Postfix Catch All and Mutt

End goal: having postfix saving all of the emails for a domain to a single “mailbox” in maildir format, and being able to send email using mutt (or similar).

Specifics to my setup

I'm not going to open port 25 on my server, I'm going through Net7's spam filter, which will then forward to my server on port 8025.

So I need to open port 8025.

The rule was already there for port 25, I just need to edit the thing.

Make Postfix listen to another port

There is always the plan of using iptables to redirect the traffic.

You can do it in Postfix as well. Open `master.cf` and find this line:

```
smtp      inet  n       -       -       -       -       smtpd
```

The `smtp` word up front is actually a port. You can replace it with this line:

```
8025      inet  n       -       -       -       -       smtpd
```

If you restart Postfix and check with netstat it should be listening to another port.

Setting up Maildir delivery

By default Postfix will output emails to a single file in /var/mail/. I'd rather have the Maildir format which separates emails into different files that I can individually move and/or delete.

I'm going off the rails here trying out things.

Looks like we're going to need these configuration options in `main.cf`:

```
home_mailbox = Maildir/
mailbox_command =
```

Make sure mailbox_command isn't set somewhere else in the file.

Reload Postfix. We should be able to test that this is working using a known user on the system. You can telnet-test like so:

```
EHLO test
MAIL FROM:<bouc@clanz.be>
RCPT TO:<william>
DATA
Test.
.
```

Where `william` is an actual Linux user on that machine. Notice that you have

to write it with no domain.

In my case this did create a /home/william/Maildir directory.

You should probably create a specific account for your email catch-all thing. Be careful that if you want to use postmaster you have to remove it from /etc/aliases (and run newaliases afterwards (I think)).

Domain configuration

Next you need to edit or create `/etc/postfix/virtual`.

Add this in it:

```
@dkvz.eu mailuser
```

Where mailuser is the system user you want to use for emails. I suppose it could also be an alias present in /etc/aliases.

Save the /etc/postfix/virtual file and postmap it:

```
postmap /etc/postfix/virtual
```

We now need to open /etc/postfix/mail.cf and add this line:

```
virtual_alias_maps = hash:/etc/postfix/virtual
```

Then reload Postfix.

Relay access config

There's a good chance you'll still get a relay access denied trying to send to the catch all domain you have configured.

This is due to this line:

```
smtpd_relay_restrictions = permit_mynetworks permit_sasl_authenticated
defer_unauth_destination
```

This soup is kind of hard to configure. You can look at the doc here: http://www.postfix.org/postconf.5.html#smtpd_relay_restrictions

For this to work as it is with the configuration line written above, you need to add your domain to the `mydestination` option (comma separated).

Specifics to my situation

I actually use an intermediate relay that filters spam. In that situation, I can use mynetworks to allow the specific IP addresses of these relays and arrange `smtpd_relay_restrictions` to only allow relay to mynetworks.

Another plan is to do exactly that but through iptables: only allow your intermediate servers to reach your mail port. I find this to be easier and safer anyway.

Rules like this one should probably do it:

```
iptables -A INPUT -p tcp -m tcp -s <SOURCE_IP_ADDRESS> --dport 8025 -m state
--state NEW,ESTABLISHED -j ACCEPT
```

```
```

Where <SOURCE_IP_ADDRESS> is the ip address you want to allow.

This works for me because these are the first rules I set for my firewalling:

```
iptables -P INPUT DROP
iptables -A INPUT -i lo -j ACCEPT
iptables -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
iptables -P OUTPUT ACCEPT
```

Which is basically drop all incoming except if on loopback or already established, also let all output connections go with no restrictions.

It's important to check that you still have local access to your SMTP or it will be... Hard to use the command line email client later on.

Reading your email

Provided your DNS is setup correctly you can now receive emails to your domain and it will pop in your Maildir directory inside /home/<YOUR_USER>.

You can read it like a warrior by checkout the files in directories cur and new.

Or you can install a command line client. The most common command line clients are:

- Mutt
- Alpine

I'm going to be using Mutt:

```
apt-get install mutt
```

Mutt can then open a maildir like so:

```
mutt -f /home/<YOUR_USER>/Maildir
```

It should be able to send mail on its own, but we should probably set some preferences first as fiddling with mutt like this with no settings will create some files for its internal use that we won't need later on (or will need to be somewhere else).

This is where the “fun” begins. Debian has default mutt settings in `/etc/Muttrc`. After loading the prefs in there, Mutt is supposed to load the prefs found in the file:

```
/home/<CURRENT_USER>/.mutt/muttrc
```

With <CURRENT_USER> being the user you're logged in as, not necessarily the Maildir real owner. You could read your emails as root but I think it's probably better to su to your email user.

There are mentions that mutt used to not be able to send email out of the box. In my case it works fine with the default configuration and does send through Postfix.

