Genmailmodule manual

Generate, using a script, which mimics an enhanced yast2-mail module,
a postfix system, and optional a dovecot server and
security features, like virus scanning, SPF and DMARC checking
and DKIM checking and signing.

Introduction

This manual is about a bash script which is a parallel development of what is presented on the wiki page https://en.opensuse.org/Mail_server_HOWTO, however it is meant to provide an enhanced version of what the yast2-mail module should do. This manual describes a procedure to configure postfix on an openSUSE system with Leap 15.5 on a Raspberry Pi 4B, just like what an enhanced version of the yast2 mail module does. With this script you can configure the same type of email server the previous script could do, except Masquerading, but makes it possible to include SPF and DMARC checking. DKIM signing and checking was present with opendkim, but configuring it was left to the user. So not very easily usable. This can also been done using the build-in DKIM support in amavis. The script also support configuring dovecot as an imap server. Generating self-signed certificates for both postfix and dovecot is also made more accessible.

The structure of the system

The mail module, `/sbin/yast2-mail.sh`, does preliminary work. Depending on what is required it does install the needed packages and does some configuration for accompanying packages, but most configuration is done with the new version of `/usr/sbin/config.postfix`. According to choices the user makes, it assigns values to parameters in a new version of `/etc/sysconfig/postfix` and in `/etc/sysconfig/mail` and `/etc/sysconfig/amavis`. These values are later used by the script `/usr/sbin/config.postfix` to configure the main configuration files of postfix, main.cf and master.cf. Also configuration files of amavis, dovecot, opendkim and opendmarc, if used, are configured and the self-signed certificates for postfix and possibly dovecot. Running these scripts after each other should give you a working email system.

Preliminary actions

Preparing the boot device of the server

Testing the scripts has been done in a fresh installed system. The structure allows re-running the scripts in case not much will be changed. As an example, a Raspberry Pi 4B image of Leap 15.5, meant for a headless system has been used for testing. Using the following commands on a host Linux system will download and check the image.

```
repository="http://download.opensuse.org/distribution/leap/15.5/appliances"
image="openSUSE-Leap-15.5-ARM-JeOS-raspberrypi.aarch64.raw.xz"
mkdir -p image
cd image
curl -OL $repository/${image}.sha256
curl -OL $repository/$image
cat ${image}.sha256 | sha256sum -c -
```

After this you insert the boot device for the Raspberry Pi in a USB slot or SD slot of this computer. This device can be a microSD card in an SD envelop or a USB disk.

After this you have to find on what device the boot device is present. The following command can be used to find out. Give this command before inserting the boot device and after inserting and deduce from the difference the device.

```
ls -l /dev/sd*
```

Most likely it is the last device in the list. In my case is was /dev/sdd ; not the one with a digit at the end. After that you give the following commands, the first ones are only necessary when your device has been used earlier for other purposes. You will be asked for the root password. ***Obviously you need to replace sdd in the following command lines by your device name.***

```
ssize=$(sudo /usr/sbin/blockdev --getss /dev/sdd)
echo Sector size = $ssize
```

The sector size needs to be 512, otherwise you need to investigate what the next commands need to be in that case. Next we clear the last blocks of the device and write the image on the boot device.

```
size=$(sudo /usr/sbin/blockdev --getsz /dev/sdd)
sudo dd if=/dev/zero of=/dev/sdd obs=1 seek=$(($size - 2)) count=2
sudo xzcat -v $image | sudo dd bs=4M of=/dev/sdd iflag=fullblock oflag=direct
sudo sync
```

The next step is to insert the microSD card or the USB device in the proper slot of the Raspberry Pi and power the device on. The system is meant to be a server, so it is better to connect the system via the Ethernet interface with your network.

Connect to the server

You need to find the IP address of the system. It will get an IP address via DHCP, so your router might provide this address. Because the system is assumed to be headless you connect to the system via ssh with:

```
ssh root@<IP_address>
```

This will ask you to accept the key from this system and asks you for the username, *root*, and the password, *linux* . The first thing to do is to change the password of root with the command `passwd`.

Assign a fixed IP address to the server

This might be the time to give your system a fixed IP address, but you can accept the given address by DHCP as well. In that case it is best to assign in your DHCP server/router a fixed assignment of the MAC address of your server to the given IP address. Anyway it is assumed that from now on the IP address of the server is fixed.

