Mailserver

Diese Anleitung erstellt keinen Home Mailserver, sondern ein Mail Tool, das eingehende Mails für Monitoring verarbeitet

Install Mailserver

Ich habe mich hier für ein Ubuntu entschieden. Es geht sicher auch mit anderen Distributionen. Mit Ubuntu habe ich allerdings meine meiste eigene Erfahrung.

Ubuntu Server Download Seite

Ich werde hier keinen Postfachserver wie Dovecot oder ähnliches installieren. Es geht rein um das Empfangen und die automatische Verarbeitung von mails.

Den Versende Weg werde ich hier über Strato konfigurieren.

Die Mails sollen per Fetchmail von einem externen Server abgeholt und mit procmail verarbeitet werden.

Server Basis Installation

Zum Testen installiere ich den Server in Parallels auf einem McBook Air M2.

Da es ein Testserver gibt, lasse ich Netzwerk als DHCP.

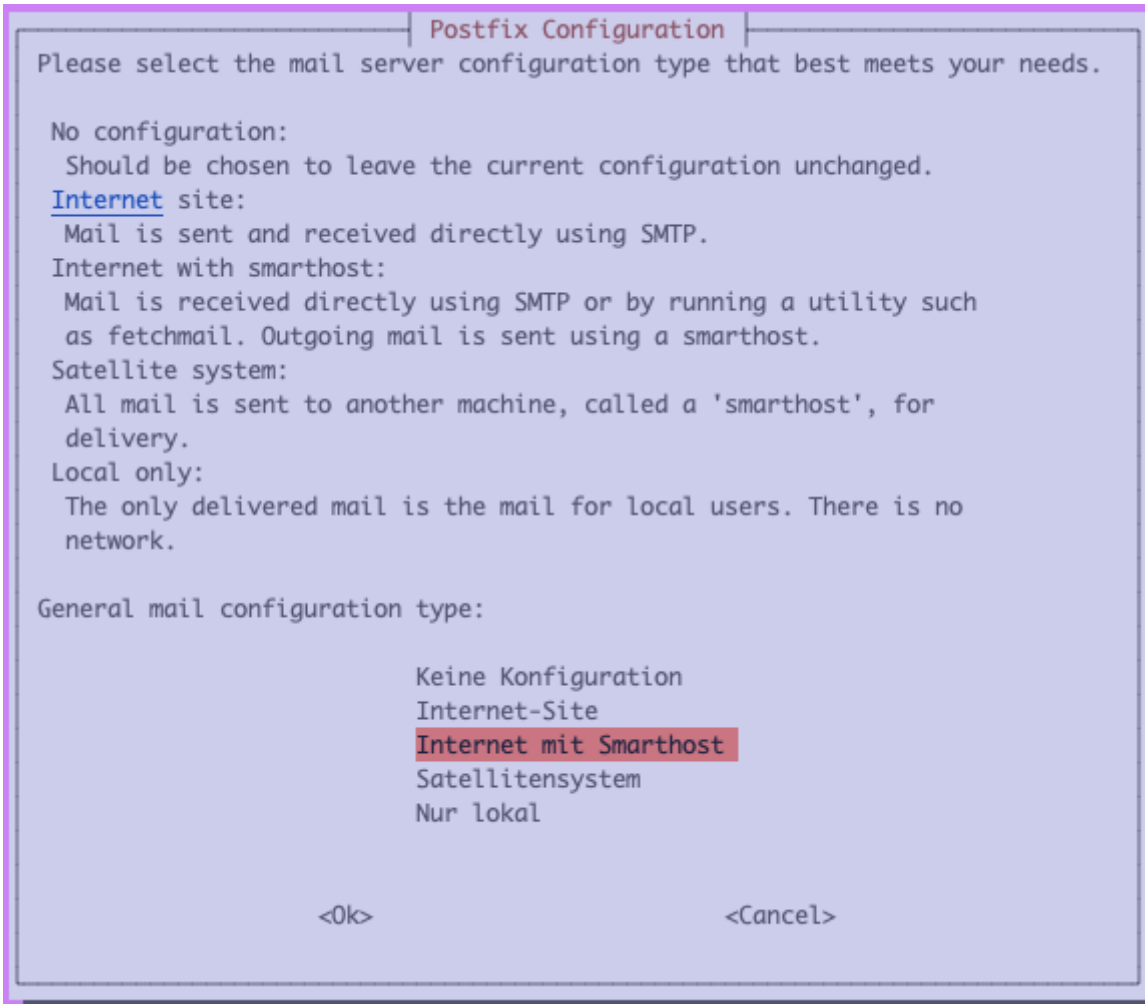
- German und Deutsch einstellen
- Openssh Server installieren
- Sonst keine Pakete auswählen
- LVM automatisch partitionieren

