DKIM Signing with Kolab 16 on Debian 9

Other Configurations

From Milan Petrovic:

> Worked like a charm! Thanks!
> Confirming it works also on **Winterfell** installed on **CentOS 7**.
> The only difference is apt install… → „yum install opendkim“ (tools are included).

Preface

I could not find any instruction for this that really worked.

For example there is an article using Amavis in the Kolab Enterprise KB. It seems to be written for CentOS or Red Hat, but it was adaptable for Debian mostly by replacing `amavisd -c /etc/amavisd/amavisd.conf` by the `amavis-new` command. Unfortunately some important point must be missing, because whatever I tried I could not get it to deliver a *valid* signing.

A post in the Kolab Users Mailing List describes using OpenDKIM for signing and claims it works for Roundcube, but not for Outlook (I dont use Outlook so that did not bother me). Unfortunately this solution ended up with my messages being signed 3 times. But this is still basically the approach I used, I just had to get rid of the unwanted additional signings.

So after figuring it out I decided to share my solution. I wanted to contribute this howto directly to the Kolab Documentation Project but I did not succeed with the contribution process. Maybe someone else who is familiar with that process can copy this to the Kolab Documentation later.

So far I tested these clients and they deliver valid DKIM signing:

- Roundcube of Kolab 16.
- Windows 10 mail app with Exchange ActivSync account
  (Windows 10 calendar and contact apps also working well with Kolab over ActiveSync btw).
- Thunderbird 68.0 on Windows 10 (all settings default).
- Android («Exchange» Account, including Contacts and Calendar)

Enabling DKIM

All following commands shoud be run as root user.

Replace the domain name, ip, ports etc with your own data.

Install opendkim

```
apt install opendkim opendkim-tools
```

Create configuration dirs

```
mkdir -p /etc/opendkim/keys
```

Save the original configuration file for reference

```
cp /etc/opendkim.conf /etc/opendkim.conf.orig
```

Edit the configuration file /etc/opendkim.conf

Here are all my settings in alphabetical order. Edit your config file to adapt them, or just clear the old content and paste all of this into the file:

opendkim.conf

```
AutoRestart             yes
AutoRestartRate         10/1h
Canonicalization        relaxed/simple
ExternalIgnoreList      refile:/etc/opendkim/TrustedHosts
InternalHosts           refile:/etc/opendkim/TrustedHosts
KeyTable                /etc/opendkim/KeyTable
LogWhy                  yes
Mode                    sv
OversignHeaders         From
PidFile                 /var/run/opendkim/opendkim.pid
SignatureAlgorithm      rsa-sha256
SigningTable            refile:/etc/opendkim/SigningTable
Socket                  inet:8891@localhost
Syslog                  yes
SyslogSuccess           yes
Umask                   002
UserID                  opendkim:opendkim
```

Create /etc/opendkim/TrustedHosts

TrustedHosts

```
127.0.0.1
::1
localhost
myhost
mydomain.com
```

```
myhost.mydomain.com
mail.mydomain.com
smtp.mydomain.com
```

Create /etc/opendkim/SigningTable

SigningTable

```
*@mydomain.com mydkim
```

Create /etc/opendkim/KeyTable

KeyTable

```
mydkim mydomain.com:mydkim:/etc/opendkim/keys/mydkim.private
```

Create the key and set the proper owner and permissions

```
cd /etc/opendkim/keys
opendkim-genkey -d mydomain.com -b 4096 -r -s mydkim
chown -R opendkim:opendkim /etc/opendkim
chmod -R go-rwx /etc/opendkim/keys
```

Display the public key

```
cat mydkim.txt
```

this will show something like

```
mydkim._domainkey IN      TXT     ( "v=DKIM1; h=sha256; k=rsa; s=email; "
  "p=MIICIjANBgkqhk ...
          .
          .
          .
  ... sCAwEAAQ==" )  ; ----- DKIM key mydkim for mydomain.com
```

Copy/paste the public key into your DNS record

The way you have to apply this varies depending on your provider. In my case I could just copy it (without the comment starting as ; —) into my server manager (Hetzner Robot).

The propagation of the DNS record may take quite a while, depending on the TTL it had before changing. In my case with a TTL of 300 it took some 15 minutes to fully propagate, but it could take days if you have a high TTL.

You can check if propagation has happened with one of these commands:

```
host -t txt mydkim._domainkey.mydomain.com
```

or

```
dig -t TXT mydkim._domainkey.mydomain.com
```

Once you have the new DNS record propagated you are ready to continue with the next step.

Check with OpenDKIM

Restart opendkim:

```
service opendkim restart
```

Then check with:

```
opendkim-testkey -d mydomain.com -s mydkim -vvv
```

The output should look like:

```
opendkim-testkey: using default configfile /etc/opendkim.conf
opendkim-testkey: checking key 'mydkim._domainkey.mydomain.com'
opendkim-testkey: key not secure
opendkim-testkey: key OK
```

The note `key not secure` is normal if not using DNSSEC, so don't worry.

The important part is `key OK` in the last line.

Integrate in Postfix as Milter

Edit `/etc/postfix/main.cf` and add the following lines at the very end:

```
#dkim
milter_protocol = 6
smtpd_milters = inet:127.0.0.1:8891
non_smtpd_milters = $smtpd_milters
milter_default_action = accept
```

Avoid multiple signings

Since Kolab defines additional smtpd commands in the mail chain, the message would also get signed multiple times by the milter.