Downloading the scripts

You may already have the scripts downloaded together with this manual, which is done by performing the following commands:

```
zypper in git
```

```
git clone https://github.com/freekdk/GenPDSDM.git
```

This will load the scripts and this documentation in the folder GenPDSDM. However a better way is to install the new version of postfix, which also installs the new versions of the files in GenPDSDM.

Configuring the email system

Preparations

The first thing to do in this virgin system is bringing it up-to-date with:

```
zypper up
```

If you are using another distribution than the one mentioned above for the Raspberry Pi, check if postfix is already active. If so, stop it and remove it with:

```
systemctl stop postfix.service
zypper rm postfix
rm /var/adm/postfix.configured
```

The last command may fail, which is OK, see later why it might be there.

Install postfix from the repository:
https://download.opensuse.org/repositories/home:/fdekruijf:/branches:/server:/mail/<type>,

where <type> is: 15.4, 15.5, openSUSE_Tumbleweed, or openSUSE_Tumbleweed_RISCV.

On the Raspberry Pi it is:

```
zypper ar
https://download.opensuse.org/repositories/home:/fdekruijf:/branches:/server:/
mail/15.5 server-mail-fdekruijf
zypper in postfix
```

You may have to accept the key of the repository and remove a conflicting package.

```
systemctl start postfix.service
```

Starting postfix, when it has not been started before, triggers execution of /usr/sbin/config.postfix. On other systems postfix might already be installed, and because it is installed, and enabled for

systemd, it will start when booting the system for the first time. However this first start creates the file which prevents executing /usr/sbin/config.postfix when postfix starts again by “using systemctl start postfix.service” or after a reboot.

This explains why you had to remove the file /var/adm/postfix.configured when postfix was active before installation of the enhanced postfix.

This also might be the time to define a normal user with:

```
useradd -c "John Doe" -m -u 1000 -p test john
```

and changing the password for this user:

```
passwd john
```

This user is meant to receive the messages for root, so a change in the /etc/aliases file:

```
sed -E -i '/^#root:[[:blank:]]+/ a root:\t\tjohn' /etc/aliases
newaliases
```

Maybe it is time for a few small tests. Execute the following commands to send test messages:

```
zypper in mailx
mailx -s test john < /dev/null
mailx -s test john@example.net < /dev/null
```

where john@example.net is an email address you have somewhere else. The first message remains within this system and the second one tries to go outside, but may be bounced. Postfix will use the MX record of example.net to send the message to. Examine the postfix log with:

```
tail -20 /var/log/mail
```

and the received messages with:

```
tail -40 /var/spool/mail/john
```

and if the outgoing message succeeds the mailbox on example.net.

Copying previously generated items

There are two items that may have been generated earlier or are needed in this system also.

The first one is /etc/dovecot.dh.pem. It might take quite some time to generate this item. On the Raspberry Pi used to develop this system it took once 160 minutes. It may have been generated somewhere else.

The second item is the content of the folder /var/sb/dkim/. It contains one or more private keys for specific domains and the TXT records with the public key for these domains. These are used both by DKIM supported in packet amavisd-new as supported by packet opendkim. For the domain example.com the names are: example.com.dkimYYYYMMDD.pem and example.com.dkimYYYYMMDD.txtrecord, where YYYYMMDD denotes the date they are generated. When you already have the TXT record in the DNS for this/these domain(s) you do not need to generate a new one. The script `/sbin/config.postfix` will skip the generation of these files when they are present in that location.

Executing yast2-mail.sh

The original version of the yast2 mail module gives a choice between sendmail and postfix. In case none of these packages are installed it asks which one should be installed. Installation of the current available package postfix or sendmail does not provide what the aim of this manual is, that's why we made the above preparations and the first output shows we are using postfix..