Pakete nachinstallieren

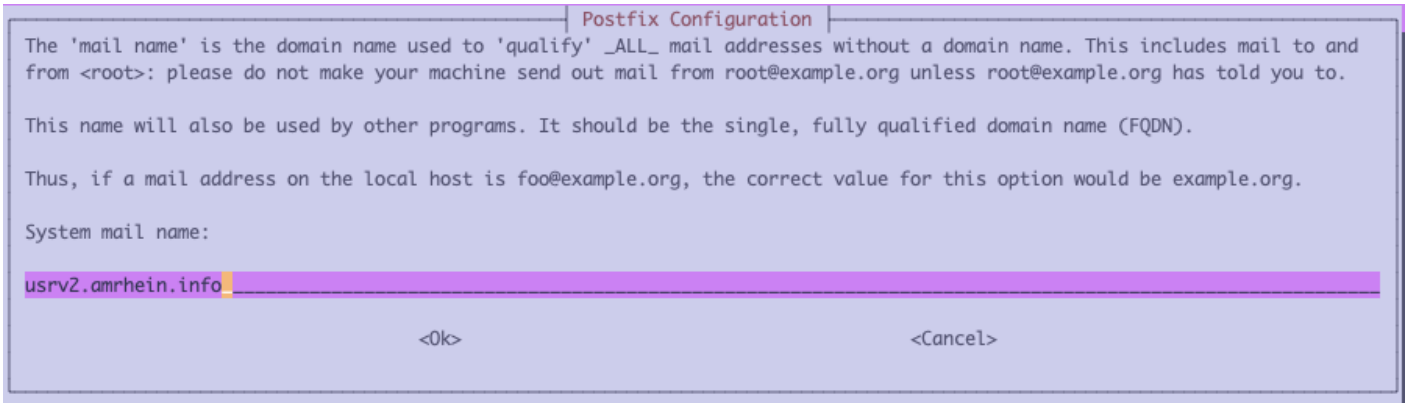
```
sudo apt-get install python3-pip
sudo apt-get install fetchmail procmail mailutils gpg ripmime
sudo apt-get install postfix
```

Mit dem install Kommando startet der Konfiguration Assistent von Postfix.