Mutt relies on your currently set default text editor if you're using Debian. For

some reason it's installing joe and setting it as default. I honestly find this a little weird.

To reset your editor, you need to use update-alternatives:

```
update-alternatives --config editor
```

About default settings

It looks like your last sent email is saved in a Sent folder created in your user home directory. However, there won't be a Trash folder. Deleting emails will purge them into the void forever. Which might be nice.

Minimal configuration file

If having date issues you can use this:

```
set locale="fr_BE"
```

I don't.

So my config file goes like this:

```
set mbox_type = Maildir
set folder = ~/Maildir/
set spoolfile = +/
set realname = "DkVZ"
set from = "dkvz@dkvz.eu"
set use_envelope_from = yes
set edit_headers = yes                                  # Will allow us to
change the from address from Mutt
set record = +/sent/
set sort = threads
auto_view text/html                                     # view html
automatically
alternative_order text/plain text/enriched text/html    # save html for last
```

I'm not setting my editor in the config file as I'm using the default system editor.

You could change the order of what is displayed in terms of alternatives. I prefer to show text/plain first, if a text/plain alternative is even present. If not, this is where auto_view comes into play, but it doesn't work as is.

Mutt will need to pass that specific content type to a third party program of your choice. Some people use elinks or lynx (command line browsers) or w3m.

First you need to install w3m:

```
apt-get install w3m
```

Next create a mailcap file in the .mutt folder that is in your user directory, put this in:

```
text/html; w3m -I %{charset} -T text/html; copiousoutput;
```

Mutt will now display emails in HTML only using w3m. Alternatively when viewing a multipart/alternative email, you can hit “v” and select the HTML part

(if any), it will open it in w3m. This works pretty well for displaying tables, for instance.

This config file makes it so that Mutt will use Maildir as default mailbox format. The default is to use a single file, which is hard to work with in my opinion.

We're saving sent items in a sent directory that will be inside Maildir (and created automatically).

With such a configuration you can just run:

```
mutt
```

As this specific user and it will load your mailbox.

Charset problems

I'm not having any charset problems so I didn't not fiddle with this.

Just using `set locale =` wasn't working for me though.

I've seen mentions of this:

```
set charset="utf-8"
set locale="fr_FR.UTF-8"
```

For french. Do note that fr_FR.UTF-8 must exist in your installed locales. You can show the locales using:

```
locale --all-locales
```

And I think that to install more you need this on Debian systems:

```
dpkg-reconfigure locale
```

Again, I haven't tested this.

Mutt saving stuff in /tmp

I'm not sure what's going on there. This might be a problem with my version of mutt (in Debian stable so quite old). Just clean /tmp up from time to time.

I use this script:

```
#!/bin/sh

rm /tmp/mutt-*
```

Advanced stuff

Mutt can do much more, such as reading from an IMAP or POP mailbox. It can also display common senders using aliases, all the colors can be customized, viewers can be set up for different content types. There's a lot that can be done, but that's out of scope of what I wanted to do myself.

Forward & Resend

Forwarding is not super intuitive in Mutt. Hitting f will never forward attachments. For that you have to “re-send”, which is Esc + E.

Edit the headers accordingly (Subject and To).

Refreshing

By default mutt only checks for new email when you press a key in its interface. If you leave mutt open and receive emails while not interacting with it, it won't show the new email. This is find by me but if you want to auto-refresh you will want to look at a configuration setting called `timeout`.

Sorting

I don't understand how sorting works. It's explained in the manual I guess. It's weirdly very complicated to just try to have the emails get sorted by latest received first. So I'll just keep it that way. I'm lazy like any sane person you know.

Handling quotas and shit

I don't want someone mailbombing me to saturate my server.

There is a software called `quota` somehwere. The other way would be to add a separate partition, possibly using a file created with dd if you don't have means to create more partitions.

Using quota

There actually is a package called `quota` that you need to install:

```
apt-get install quota
```

You then have to edit /etc/fstab. Let me say beforehand that it would be ideal to have a separate /home partition and do all these operations on that. In my case I have a single volume, so I'm putting quotas on the root partition.

This procedure applies to “journaled” quota, which is kind of old but kind of never mentioned either.

You need to add these options to the partition you want to apply quotas on:

```
usrjquota=aquota.user,grpjquota=aquota.group,jqfmt=vfsv0
```

Reload the partition (in my case it's “/” as mentioned before):

```
mount -o remount /
```

Next we need to create the quota usage database thingy:

```
quotacheck -avugm
```

This will create two files in / (or in /home if you used it there).

Now we need to enable quotas:

```
quotaon -avug
```

Now you can create quotas for your users:

```
edquota <USERNAME>
```

Where <USERNAME> is the user you want to apply a quota to.