Below is an example of executing this script for the first time when all answers are correct. The test which is typed is marked in green with a yellow marking. Start the script with:

```
/usr/sbin/yast2-mail.sh
```

```
================================
= We will continue with Postfix =
================================

====================
= General settings =
====================
Connection type for email server
There are four options for this type
Permanent, Dial-up, No connection or Do not start as Daemon
Enter P, D, c or o : p
```

Another answer than p will not give us the extra services this system offers.

```
A null client is a machine that can only send mail. It receives no
mail from the network, and it does not deliver any mail locally.
It will send locally submitted messages onto the relayhost.
Currently you did NOT configure a nullclient type postfix.
Is this OK?
Enter Y[es] or n[o] : y
```

Again another answer than n will not bring us the extra services.

```
Do you want outgoing email to a relayhost (the server of your provider)?
Enter Y[es] or n[o] : y
```

The recommended answer is yes. Otherwise messages will go directly to port 25 of the server of the destination in the email address, which often will not accept the message.

```
Enter the name of the relayhost and the entry port like :
(maybe :port is not necessary) smtp.provider.tlp:port : smtp.example.net:587
```

Your provider did provide the details about how to submit messages to the server.
Note: the script will check whether the name you entered does have an IP address in the Domain Name Server (DNS). So you might see packages loading to perform this test.

```
Should postfix lookup the MX record of smtp.example.net.
Enter Y[es] or n[o] : n
```

The recommended way is no. Normally your provider specifies the name to be looked up without using an MX record.

```
Enter the username for access to smtp.example.net, often an email address :
john.doe@example.net
```

This most likely is your email address at your provider or something similar.

```
Enter the password for access to smtp.example.net : AbCdeFg12345
```

This password the username and the relayhost are stored in the file /etc/postfix/sasl_passwd. The value of relayhost will also be a parameter in /etc/postfix/main.cf.

```
Currently outgoing email will not be encrypted
Is this OK?
Enter Y[es] or n[o] : n
```

It is recommended to encrypt outgoing messages.

```
Three posibilities: No, If possible, or Always. If possible is recommended.
Your choice is: No?
Enter Y[es] or n[o] : n
Your choice is: If possible?
Enter Y[es] or n[o] : y
```

The recommended value is chosen.

```
Currently remote access to your incoming port 25 is not enabled.
Do you want access enabled? This means that other parameters will be made
consistent.
Enter Y[es] or n[o] : y
```

The obvious answer is yes..

```
Currently virus scanning with AMaVis is not enabled
However it is recommended to use this when listening to remote servers.
Is this OK?
Enter Y[es] or n[o] : n
```

Answering no means that the package amavisd-new and a lot of other packages will be installed.

```
Currently you allow incoming email on port 25, but it is not encrypted?
Do you want encrypted access?
Enter Y[es] or n[o] : y
```

The answer yes implies that you need a self-signed certificate, the details of which will be asked later on.

```
Currently you only allow encrypted incoming email on port 25.
Do you also want authenticated access on port 587?
Enter Y[es] or n[o] : y
```

Yes means that you can submit messages from anywhere to your email server with sender addresses some_name@example.com to anywhere in the world, where example.com is your domain name. When DKIM support is enabled these messages will be signed by DKIM.

```
Currently you have enabled basic spam prevention with type off
Possibilities are: none, medium, hard, and custom.
Is the current one OK?
Enter Y[es] or n[o] : n
```

With no the next question will be:

```
Enter none, medium, hard, or custom : hard
```

In fact your choice here is medium or hard. With hard there is the possibility to blacklist email from servers, IP addresses, that are known to send spam or which are entered by providers who do not want their customers to send non-authorized messages.

```
You did not specify any server with hosts to be blacklisted.
Do you want anything in your list? Options will be given.
Enter Y[es] or n[o] : y
```

Yes is recommended.

```
Enter digit 1 or a combination of 2,3, and 4, which belong to the following
options:
1: nothing in the list of servers with blacklisted hosts
2: server bl.spamcop.net
3: server cbl.abuseat.org
4: server zen.spamhaus.org
4
```

Rather arbitrary the answer is 4, but 1, or 234 or other combinations are also valid answers.

```
Trying to find host name and domain name...
Enter the name of the system (should not contain a dot) : rpi-host
```

This is the local name of the system, which name does not need to be exposed to the outside world.

```
Enter the domain name (should contain a dot): example.com
The domain name found with 'hostname -f', localhost,  does not correspond
with the one entered or in /etc/hosts. Assumed is: 'example.com' is the right
one
Checking for existing records in the DNS
The system will reboot now to show the system name rpi-host.example.com and
please run this script again
```

As indicated the script checks whether the domain name example.com has the proper entries in the DNS.
The value of the host name will be entered in /etc/hostname and an entry with rpi-host.example.com will be made in /etc/hosts. After a reboot the command ‘hostname -f’ will provide the fully qualified host name rpi-host.example.com.