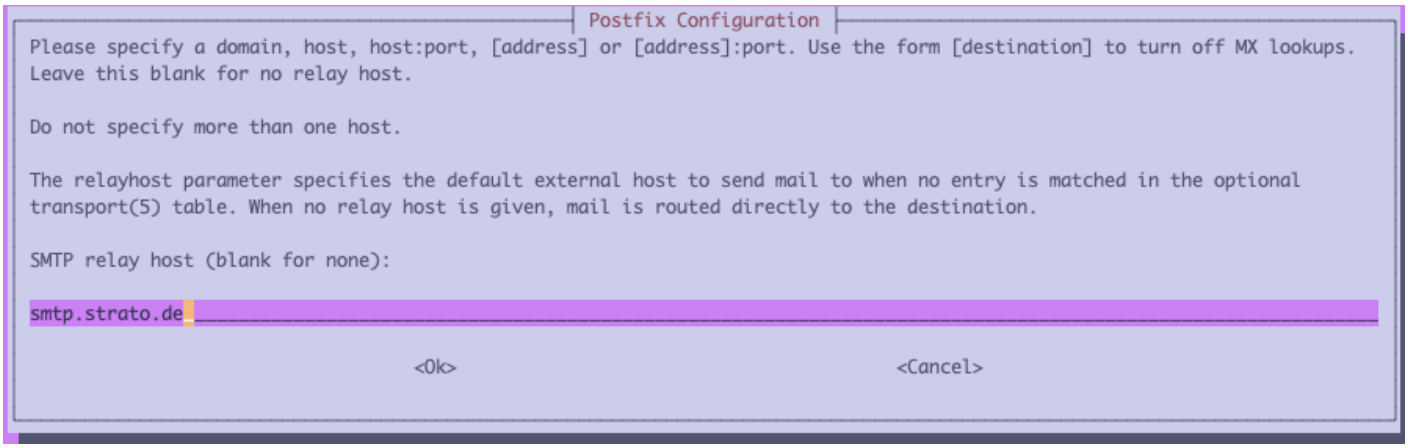
Internet mit Smarthost bedeutet, dass der Server Mails empfängt und via eines SmartHosts (hier dann Strato) versendet.



Unbedingt eine andere Domain als die Hauptdomäne eintragen. Ansonsten würde der Server versuchen alle Mails zu sich selbst zuzustellen.



Hier nun das Mail Delay eintragen. Die Konfiguration dazu folgt später.


Postfix Configuration
Please specify a domain, host, host:port, [address] or [address]:port. Use the form [destination] to turn off MX lookups.
Leave this blank for no relay host.
Do not specify more than one host.
The relayhost parameter specifies the default external host to send mail to when no entry is matched in the optional
transport(5) table. When no relay host is given, mail is routed directly to the destination.
SMTP relay host (blank for none):
smtp.strato.de
<Ok>
<Cancel>

Mailserver Self Signed Zertifikate

Zur Verifizierung und Absicherung werden Zertifikate erstellt.

```
# erstelle den Private Key
openssl genrsa -des3 -out usrv2.amrhein.info.key 2048
chmod 600 usrv2.amrhein.info.key

# erstelle den Zertifikats CSR
openssl req -new -key usrv2.amrhein.info.key -out usrv2.amrhein.info.csr
# erstelle Zertifikat
openssl x509 -req -days 365 -in usrv2.amrhein.info.csr -signkey usrv2.amrhein.info.key -out
usrv2.amrhein.info.crt

# erstelle unverschlüsselten Private Key
openssl rsa -in usrv2.amrhein.info.key -out usrv2.amrhein.info.key.nopass
# backup des passwort gesicherten Keys
mv usrv2.amrhein.info.key usrv2.amrhein.info.key.pass
# verwende den unverschlüsselten private key
mv usrv2.amrhein.info.key.nopass usrv2.amrhein.info.key

# erstelle Server Key und Zertifikat
openssl req -new -x509 -extensions v3_ca -keyout cakey.pem -out cacert.pem -days 3650
chmod 600 usrv2.amrhein.info.key
chmod 600 cakey.pem

# verteile die Zertifikate und Keys
mv usrv2.amrhein.info.key /etc/ssl/private/
mv usrv2.amrhein.info.crt /etc/ssl/certs/
mv cakey.pem /etc/ssl/private/
mv cacert.pem /etc/ssl/certs/

# postfix Konfiguration
postconf -e 'smtpd_tls_auth_only = no'
```

```
postconf -e 'smtp_use_tls = yes'
postconf -e 'smtpd_use_tls = yes'
postconf -e 'smtp_tls_note_starttls_offer = yes'
postconf -e 'smtpd_tls_key_file = /etc/ssl/private/usrv2.amrhein.info.key'
postconf -e 'smtpd_tls_cert_file = /etc/ssl/certs/usrv2.amrhein.info.crt'
postconf -e 'smtpd_tls_CAfile = /etc/ssl/certs/cacert.pem'
postconf -e 'smtp_tls_CApath = /etc/ssl/certs'
postconf -e 'smtpd_tls_loglevel = 1'
postconf -e 'smtpd_tls_received_header = yes'
postconf -e 'smtpd_tls_session_cache_timeout = 3600s'
postconf -e 'tls_random_source = dev:/dev/urandom'
postconf -e 'myhostname = mail.example.com'

# restart postfix
systemctl restart postfix
```