This will open the default editor into a file with columns. If you want something quick and simple, only look at the hard column. This is the actual limit before the user can't write to disk anymore. The fun part is that this is not in bytes or related but in **blocks**. You need a Gigabyte to blocks calculator, there is one here: http://www.unitconversion.org/data-storage/gigabytes-to-blocks-conversion.html

The “blocks” column (the first one) shows the current usage.

With this setup the quotas will get disabled at the next reboot (and your per-user hard and soft limits will be lost as well). To have quotas being applied at boot you need to make it so that the script in /etc/init.d/quota gets executed in the boot sequence. How to do this is different whether you have systemd or not.

The latest Debian is currently using systemd, so here's how to enable quota at startup:

```
systemctl enable quota.service
```

When quotas are active, if the user to which Maildir Postfix needs to write to has exceeded their quota, the delivery will fail and Postfix will inform the sender of the failure, and add the reason that quota has been exceeded:

```
May 31 16:22:43 deb8 postfix/local[1885]: 06CF120055C: to=<joe@deb8>,
relay=local, delay=9, delays=8.9/0/0/0.1, dsn=5.2.2, status=bounced (maildir
delivery failed: error writing message: Disk quota exceeded)
```

As root you can use:

```
repquota -a
```

If you want a report with current usage and limits set to all the system users.

Postfix mailbox size limit

Your Postfix main.cf config file should come out of the box with this option:

```
mailbox_size_limit = 0
```

Well this thing doesn't do anything interesting, it doesn't really (as of this day) enforce a set limit for a Maildir, it instead prevents the local delivery part of Postfix to write any file bigger than this limit (in bytes). This is completely useless for what we're trying to do, but may be interesting to set if you have a `message_size_limit` that is huge (or unlimited) because you need to send huge files, but you do not want to receive huge files.

In my case I'm leaving message_size_limit to its default and mailbox_size_limit

to unlimited (thus 0).

Using "policy" third party software

Postfix can use external daemons to check stuff before delivery, including mailbox size. This is beyond the scope of what I wanted to do, which is going to simple route with no additional processes in memory.

Use a separate partition

I haven't tested what happens if you just use up all the space on the partition on which the mailbox is in. I would assume Postfix sends back a non-delivery report.

There's more than one way to do this, the best would be to really have a separate partition somewhere for the mailbox. You could also use LVM to create a logical volume for it (which can easily be extended later on, also, snapshots).

Now you can also create a “disk” with dd with a specific size that you can then mount. This method is flexible both for creation and mounting but as far as I know it requires provisioning the entire space reserved for it (a separate partition does that too anyway).

Apparently it may be possible to create thin provisioned disk files with dd using such a command:

```
dd if=/dev/zero of=filesystem.img  bs=1k seek=1024M count=1
```

More on that here: https://wiki.archlinux.org/index.php/Sparse_file

Anyway this is still half a solution as these sparse file never reclaim the previously used space, even if you delete a lot of files on it. So it can only end up using up everything anyway if you let it.

Avoiding being marked as spam

Gmail will always default to mark you as spam when you start sending from your personnal server.

There are several things you can try to avoid that.

Actually when I made my own thing I had the wrong SPF record so I'm not too surprised it was flagged.

SPF record

I recommend using a SPF generator like this one: http://www.spfwizard.net/

Then add the created TXT record. There are ways to easily allow your MX servers and the root domain A record to be able to send. This is actually enough for me, but I repeated the IP address from the said A record just in case.

As for the policy, the vast majority of people seem to be using SoftFail.

```
dkvz.eu.  IN TXT "v=spf1 mx a ip4:51.255.166.120 ~all"
```

This being the line as in a .dns file. If you need to add the TXT record the content is just `v=spf1 mx a ip4:51.255.166.120 ~all` without quotes.

You should then use a SPF checking tool to see if everything works. I find https://mxtoolbox.com/spf.aspx to be nice.

If you send to one of your Gmail addresses you can also go to Spam (if it's in there), click your email or conversation and use the menu attached to that email to *Show Original*. This will display all the headers, and Gmail keeps results of SPF checks in there.

Gmail tends to keep SPF records in cache for a long time. Not much to do about that besides being patient.

Just having correct SPF (with SoftFail, haven't tried hard SPF) won't prevent you to land in spam. I guess you need more.

DomainKeys

DomainKeys has been replaced by something called DKIM (DomainKeys Identified Email).

It's kind of like SPF except you have to create key pairs for your domains, and mention the public key in your DNS. It's explained here: https://www.digitalocean.com/community/tutorials/how-to-install-and-configure-dkim-with-postfix-on-debian-wheezy

I find this to be too cumbersome for my needs.

As I mentioned for SPF you can see the results of DKIM checks in “Show Original” from Gmail.