You have to access the system again using the following command and execute the script again:

```
ssh root@<IP_address>

/usr/sbin/yast2-mail.sh
================================
= We will continue with Postfix =
================================

====================
= General settings =
====================
Connection type for email server
There are four options for this type
Permanent, Dial-up, No connection or Do not start as Daemon
Enter P, D, c or o : p
```

```
A null client is a machine that can only send mail. It receives no
mail from the network, and it does not deliver any mail locally.
It will send locally submitted messages onto the relayhost.
Currently you did NOT configure a nullclient type postfix.
Is this OK?
Enter Y[es] or n[o] : y
Currently you want outgoing email to relayhost: smtp.example.net:587
Is this OK?
Enter Y[es] or n[o] : y
Username and password may be in /etc/postfix/sasl_passwd and possibly are:
johndoe@example.net : AbCdeFg12345
Is this OK?
Enter Y[es] or n[o] : y
Currently outgoing email will be tried to be encrypted
Is this OK? Alternatives are: No, If possible, or Always.
Enter Y[es] or n[o] : y
Currently remote access to your incoming port 25 is enabled.
Is this OK?
Enter Y[es] or n[o] : y
Currently virus scanning with AMaVis is enabled.
Is this OK?
Enter Y[es] or n[o] : y
Currently you allow incoming email on port 25 with encryption.
Is this OK? Otherwise encryption on this port will be removed.
Enter Y[es] or n[o] : y
Currently you allow encrypted incoming email on port 25 and authenticated access
on port 587.
Is this OK? Otherwise authenticated access on port 587 will be removed.
Enter Y[es] or n[o] : y
Currently you have enabled basic spam prevention with type hard
Possibilities are: none, medium, hard, and custom.
Is the current one OK?
Enter Y[es] or n[o] : y
You have specified to blacklist hosts from the server(s): zen.spamhaus.org
Is this OK?
Enter Y[es] or n[o] : y
Trying to find host name and domain name...
Currently your hostname is: rpi-host
Is this OK?
Enter Y[es] or n[o] : y
The domain name to be used is example.com
```

In the previous part all the previous settings were displayed and the choice was made to keep it that way. After this the following question will be asked:

```
Currently you want to use cyrus for authentication
Is this OK?
Enter Y[es] or n[o] : n
```

No means that dovecot will be used for authentication, which also means that the dovecot imap server will be implemented. This also means that email messages will be delivered in the Maildir format (each message in a separate file in sub folders of the Maidir folder in the home folder of the user.
In case cyrus will be used, the script will install what is necessary for cyrus, but now the dovecot package will be installed, but only the really necessary ones.

```
Information for generating a self-signed certificate for postfix
Dovecot authentication also means a secured imap server by dovecot,
```

```
which also needs a self-signed certificate
Information to generate a self-signed certificate
```

The next output of the script is self explained, so it will not be interlaced with comment.

```
Enter the two letter code of your country : AA
Enter the name of your state/province : Some-State
Enter the name of your city/locality : Some-city
Enter the name of your organization : ACME company
Enter the name of your organizational unit : Email Department
Your common name for the Certificate Authority is : Certificate Authority
Is this OK?
Enter Y[es] or n[o] : y
Enter the common name for this certificate, should be the
name in the MX record for the domain like {mail,smtp}.example.com :
smtp.example.com
Enter the organizational name for this certificate,
something like IMAP-server or Email Department : Email Department
Enter the common name for this certificate, should be the name in
the DNS for access to the imap server (recommended: imap.example.com) :
imap.example.com
Enter the email address of the Certificate Authority (recommended
ca@example.com) : ca@example.com
Enter the email address for the certificate of the
email server (recommended postmaster@example.com) : postmaster@example.com
Currently the domains this server considers or should consider local domains
are:
rpi-host.example.com, example.com, localhost.example.com , smtp.example.com and
localhost
```