Fetchmail

Um die Mails aus einem vorhandenen Postfach abzuholen konfiguriere ich fetchmail für den Benutzer.

Fetchmail muss dann allerdings manuell gestartet werden.

Starten: "fetchmail -vvvv" - die vielen V's für die sehr verbose Ausgabe.

Stoppen: "fetchmail -q" - stoppt den Daemon Prozess

Um Fetchmail als Daemon laufen zu lassen muss dies zugelassen werden.

Dafür in den Defaults "START_DAEMON" auf "yes" setzen.

```
sudo vi /etc/default/fetchmail

# This file will be used to declare some vars for fetchmail
#
# Uncomment the following if you don't want localized log messages
# export LC_ALL=C

# If you want to specify any additional OPTION to the start
# scripts specify them here
# OPTIONS=...

# Declare here if we want to start fetchmail. 'yes' or 'no'
START_DAEMON=yes
```

Die Benutzerkonfiguration erfolgt in ~/.fetchmailrc

```
set daemon 10                # Abfrageintervall 10 sec - nur zum testen
set syslog
set postmaster  root

set no bouncemail
defaults:
timeout 300
antispam -1
```

```
batchlimit 100

poll imap.strato.de
  proto imap
  user "user@amrhein.info"
  is parallels                # lokales Postfach
  pass "XXXXXX"
  folder "testmails"          # abzuholender Ordner
  ssl
  sslproto tls
  fetchall
  nokeep
```

Maildir anstatt MBOX Format für Benutzer verwenden

Ich möchte die Mail Dateien aus diversen Gründen erhalten und diese im Maildir Format ablegen.

https://wiki.debian.org/MaildirConfiguration

Konfiguration Postfix

Alle Konfiguration erfolgt in /etc/postfix/main.cf

```
smtpd_banner = $myhostname ESMTP $mail_name (Ubuntu)
biff = no

# appending .domain is the MUA's job.
append_dot_mydomain = no

# Uncomment the next line to generate "delayed mail" warnings
#delay_warning_time = 4h

readme_directory = no

# See http://www.postfix.org/COMPATIBILITY_README.html -- default to 3.6 on
# fresh installs.
compatibility_level = 3.6



# TLS parameters
smtpd_tls_cert_file = /etc/ssl/certs/usrv2.amrhein.info.crt
smtpd_tls_key_file = /etc/ssl/private/usrv2.amrhein.info.key
smtpd_tls_security_level=may

smtp_tls_CApath = /etc/ssl/certs
smtp_tls_security_level=may
smtp_tls_session_cache_database = btree:${data_directory}/smtp_scache

smtpd_relay_restrictions = permit_mynetworks permit_sasl_authenticated defer_unauth_destination
myhostname = usrv2.amrhein.info
alias_maps = hash:/etc/aliases
alias_database = hash:/etc/aliases
myorigin = /etc/mailname
mydestination = usrv2.amrhein.info, $myhostname, usrv2, localhost.localdomain, localhost
relayhost = smtp.strato.de:587
```

```
smtp_tls_security_level = encrypt
smtpd_relay_restrictions = permit_mynetworks permit_sasl_authenticated defer_unauth_destination
smtp_sasl_auth_enable = yes
smtp_sasl_password_maps = hash:/etc/postfix/sasl_passwd
smtp_sasl_security_options = noanonymous

mynetworks = 127.0.0.0/8 [::ffff:127.0.0.0]/104 [::1]/128
mailbox_command = procmail -a "$EXTENSION"
mailbox_size_limit = 0
recipient_delimiter = +
inet_interfaces = all
inet_protocols = all
home_mailbox = Maildir/

smtpd_tls_auth_only = no
smtp_use_tls = yes
smtpd_use_tls = yes
smtp_tls_note_starttls_offer = yes
smtpd_tls_CAfile = /etc/ssl/certs/cacert.pem
smtpd_tls_loglevel = 1
smtpd_tls_received_header = yes
smtpd_tls_session_cache_timeout = 3600s
tls_random_source = dev:/dev/urandom
```

