstellt nun unter /etc/iptables zwei Dateien, rules.v4 und rules.v6. Die Dateien add-blocked.ips sowie blocked.ips stammen von einem eigenen Erweiterungskript, mit dem sich IP-Adressen einfach einer Sperrliste hinzufügen lassen. Darauf werde ich hier nicht weiter eingehen.

```
ll /etc/iptables/
insgesamt 24
drwxr-xr-x  2 root root 4096 Feb  7 23:47 ./
drwxr-xr-x 99 root root 4096 Feb  7 23:18 ../
-rwxr-xr-x  1 root root  742 Feb  7 23:43 add-blocked.ips*
-rw-r--r--  1 root root    0 Feb  7 23:18 blocked.ips
-rw-r-----  1 root root 4189 Feb  7 23:46 rules.v4
-rw-r-----  1 root root  183 Feb  7 23:46 rules.v6
```

Die Firewall sollte nun bereits einsatzfähig sein.

Logfile

Dummerweise schreibt iptables das syslog voll, welches somit unübersichtlich wird. Mit Hilfe des rsyslogd leite ich die Ausgaben in eine eigene Datei um:

```
vi /etc/rsyslog.d/25-iptables.conf
```

Damit dieser Weg funktioniert, habe ich mittels des Parameters –log-prefix von iptables der Ausgabe das Präfix „IPT:“ hinzugefügt. Das könnnen wir uns als Filter zur Nutze machen.

/etc/rsyslog.d/25-iptables.conf

```
:msg,contains,"IPT:" -/var/log/iptables.log
& ~
```

Beim ersten Mal muss die Datei erstellt werden und mit Rechten für den rsyslogd versehen werden.

```
touch /var/log/iptables.log
chown syslog.adm /var/log/iptables.log
```

Die Änderungen werden erst nach einem Dienstneustart übernommen.

```
service rsyslog restart
```

Das Logfile wird schnell groß und sollter daher rotiert werden:

/etc/logrotate.d/iptables

```
/var/log/iptables.log
{
        rotate 7
        daily
        missingok
        notifempty
        delaycompress
        compress
        create 640 syslog adm
        sharedscripts
}
```

Fail2Ban

Fail2Ban sollte meiner Meinung nach auf jeder Maschine laufen, die über SSH im Internet administriert wird. Natürlich ist die Absicherung weiterer Dienste wie SMTP, FTP, usw. ebenso sinnvoll.

Installation und erste Konfiguration

```
# aptitude install fail2ban
```

```
[...]

# "ignoreip" can be a list of IP addresses, CIDR masks or DNS hosts.
Fail2ban
# will not ban a host which matches an address in this list. Several
addresses
# can be defined using space (and/or comma) separator.
#ignoreip = 127.0.0.1/8 ::1
ignoreip = 127.0.0.1/8

[...]

# External command that will take an tagged arguments to ignore, e.g. <ip>,
# and return true if the IP is to be ignored. False otherwise.
#
# ignorecommand = /path/to/command <ip>
ignorecommand =

# "bantime" is the number of seconds that a host is banned.
bantime  = 86400

# A host is banned if it has generated "maxretry" during the last "findtime"
# seconds.
findtime  = 600

# "maxretry" is the number of failures before a host get banned.
maxretry = 6

[...]

#
# JAILS
#

#
# SSH servers
#

[sshd]
enabled = true
# To use more aggressive sshd modes set filter parameter "mode" in
jail.local:
# normal (default), ddos, extra or aggressive (combines all).
# See "tests/files/logs/sshd" or "filter.d/sshd.conf" for usage example and
details.
#mode   = normal
port    = 4444
logpath = %(sshd_log)s
backend = %(sshd_backend)s
```

```
[...]
```