This looks like a good tutorial for DKIM: https://www.linode.com/docs/email/postfix/configure-spf-and-dkim-in-postfix-on-debian-8

Reverse DNS

You just have to change this where you can. Make sure that it's a hostname that is either the root domain or part of the domain from which you send email, and also that it matches the name the mail server uses as EHLO (`myhostname` in Postfix main.cf configuration file).

I don't think there's an easy way to check if Google has the right reverse record or something cached from the past. I find that if you don't encrypt your emails (which is the case by default) you can click the down arrow next to the lock and see “XXX did not encryt this message” where XXX is part of the name gotten through reverse DNS lookup.

DMARC

It's this: https://dmarc.org/

It seems to me that DMARC doesn't make sense if you don't have both SPF and DKIM and as of now I don't have DKIM.

TLS

With the default Postfix you're always going to see a red open lock next to “Me” when viewing emails from your server, with the mention that it did not encryt the message. I have no idea if this does anything to spam classification but I figured why not try.

When you talk to a Postfix with default TLS config it does present STARTTLS but apparently noone will use it? The author of Postfix is not fond of using external OpenSSL stuff as it may introduce more bugs so Postfix is one of thos MTA that has TLS completely disabled by default.

Anyway the preferred practice is to enable it like so in /etc/postfix/main.cf:

```
smtpd_tls_security_level = may
smtp_tls_security_level = may
```

Except these two only works with Postfix 2.3+, and Debian Stable has 2.1. Yeah.

In short we're probably going to need this: http://www.postfix.org/TLS_LEGACY_README.html

In fact by default we do accept server TLS. But what we want is client TLS, and to enable it we must add this line:

```
smtp_use_tls=yes
```

Now the client will propose TLS. Which will make Gmail mark the email as encrypted. Even if we're using the aweful self-signed so called snakeoil certificate. Apparently they don't care too much about authentication.

Google Headache

I have these things set up:

- Reverse DNS, correctly shown by Google (had to force refresh their cache, took 1 night to apply)
- SPF record working, soft fail (might try hard fail at some point)
- Registered and authenticated domain in Postmaster tools: https://postmaster.google.com (must be registered with a Google account)
  - Postmaster tools require a lot of email traffic from your domain to start recording stuff (at least they're saying that on their help page)
- Enabled TLS, getting the “Standard TLS” specified by Gmail
- Despair mode:
  - Changing mutt user agent to `Mozilla/5.0 (Windows NT 10.0; WOW64; rv:45.0) Gecko/20100101 Thunderbird/45.4.0` → Doesn't do anything.
  - Try using a smart host that is not hosted by OVH
    - At this point I started to get “dkvz.eu is sending too much spam” which is different to the content message I had before. So I panicked and marked everything as not spam.

This still marks me as spam for “content” reason.

Reputation lists

Gmail uses their own lists. But IP range do have reputation that sticks sometimes for a long time.

I see this site being referred to a lot when it comes to IP reputation: https://www.senderscore.org

You should create an account and check your IP and/or email domain score.

Another tool that can be kind of helpful: https://www.mail-tester.com ; You send an email to the given address, you wait for a while, then you click the button.

Sending MORE emails

People seem to be indicating (in postmaster.google.com and other places) that a “new” IP address sending email will be automatically considered suspect. Maybe by sending hundreds of emails I'll end up passing?

The conclusion

After setting up a script to send me an email with the content of the command `free -m` every 5 minutes, all the time, I ended up really having my emails not going to spam, even on a new account.

I'm not really sure what did it. Also I'm still not getting any info from postmaster.google.com

Block specific recipient addresses

Since we now have an infinity of email addresses getting accepted, we might want to block one or another at some point, when it's being polluted beyond repair.

You can do this using recipient restrictions. We could make this allow everything except what is mentioned as REJECT in the recipient hash files, or the contrary. In my case I want to allow everything except specific addresses. Source: http://www.linuxmail.info/postfix-restrict-sender-recipient/

Create a file: `/etc/postfix/recipient_access`.

The syntax goes as follows:

```
janedoe@acme.local  REJECT
bugsbunny@acme.com  OK
acme.com            REJECT
```

In my case I don't need any OK lines, this is an example.

Save your file and postmap it:

```
postmap recipient_access
```

Provided you are in /etc/postfix of course.

Now edit your /etc/postfix/main.cf and add this at the end:

```
smtpd_recipient_restrictions = check_recipient_access
hash:/etc/postfix/recipient_access, permit
```

This is the scenario where everything is accepted except what is rejected in recipient_access. If you want to contrary, you'll want this line instead:

```
smtpd_recipient_restrictions = check_recipient_access
hash:/etc/postfix/recipient_access,  reject_unauth_destinations
```

And your recipient_access files should then contain lines with OK at the end to specify the working mailboxes.

You need to reload Postfix to apply the changes. This is try everytime you make a modification to recipient_access and postmap the file (don't forget to postmap the file after modifications).