der Benutzer für das SMTP Relay wird als Hash abgelegt

/etc/postfix/sasl_passwd

```
smtp.strato.de relayuser@domain.tld:passwort
```

Progmail / GPG / Test

Alle eingehenden Mails sollen automatisiert verarbeitet werden.

LIVE: export Check Config oder State an Check Monitoring Sites

TEST: touch field with Date in File and Dir name

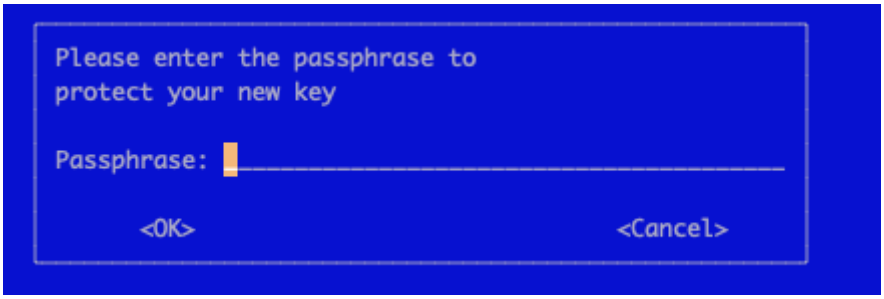
Die Dateien werden gpg verschlüsselte Anhänge haben. Um diese zu extrahieren verwende ich das Tool ripmime