The next question is there because the original file /etc/sysconfig/postfix has an item to add additional domains which are considered local. This means that when you have MX records of another domain pointing to this server (smtp.example.com), these message are accepted and processed accordingly. The user part of such a destination address must be a local user or an entry in the /etc/aliases file in the current setup. Sending messages with a from address with this domain name via this postfix server will not be signed with the DKIM signature. See the chapter Manual enhancing the configuration of DKIM.

```
Do you want (more) additional domains?
Enter Y[es] or n[o] : n
Currently you do NOT want to check incoming connections on port 25 on SPF
information
Is this OK?
Enter Y[es] or n[o] : n
```

Answering no means that package postfix-policyd-spf-perl from a repository other than the standard repository needs to be installed. It is needed to trust this repository. After installation the repository will be disabled because it will otherwise interfere when you want to update the system with “zypper up”.

DKIM support by AMaVIS

```
Currently DKIM support has not been enabled
Do you want to enable DKIM support
Enter Y[es] or n[o] : y
Currently DKIM support is done using openDKIM?
```

```
Do you want to change that into DKIM support by AMaVis?
Enter Y[es] or n[o] : y
```

Answering two times yes, enables DKIM support which is build-in in AmaVis.
The alternative is DKIM support by the package opendkim. See section **DKIM support by opendkim**.
Note that you did not see a message about generating the DKIM private key and TXT record. This the files with this data was already present.

```
Currently you do NOT want to check incoming email on port 25 on DMARC
information
Is this OK?
Enter Y[es] or n[o] : n
```

The answer no means that packet opendmarc will be installed, again from another repository than the standard repositories of openSUSE. This repository is also disabled. The last output of the script is:

```
==========================
Configurating firewalld...
==========================
success
success
success
success
success
success
success
success
success
```

The output success is from the different commands to open the ports 25, 587 and 993 in firewalld.

To really configure postfix and other services the program /usr/sbin/config.postfix needs to be executed. It is a bash script with calls to perl scripts. The input are the files /etc/sysconfig/postfix, /etc/sysconfig/mail and /etc/sysconfig/amavis.

/usr/sbin/config.postfix

```
Reading /etc/sysconfig and updating the system...
Private RSA key successfully written to file
"/etc/dkim/example.com.dkim20230906.pem" (2048 bits, PEM format)
Created symlink /etc/systemd/system/multi-user.target.wants/amavis.service ->
/usr/lib/systemd/system/amavis.service.
Job for amavis.service failed because the control process exited with error
code.
See "systemctl status amavis.service" and "journalctl -xeu amavis.service" for
details.
creating CA request/certificate...
creating certificate request for postfix...
signing server certificate for postfix...
creating certificate request for dovecot...
signing server certificate for dovecot...
Setting up postfix local as MDA...
Setting up hard SPAM protection...
rebuilding /etc/postfix/sasl_passwd.lmdb
```

Sometimes, like above with amavis.service, a service does not start properly. This is due to timing issues between these services. A (re)start of the service will fix the issue. Maybe a few sleep commands between starts of these services will be needed.

DKIM support by opendkim

```
/usr/sbin/yast2-mail.sh
```

This a rerun of this script in which nothing has been changed until...

```
Currently DKIM support has been enabled
Is this OK? Otherwise it will be disabled.
Enter Y[es] or n[o] : y
Currently DKIM support is done using DKIM in AMaVis?
Do you want to change that into DKIM support by openDKIM?
Enter Y[es] or n[o] : y
```

After this you will see the installation of the package opendkim from the standard repositories of openSUSE.

```
Currently you want to check incoming email on port 25 on DMARC information
Is this OK?
Enter Y[es] or n[o] : y
==========================
Configurating firewalld...
==========================
success
```

After this you run:

```
/usr/sbin/config.postfix
Reading /etc/sysconfig and updating the system...
Created symlink /etc/systemd/system/multi-user.target.wants/opendkim.service -
> /usr/lib/systemd/system/opendkim.service.
Setting up postfix local as MDA...
Setting up hard SPAM protection...
```