You can stop that by adding `-o receive_override_options=no_milters` to all smtpd commands except the first.

Edit `/etc/postfix/master.cf` and append the option to the respective services.

In my case I had to add it to the `submission` service and the `127.0.0.1:10025` listener:

```
submission          inet        n       -       n       -       -
smtpd
    -o cleanup_service_name=cleanup_submission
                        .
                        .
                        .
    -o receive_override_options=no_milters
```

```
# Listener to re-inject email from Amavisd into Postfix
127.0.0.1:10025     inet        n       -       n       -       100
smtpd
    -o cleanup_service_name=cleanup_internal
                        .
                        .
                        .
    -o receive_override_options=no_milters
```

Now restart postfix and basically you are done:

```
service postfix restart
```

Testing

An easy way to test is send a mail from your Kolab Roundcube to another mail account where DKIM check is activated.

When looking at the received header in the other mail account, you should see something like this:

```
Return-Path:            <myname.myfam@mydomain.com>
X-Spam-Checker-Version: SpamAssassin 3.4.0 (2014-02-07) on
otherhost.otherdomain.com
X-Spam-Level:
X-Spam-Status:          No, score=-0.1 required=4.0
tests=DKIM_SIGNED,DKIM_VALID,DKIM_VALID_AU,HTML_MESSAGE,...
X-Original-To:          othername@otherdomain.com
Delivered-To:           othername.otherfam-
otherdomain.com@otherhost.otherdomain.com
```

```
Received:               from myhost.mydomain.com by
otherhost.otherdomain.com (Postfix) with ....
DKIM-Filter:            OpenDKIM Filter v2.11.0 otherhost.otherdomain.com
88A2F3DE75
Authentication-Results: otherhost.otherdomain.com; dkim=pass (4096-bit key)
header.d=mydomain.com header.i=...
MIME-Version:           1.0
DKIM-Signature:         v=1; a=rsa-sha256; c=relaxed/simple; d=mydomain.com;
s=mydkim; t=1567929356; bh=... ...=
To:                     otherfam othername <othername@otherdomain.com>
From:                   "myfam, myname" <myname.myfam@mydomain.com>
Subject:                DKIM Test
Date:                   Sun, 8 Sep 2019 09:55:56 +0200
Importance:             normal
X-Priority:             3
Content-Type:           multipart/alternative;
boundary="_C801F48A-89D1-4981-89A1-55A3A096334D_"
Message-ID:             <3e79ae1f5936c9c7eb2e987b9b569354@mydomain.com>
```

In `Authentication-Results` you can see that the DKIM test passed, and also in `X-Spam-Status` you can see that DKIM ist valid. Your headers my vary from this example, but if the receiver does DKIM checking you should find such informations.

Another convenient option is to use one of the many free mail check services, for example https://www.appmaildev.com/.

Possible fails

If your signing does not pass, most likely the DNS has not yet fully propagated in the web. You might be seeing it correct from your server and client, but it has not yet arrived in the area of the receiving machine. Just give it some time and then test again. If you have a long TTL in your DNS, you may want to disable signing for the long waiting time, by removing the lines added to `/etc/postfix/main.cf` and restarting postfix.

Another reason might be that the DKIM key in the DNS record does not match the key on the server, double check that it is absolutely correct.

DKIM verification

DKIM verification is included automatically when using OpenDKIM. You may check the header of a DKIM signed mail received in your Kolab Roundcube to find a entry as:

```
Authentication-Results: myhost.mydomain.com;
    dkim=pass (4096-bit key; unprotected) header.d=otherdomain.com
header.i=@otherdomain.com header.b="Xj5q83C9";
    dkim-atps=neutral
```

Again `4096-bit key; unprotected` does not mean something is wrong, it just says that DNSSEC is not enabled for the other host that you received the mail from.

Amavis also has a DKIM verification that you can enable. It is easy to activate, just add this line to `/etc/amavis/conf.d/50-user`:

```
$enable_dkim_verification = 1;
```

The whole file will look as this if nothing else was added before:

50-user

```
use strict;

#
# Place your configuration directives here.  They will override those
in
# earlier files.
#
# See /usr/share/doc/amavisd-new/ for documentation and examples of
# the directives you can use in this file
#

$enable_dkim_verification = 1;

#------------ Do not modify anything below this line -------------
1;  # ensure a defined return
```

Then restart amavis:

```
service amavis restart
```

What else

Reverse DNS (PTR Resource Records)

It is mandatory to have correct Reverse DNS set up for the servers sending mails if you don't want your mails to get classified as spam. Check in the server configuration tool of your provider if you can do that yourself. Ask your provider to set it up otherwise (for both ip4 and ip6 in case).

SPF and DMARC

Once you have DKIM signing and verification up, you might want to also add SPF and DMARC to complete things. Thankfully that is quite easy since you don't need to change anything within your mailserver, but just add the proper lines to your DNS. There are other howtos and there are also services available to help compose those entries. Nothing special for Kolab here.

DNSSEC

DNSSEC is very slowly spreading, and for user domains I'm not sure it will get wide spread before something easier to handle comes up. First the hosting providers would need to catch up and give you the tools to manage it (zone keys etc). But even my provider (Hetzner) which is among the biggest and most reputable providers in Europe has for the time being abandoned introduction because it is so complex to implement and there is little demand. Conclusion: Nice to have, but don't worry if not having it.