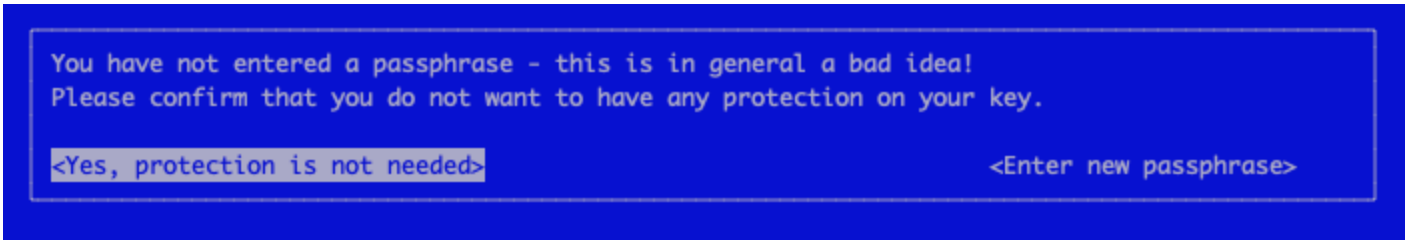
GPG Initialisierung

https://www.hauke-laging.de/sicherheit/openpgp.html

```
root@vm-mail:~# gpg --full-generate-key
gpg (GnuPG) 2.2.27; Copyright (C) 2021 Free Software Foundation, Inc.
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.

Please select what kind of key you want:
   (1) RSA and RSA (default)
   (2) DSA and Elgamal
   (3) DSA (sign only)
   (4) RSA (sign only)
  (14) Existing key from card
Your selection? 1
RSA keys may be between 1024 and 4096 bits long.
What keysize do you want? (3072)
Requested keysize is 3072 bits
Please specify how long the key should be valid.
         0 = key does not expire
      <n>  = key expires in n days
      <n>w = key expires in n weeks
      <n>m = key expires in n months
      <n>y = key expires in n years
Key is valid for? (0) 0
Key does not expire at all
Is this correct? (y/N) y

GnuPG needs to construct a user ID to identify your key.

Real name: USRV-MAIL
Email address: admin@amrhein.info
Comment:
You selected this USER-ID:
    "USRV-MAIL <admin@amrhein.info>"

Change (N)ame, (C)omment, (E)mail or (O)kay/(Q)uit? o
```

Danach soll eine Passphrase eingegeben werden. Hier zum Test lasse ich das leer und bestätige das im nächsten Schritt. Die Aufforderung kommt 2x.





.procmailrc

Diese Datei steuert die procmail Verarbeitung.

Jedes Mail wird als Betreff haben "STATE <Chckmk Site name>"

Das das Regex Element nach '\/' füllt die Variable MATCH.

```
ORGMAIL=$HOME/Maildir
DEFAULT=$ORGMAIL
MAILDIR=$HOME/Maildir
VERBOSE=on
PMDIR=$HOME/.procmail
LOGFILE=$HOME/procmaillog
SHELL=/bin/bash
DATE=`date +%Y%m%d_%H-%M-%S`

SUBJECT=| formmail -xSubject:
DELIVERY=| formmail -xX-Delivery-Time:
DTIME=`echo $DELIVERY | date +%Y%m%d_%H-%M-%S`

:0
* ^Subject:.*STATE \/.*
| mkdir -p $HOME/received/$MATCH/$DTIME && ripmime --overwrite --no-nameless -i - -d
$HOME/received/$MATCH/$DTIME/ && $HOME/extract $MATCH/$DTIME
```

TEST

zum Test nutze ich ein bashscript "sendstate". Die Verarbeitung das Skript "extract"

```
#!/bin/bash

sendPATH="$HOME/send"
mkdir -p $sendPATH

cd $sendPATH

touch state
tar czvf state.tgz state
```

```
gpg -e --batch --yes -r USRV2-MAIL -o state.gpg state.tgz

for s in site01 site02 site03 site04 site05 site06 site07 site08 site09 site10 site11 site12 site13 site14 site15
site16 site17 site18 site19 site20
do
echo "STATE $s" | mail -s "STATE $s" -A state.gpg -r gerald@amrhein.info admin@amrhein.info
done
```

```
#!/bin/bash

recvPATH="$HOME/received/$1"

cd "$recvPATH"

gpg -d --batch --yes -r USRV2-MAIL state.gpg >state.tgz
tar xzvf state.tgz
myDATE=$(echo $(stat state -c %Z) | date +%Y%m%d_%H-%M-%S)
mv state ${myDATE}_state
```

Fetchmail SSL Fingerprint

von Geiststreicher am 30. April 2014

in HowTo, Linux, Tipps & Tricks

Was lange währt, wird endlich gut und wurde als DAS neue Sicherheitsfeature von GMX und web.de verkauft. Die SSL-Verschlüsselung beim Abruf von Mails per POP3. Ruft man seine Mails auf einem Linux-Rechner per fetchmail ab, muss man unter Umständen ein wenig an seiner Konfiguration drehen. Wie es funktioniert, möchte ich hier kurz erklären.

Wie ich in meiner Artikelserie „Bye, bye, cloud“ schon angedeutet habe, rufe ich meine verschiedenen Postfächer automatisch per fetchmail ab und führe sie auf meinem Server zusammen. Die Einzelheiten dazu findet ihr demnächst im „Bye, bye, cloud“-Artikel, der sich speziell mit meiner Mailkonfiguration befasst.

