Postfix with Radius Authentication on FreeBSD

Toni Schmidbauer

April 16, 2005

Abstract

This document describes how to configure postfix with radius authentication on FreeBSD. Usernames/passwords are stored on a freeradius server and postfix uses radius to authenticate users for smtp relaying (smtp-auth). Installation of freeradius is not covered in this document, so as a requirement you need a working freeradius installation.

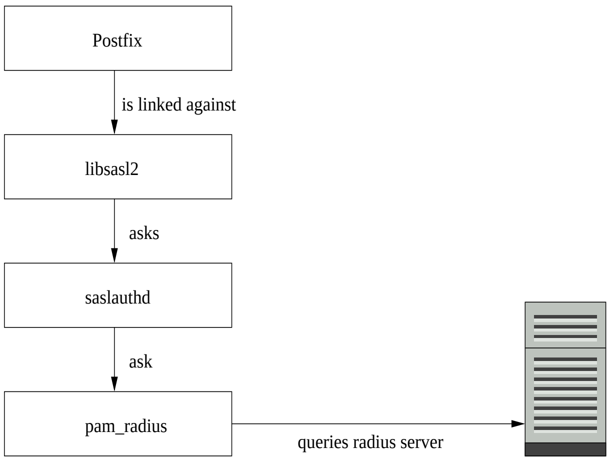
Contents

1 Why SMTP authentication

Normally relaying mails via postfix is only allowed to local users, for hosts specified with `$mynetworks`, virtual domains and domains in the transport table. Sometimes it's necessary to allow relaying for users if you are e.g. an isp. You could add the ip addresses or network ranges, but thats a bad idea IMHO. So the solution for us was to add smtp-auth to postfix. now only authenticated users (username/password) can relay mails via our postfix mailserver.

It should be noted that this adds no security to mail relaying. Usernames and password are transmitted plain text without encryption. If you want to add security use smtp of ssl (starttls).

2 Big Picture



Postfix is linked against libsasl2 and asks the sasl library for authentication. libsasl2 queries the saslauthdaemon via an unix domain socket. saslauthdaemon itself requests pam_radius if the supplied username/password is valid. finally this request is forwarded to the radius server which responds with an access permitted or denied message.

3 Installation

First update your ports tree to be sure to get the latest version of postfix and sasl (see http://www.freebsd.org/doc/en_US.ISO8859-1/books/handbook/ports-using.html). Then proceed with the following steps:

- `su - root`
- `cd /usr/ports/mail/postfix`
- `make install clean`
- Choose: [X] Cyrus SASLv2 (Simple Authentication and Security Layer)
- OK
- next is saslauthd so `cd /usr/ports/security/cyrus-sasl2-saslauthd && make install clean`

Thats it, all required ports are now installed.

4 Configuration

4.1 Freeradius

First you have to tell freeradius where the authentication requests will come from, so in our case the postfix server. Locate your `$FREERADIUS_INSTALLDIR/etc/raddb/clients.c` file and add the following entry:

```
client 10.0.0.1 {
        secret          = yourstrongsecrethere
        shortname       = postfix
}
```

Where freeradius stores usernames and password depends, but in our case its `$FREERADIUS_INSTALLDIR/etc/raddb/users`:

```
pinhead@stderror.at Auth-Type := Local, User-Password == "password"
```

Finally restart freeradius.

4.2 Sasl

Create `/usr/local/lib/sasl2/smtpd.conf`:

```
pwcheck_method: saslauthd
mech_list: plain login
```

That's it, libsasl is configured, easy isn't it?

4.3 Saslauthd

Add the following entries to `/etc/rc.conf`:

```
saslauthd_enable="YES"
saslauthd_flags="-a pam -r"
```

The "-r" flag is required because our radius users contain the realm in their username (the part after the @ so @stderror.at). normally saslauthd doesn't include the realm when authenticating users and the radius server will reject the request. "-r" makes sasl and freeradius happy.

4.4 pam_radius

Two files have to be created for `pam_radius` to work:

`/etc/radius.conf`:

```
auth 10.0.0.2 yourstrongsecrethere
```

`10.0.0.2` is the ip address of our freeradius server `yourstrongsecrethere` is the password for authentication to the radius server (see 4.1).

`/etc/pam.d/smtp`:

```
auth            required        pam_radius.so

account         required        pam_permit.so
```

this is the configuration for pam. we only require a successful radius login, no accounting will be done.

4.5 Postfix

Now that we configured sasl, saslauthd and pam we can glue everything together via postfix.