Testing DKIM signing

You can test if DKIM signing works by creating the following script (obviously you need to replace some of these items by your data):

```
#!/usr/bash
domain="example.com"
authbase64=$(echo -en "\0john\0password" | base64)
date=$(date --date=now "+%d %b %Y %H:%M:%S")
openssl s_client -connect localhost:587 -starttls smtp -quiet <<EOF
EHLO smtp.$domain
AUTH PLAIN $authbase64
mail from:<root@$domain>
rcpt to:<root@localhost.$domain>
data
Date: $datum
From: root@$domain
To: root@localhost.$domain
Subject: test DKIM
MIME-Version: 1.0
Content-Type: text/plain; charset=us-ascii
```

```
Content-Transfer-Encoding: 7bit

Test DKIM
.
QUIT
EOF
```

You can also enter these lines manually where you substitute the above parameters yourself.
The result should be either in /var/spool/mail/john or in /home/john/Maildir/new/* and should contain an entry starting with:

```
DKIM-Signature: v=1; a=rsa-sha256; c=relaxed/simple; d=example.com;
```

and a few lines below:

```
:received:received; s=dkim...;
```

SPF TXT record explained

The script does **not** check on the presence of a SPF TXT record in the DNS of your domain.

In case you want to protect others from using email addresses of your domain, you are depending on mailers in the rest of the world. These mailers need to use SPF validation and you only need to have a SPF record in the DNS of your domain. Remember this check is done using the domain part of the address in the MAIL FROM and the IP-address that makes the connection. Such a record is a TXT record which contains the following:

```
@ IN TXT "v=spf1 mx a ip4:192.0.7.25 ip6:2001:1234:4567::32 a:example.com ~all"
```

- @ means it is a record in your top domain: example.com
- v=spf1 indicates the TXT record is a SPF record
- mx means that the IP address associated with your MX server is allowed to send email with your domain addresses
- a means that IP-addresses in your A and AAAA record are allowed to send email with your email addresses
- ip4:192.0.7.25 means that a system with this IP address is allowed ...
- ipv6:2001:1234:4567::32 means that this IPv6 address is allowed ...
- a:example.com means that IP-addresses mentioned in A and AAAA records of example.com are allowed ...
- ~all should always be the last element in your record, where ~ means: when there is no IP address allowed it is a soft failure, which adds up to the spam count. When this character is -, it means deny access.

When MAIL FROM contains an empty address, in case of a.o. a non-delivery message, there is no domain to check for an SPF record. In that case the receiving server will use the name in the HELO/EHLO command. So if your server uses smtp.example.com as the name in this command, you should also have the following record in your DNS:

```
smtp IN TXT "v=spf1 a -all"
```

Obviously you need an A and/or AAAA record for smtp.example.com and any server trying to mimic your server using this name in the HELO/EHLO command will be denied access.

More information about SPF records can be found in RFC7208.

DKIM explained

DKIM, short for DomainKeys Identified Mail, gives the body and header of an outgoing email a digital signature. The public key is published via DNS, so a receiving server can verify the digital signature of incoming messages.

The package amavisd-new contains the necessary support for both validating incoming messages and providing the signatures for header and body of outgoing messages. There is also the package *opendkim*, which is meant for large servers, which is also supported in this script.

Note that there might not be a relation between the domain of the sender in a message and the domain which provides the signature. It is up to the receiving server what to do with the result of a positive result. The hope is that this server is trusting the message more than non-signed messages.

The private key used for the signature is in the folder `/etc/dkim/` and so is the public key to be entered in the DNS.

Maintenance of DKIM keys

It is good practice to renew DKIM keys occasionally. Depends on how many times you did send a signed message. The best way is to use a new signing key like the one in the above mentioned folder. The name can be any name that can be used in the DNS. After having implemented a new signing key (see below) the previous one can be removed from the DNS after the ttl time (10 days in the example).