Hier soll es nur kurz um die Konfiguration von fetchmail im Speziellen gehen und darum, wie man das aktuelle Zertifikat und den Fingerprint für die SSL-Verschlüsselung bekommt und nutzt. Alles, was ich im folgenden exemplarisch für GMX sage, gilt 1:1 auch für web.de und lässt sich leicht auf andere Email-Dienste übertragen.

Als zusätzliches Beispiel gibt es noch die Befehlszeile für die Ermittlung des Fingerprints über einen IMAP-Abruf mit STARTTLS.

Fetchmail

Der Eintrag in meiner fetchmailrc sieht folgendermaßen aus:

```
poll pop.gmx.net with service pop3s interval 6
    user '1234567' there with password 'MyVeryS3cr3tP@assword' is sascha here options fetchall
    ssl
    sslproto tls1
    sslfingerprint "8A:B7:78:CF:0D:73:4E:EE:FF:EB:B8:C0:90:7D:46:56"
    sslcertck
    sslcertpath /etc/ssl/certs
```

Wie man sieht rufe ich meinen GMX-Account per pop3s ab. Der Service ist in /etc/services definiert und bestimmt den Port, mit dem auf GMX zugegangen wird. Im Falle eines SSL-gesicherten Abrufs ist das fast immer Port 995.

Anschließend sieht man die Anmeldung am GMX-Server und welcher Benutzer auf meinem Server das Empfängerkonto ist.

Der interessante Teil bezüglich der Sicherheit spielt sich in den folgenden Zeilen ab. Zuerste wird prinzipiell der Abruf per SSL „eingeschaltet“, aschließend die Protokollversion festgelegt. Dann wird sowohl der Server-Fingerprint überprüft, wie auch das komplette Zertifikat, dass man vorher im /etc/ssl/certs/ Directory abgelegt haben muss. Schlägt eine der Prüfungen fehl, findet keine weitere Datenübertragung statt.

In einer Mehrbenutzerumgebung würde man das Passwort natürlich nicht in die fetchmailrc schreiben, sondern ggf. in einer Userspezifischen .netrc.

Zertifikat abrufen

Das Serverzertifikat wird übermittelt, wenn man eine SSL-Verbindung mit GMX aufbaut. Das kann man auf der Console mit folgendem Befehlt tun:

```
openssl s_client -connect pop.gmx.net:995 -showcerts
```

In der Ausgabe sieht man alle Zertifikate der Trust-Chain. An den Bezeichnungen erkennt man, dass das erste Zertifikat auf die „1&1 Mail und Media GmbH“ ausgestellt ist. Dieses kopiert man einschließlich der Zeilen `-----BEGIN CERTIFICATE-----` und `-----END CERTIFICATE-----` und speichert es in einer Datei im Verzeichnis /etc/ssl/certs/

Anschließend muss man in dem Directory noch ein `c_rehash` ausführen!

Fingerprint ermitteln

Der Fingerprint leitet sich vom gespeicherten Zertifikat ab und kann ebenfalls über einen openssl-Befehl ermittelt werden:

```
openssl x509 -dates -fingerprint -md5 -noout -in /etc/ssl/certs/gmx_de.pem
```

Dabei gehe ich davon aus, dass das im vorherigen Abschnitt ermittelte GMX-Zertifikat in /etc/ssl/certs/gmx_de.pem gespeichert ist. Der Befehlt gibt zusätzlich zum Fingerprint auch die Gültigkeit mit aus.

Hat man das Zertifikat noch nicht lokal vorliegen, kann man den Fingerprint auch mit einer Befehlszeile ausgeben lassen, die den Connect und die Formatierung der Ausgabe komplett übernimmt.

In diesem Beispiel erfolgt der Abruf über IMAP von meinem Provider Uberspace mit STARTTLS:

```
openssl s_client -connect vulpecula.uberspace.de:143 -starttls imap </dev/null  2>/dev/null   | sed -n
/BEGIN/,/END/p  | openssl x509 -dates -fingerprint -md5  -noout
```