IP entsperren

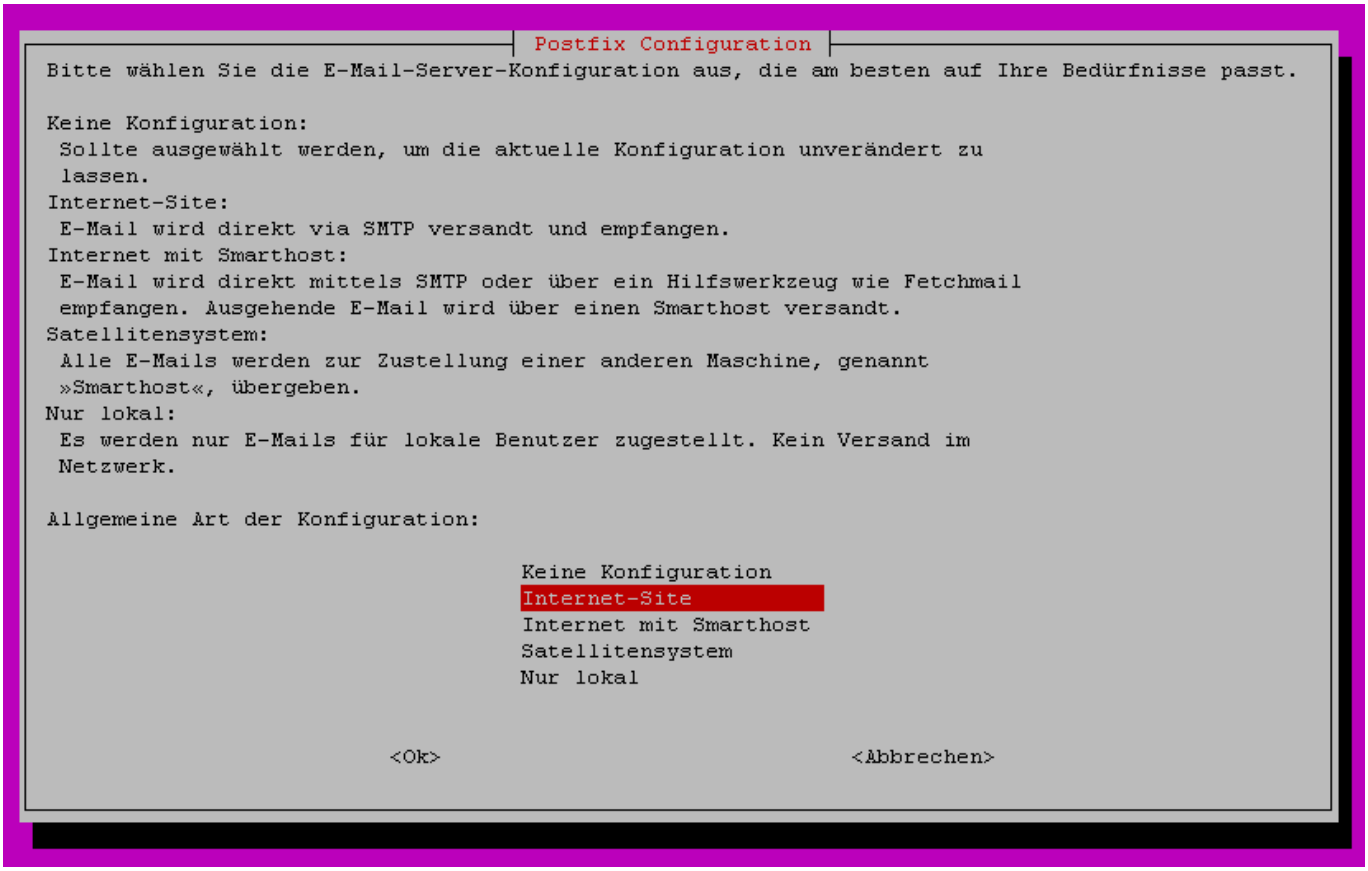
```
# fail2ban-client set <JAIL> unbanip <IP>
```

"Mini" Postfix

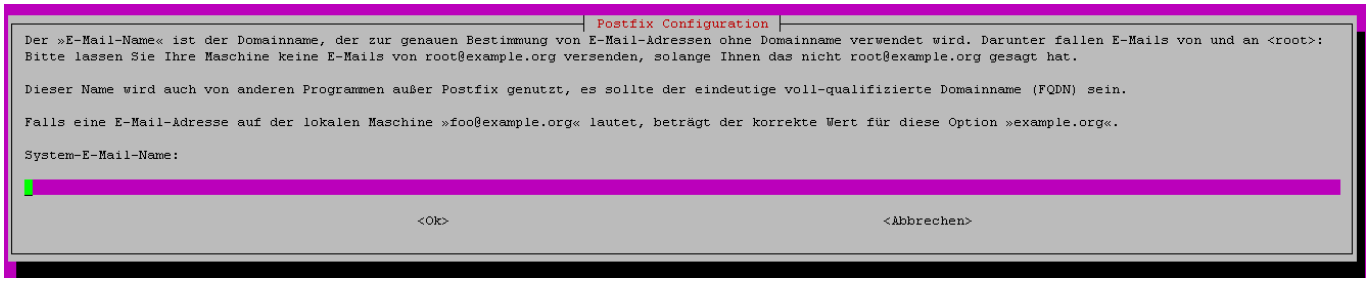
Der MTA Postfix soll nur dazu dienen Mails zu versenden. So können Informationen, zum Beispiel an den Admin, vom System versendet werden oder Webseiten können mit Ihren Benutzern kommunizieren, wenn beispielsweise ein Passwort zurückgesetzt werden soll.

Zunächst sind die benötigten Pakete zu installieren.

```
aptitude install postfix
Die folgenden NEUEN Pakete werden zusätzlich installiert:
  postfix ssl-cert{a}
0 Pakete aktualisiert, 2 zusätzlich installiert, 0 werden entfernt und 8
nicht aktualisiert.
1.164 kB an Archiven müssen heruntergeladen werden. Nach dem Entpacken
werden 4.141 kB zusätzlich belegt sein.
Möchten Sie fortsetzen? [Y/n/?]
```



Hier die Default-Maildomäne eintragen:



Folgende Konfigurationsparameter anpassen:

/etc/postfix/main.cf

```
smtp_generic_maps = hash:/etc/postfix/generic
mydestination = $myhostname, myhostname.mydomain.de, localhost
inet_interfaces = loopback-only
inet_protocols = ipv4
relayhost =  [smtp.myprovider.de]
```

/etc/postfix/generic

```
root@myhostname.mydomain.de             something@mydomain.de
@myhostname.mydomain.de                 @mydomain.de
```

/etc/aliases

```
# See man 5 aliases for format
postmaster:    root
root:   something@mydomain.de
```

Die Konfigurationen anwenden:

```
postmap hash:/etc/postfix/generic
newaliases
service postfix restart
```