To generate a new key the following script can be used:

for DKIM support in AmaVis

The script is:

```
#!/bin/bash
date=$(date --date=now +%Y%m%d)
amavisd -c /etc/amavisd.conf genrsa /etc/dkim/example.com.dkim${date}.pem 2048
```

After this you need to change the reference to the original file in a reference to this new file at the end of `/etc/amavis.conf` . After that you have to generate the public key:

```
amavisd -c /etc/amavisd.conf showkeys > /etc/dkim/domain.com.dkim${date}.txtrecord
```

for DKIM support in opendkim

The script is:

```
#!/bin/bash
cd /etc/dkim
date=$(date --date=now +%Y%m%d)
opendkim-genkey -b 2048 -h sha256 -s dkim$date -d example.com
chmod 640 dkim$date.private
mv dkim$date.private example.com.dkim$date.pem
```

```
mv dkim$date.txt example.com.dkim$date.txtrecord
```

In /etc/opendkim/opendkim.conf you have to change the filename in the line starting with: KeyFile

for both

Use the file .txtrecord to add the new TXT record in the DNS. You should leave the previous one, at least foor 10 days in the DNS. This allows for delays of 10 days in delivery of your message on the receiving server.

DMARC explained

DMARC, short for Domain-based Message Authentication, Reporting and Conformance, is an addition to the other 2 security standards for email, SPF and DKIM. DMARC gives receiving servers an indication on how to process incoming email messages that do not have signatures that provide valid SPF and DKIM results. These can be discarded or quarantined.

DMARC can check if the sender domain in the ‘From’ header matches with the sender domain in the envelope (SPF) and with the signing domain in the DKIM header. This means that DMARC forces a relation between the validated SPF and DKIM sender domains and the sender domain in the 'From'. This means that a message that gives a positive result on the validation of SPF and DKIM, still can be rejected by DMARC.

The DMARC policy is published via a record in the DNS. This record can also contain email addresses which can be used by mail systems to report accepted/refused messages. This way the manager of the mail domain will get some insight in the delivery of both real as falsified messages.

The DMARC TXT record in the DNS

To enable DMARC you need a DMARC TXT record in your DNS. An example of such a record is:

```
_dmarc    IN    TXT    "v=DMARC1; p=none; sp=none; adkim=s; aspf=s;
rua=mailto:dmarc-reports@example.com; ruf=mailto:dmarc-reports@example.com
fo=1;"
```

The meaning of the elements in this record is as follows:

- v=DMARC1 indicates together with the name of the record _dmarc that the TXT record is a DMARC record
- p=none indicates the policy for the domain
- sp=none indicates the policy for subdomains, when these subdomains do not have a DMARC TXT record
- adkim=s indicated the strength with which the receiving server should test the alignment between the domain name in the FROM address and the DKIM key inserted in the message
- aspf=s indicates the strength of with which the receiving server should test the alignment between the sending IP address and the allowed IP addresses in the SPF record
- rua=mailto:dmarc-reports@example.com indicates the email address where aggregated reports should be send to, by default once a day (another parameter, ri, may change this value, parameter pct, default 100, indicates that only a percentage of the reports need to be send)

- ruf=mailto:dmarc-reports@example.com indicates the email address where a reports should be send to in case the server does not accept the message
- fo=1 indicates that reports about not accepted messages should be send (see rfc7489 section 6.3 for additional values)

The possible policies are *none*, *quarantine* and *reject*. One should start with *none* which means that tests at the receiving end should be performed and reports should be send, but the message should be accepted. It is meant for situations where there are several systems being able to send message with your domain name in the FROM address and you expect that your SPF record may not be correct or not all sending servers have implemented the DKIM signing.

The strength in testing in *aspf* and *adkim* can be *s* for strict and *r* for relaxed. With adkim=s the domain part of the from address in the header **must** match the domain name in the DKIM signature. For adkim=r the domain part of from address in the header is allowed to be a subdomain. For aspf=s the domain part of from address **must** exactly match the domain part in the smtp protocol command MAIL FROM, also called the envelop sender. For aspf=r this match is allowed to be partly.

Script to test the generation of a DKIM element in the header

The file `testdkim.sh` can be used to test the setup of your server. It asks for the domain name of your server, the user name of a user on your system and its password. After that a very small email message will be send via the port submission (587). This should enable the insertion of the DKIM item in the header of the message. The message is send to root, which means that it will be redirected to the user where messages for root are directed. In the folder `~/Maildir/new/` of that user you will find the message and you can inspect it for the presence of the DKIM item in the header.

Manual enhancing the configuration of DKIM

The writing of this chapter is pending and you are invited to cooperate.