Add the following entries to `/usr/local/etc/postfix/main.cf`:

```
smtpd_sasl_auth_enable = yes
smtpd_sasl_security_options = noanonymous
smtpd_sasl_local_domain = $myhostname
broken_sasl_auth_clients = yes

smtpd_recipient_restrictions =
  permit_sasl_authenticated,
  permit_mynetworks,
  reject_unauth_destination,
  reject_non_fqdn_recipient,
  reject_unknown_recipient_domain,
  reject_unauth_destination
```

Important are the smptd_sasl options and `permit_sasl_authenticated`.

5 Testing

5.1 Saslauthd

5.2 Postfix

First thing to check is if postfix announces the smtp auth service. So we simply telnet to our postfix server and ask him:

```
$ telnet localhost 25
220 postfix.lan.at ESMTP Postfix
ehlo postfix
250-postfix.lan.at
250-PIPELINING
250-SIZE 10240000
250-VRFY
250-ETRN
250-AUTH LOGIN PLAIN
250-AUTH=LOGIN PLAIN
250 8BITMIME
```

as we can see `AUTH LOGIN PLAIN`, postfix is ready to authenticate us. Next we try to use a fake authentication string:

```
$ telnet localhost 25
220 postfix.lan.at ESMTP Postfix
ehlo postfix
250-postfix.lan.at
250-PIPELINING
250-SIZE 10240000
250-VRFY
250-ETRN
250-AUTH LOGIN PLAIN
250-AUTH=LOGIN PLAIN
250 8BITMIME
auth plain itsme
535 Error: authentication failed
```

Ok, this seems to work to. Now we use an existing username/password. postfix expects this to be base64 encoded, so this little perl magic will help:

```
$ perl -MMIME::Base64 -e 'print encode_base64("pinhead\@stderror.at\0pinhead\@stder
cGluaGVhZEBzdGRlcnJvci5hdABwaW5oZWFkQHN0ZGVycm9yLmF0AHBhc3N3b3Jk
```

and in a telnet session:

```
$ telnet localhost 25
220 postfix.lan.at ESMTP Postfix
ehlo postfix
250-postfix.lan.at
250-PIPELINING
250-SIZE 10240000
250-VRFY
250-ETRN
250-AUTH LOGIN PLAIN
250-AUTH=LOGIN PLAIN
250 8BITMIME
auth plain cGluaGVhZEBzdGRlcnJvci5hdABwaW5oZWFkQHN0ZGVycm9yLmF0AHBhc3N3b3Jk
235 Authentication successful
```

Seems to work to.

5.3 Network traffic

With the help of tcpdump we can watch the radius packets on the wire:

```
$ tcpdump -vvv -s 1500 -n -i fxp0 host 10.0.0.2
tcpdump: listening on fxp0, link-type EN10MB (Ethernet), capture size 1500 bytes
15:08:22.327087 IP (tos 0x0, ttl  64, id 15979, offset 0, flags [none], length: 107)
        Access Request (1), id: 0x79, Authenticator: 3bc8aab347f9603aeb42d85533f3a10
          Username Attribute (1), length: 13, Value: pinhead@stderror.at
            0x0000:  6969 6465 4063 6f69 2e61 74
          Password Attribute (2), length: 18, Value:
            0x0000:  e049 b950 ca84 11d8 5d78 f2f0 7dc3 bd24
          NAS ID Attribute (32), length: 22, Value: prosl.it-austria.net
            0x0000:  7072 6f73 6c2e 6974 2d61 7573 7472 6961
            0x0010:  2e6e 6574
          Service Type Attribute (6), length: 6, Value: Authenticate Only
            0x0000:  0000 0008
15:08:22.327769 IP (tos 0x0, ttl  64, id 3, offset 0, flags [DF], length: 78) 10.0.0
        Access Accept (2), id: 0x79, Authenticator: fb8b7fd5901deceb8d7588c6c3c5b69c
          Service Type Attribute (6), length: 6, Value: Framed
            0x0000:  0000 0002
          Session Timeout Attribute (27), length: 6, Value: 06:00:00 hours
            0x0000:  0000 5460
          Framed Protocol Attribute (7), length: 6, Value: PPP
            0x0000:  0000 0001
          Framed IP Address Attribute (8), length: 6, Value: 192.168.1.1
            0x0000:  a219 f045
          Framed IP Network Attribute (9), length: 6, Value: 255.255.255.255
            0x0000:  ffff ffff
```