

Reclaiming Email with Astroid

Rohit Goswami · 2020-12-30 · rant · tools · email · workflow

Migrating Imap, Gmail and Exchange, mail accounts from GUI clients to [Astroid](#)

Background

Initially, I had planned this post to start with a brief history of the decline of email clients for Linux. That quickly got out of hand, and was therefore spun out into a post of its own (TBD). To keep things brief. Thanks to the incredible ineptitude of the Thunderbird steering committee, I ended up requiring a new mail client. Having despaired of the GUI based bloat heavy approaches of most clients, I decided to go the old fashioned route and build one up in a modular manner.

Packages

We will need the following:

astroid

for handling the email interface

isync/mbsync

for general IMAP account synchronization

lieer

for GMail accounts

davmail

for Exchange accounts

notmuch

for indexing emails

pass

for handling secrets (`secret-tool` or anything else would work too)

smtp

for sending email

nix

for handling dependencies in a reproducible manner (or `pixi`)

cryptomator

for encryption at-rest (or `tomb`)

rc1one

for handling cloud synchronization of email contents

The Accounts

In no order of preference, for a variety of reasons, I have 23 distinct email accounts. However, these are actually broken down into a few basic types and associated mail fetching software.

Generic IMAP

Yahoo, Mail.ru and the rest are managed with [isync](#)

Exchange

A school account of mine uses Office 365, and will be handled with [davmail](#) in conjunction with `isync`

Gmail

There are a few of these, two personal, and two institutional, all of which are handled with [lieer](#)

Security

We will use `pass` for handling the secrets [^fn:1].

```
pass git init $GPG_KEY # Optional, use a secure private git repo
pass init $GPG_KEY
```

Additionally, we will store the actual contents of the mail in an encrypted store. Generally `Cryptomator` works really well for data which is not updated very often, but has performance issues for many small file[^fn:2]. However, given its focus, and the fact that this setup can end up with a rather heavy set of files, we will instead use `tomb` which interacts with a `LUKS` volume ([website](#)).

****Warning****

Although LUKS volumes can be expanded to become larger, shrinking to reclaim space is not a supported operation [^fn:3].

In any case:

```
tomb dig -s 10000 mail.tomb
tomb forge mail.tomb.key
tomb forge mail.tomb.key -f
tomb lock mail.tomb -k mail.tomb.key
tomb open mail.tomb -k mail.tomb.key
```

Which means we can now use the `/media/mail` folder for storing data.

It is easiest to store the `.mail` folder where necessary and bind mount to a location like `/media/mail`:

```
sudo mount --bind "$HOME/.mail" /media/mail
```

ISync

For general IMAP accounts (anything which isn't backed by `gmail` or `outlook`) an `mbsync` approach works best. Exchange accounts need `davmail` and are described in a separate sub-section. Before we get to the `poll.sh` script and `astroid`, it is a good idea to run each of these as we set them up, especially as the first run can take quite a long time (around an hour for 9205 messages).

```
mbsync -V blah
```

General IMAP

For a standard IMAP account (anything which isn't Exchange or Gmail) the only thing we need to keep track of is a good way to obfuscate passwords. We will use `pass` for this, as configured earlier.

```
pass add mail/rgoswami.iitk
```

With this, we can configure a general IMAP account in `$HOME/.config/isyncrc` [^fn:4].

```
IMAPAccount rgoswami.iitk
Host qasid.iitk.ac.in
User rgoswami
PassCmd "pass show mail/rgoswami.iitk"
AuthMechs LOGIN
SSLType IMAPS

IMAPStore rgoswami.iitk-remote
Account rgoswami.iitk

MaildirStore rgoswami.iitk-local
SubFolders Verbatim
Path /run/media/Storage/.mail/rgoswami_iitk/
INBOX /run/media/Storage/.mail/rgoswami_iitk/Inbox
Trash trash:///

Channel rgoswami.iitk
Master :rgoswami.iitk-remote:
Slave :rgoswami.iitk-local:
Patterns *
Create Both
SyncState *
```

Exchange Accounts

For Exchange accounts we need to setup `davmail`. Thankfully the process is actually excessively simple for a single account. For each account, a `.properties` file needs to be generated. The password section is kept blank in this case, since we will authenticate with the `0365Interactive` protocol.

```
# The default setup
davmail ~/.davmail.properties
```

Use the following properties in the `.davmail.properties` file to prevent timeout errors:

```
davmail.folderSizeLimit=50
davmail.clientSoTimeout=0
davmail.enableKeepAlive=true
```

****Note****

``java-openjfx`` is required for the interactive setting, and ``swt`` for the icon.

Further information is available at [the official docs](#).

Main Network Encryption Logging Advanced

Gateway

Exchange Protocol: **O365Interactive** ▼

OWA or EWS (Exchange) URL: office365.com/EWS/Exchange.asmx

Local POP port: ☒ 1110 ☐ No SSL

Local IMAP port: ☒ 1143 ☐ No SSL

Local SMTP port: ☒ 1025 ☐ No SSL

Caldav HTTP port: ☒ 1080 ☐ No SSL

Local LDAP port: ☐ [Local SMTP server port to configure in mail client](#)

Delays

Trash keep delay (POP):

Sent keep delay (POP):

Calendar past events (Caldav):

IDLE folder monitor delay (IMAP):

Save Cancel Help

Figure 1: 'davmail' setup

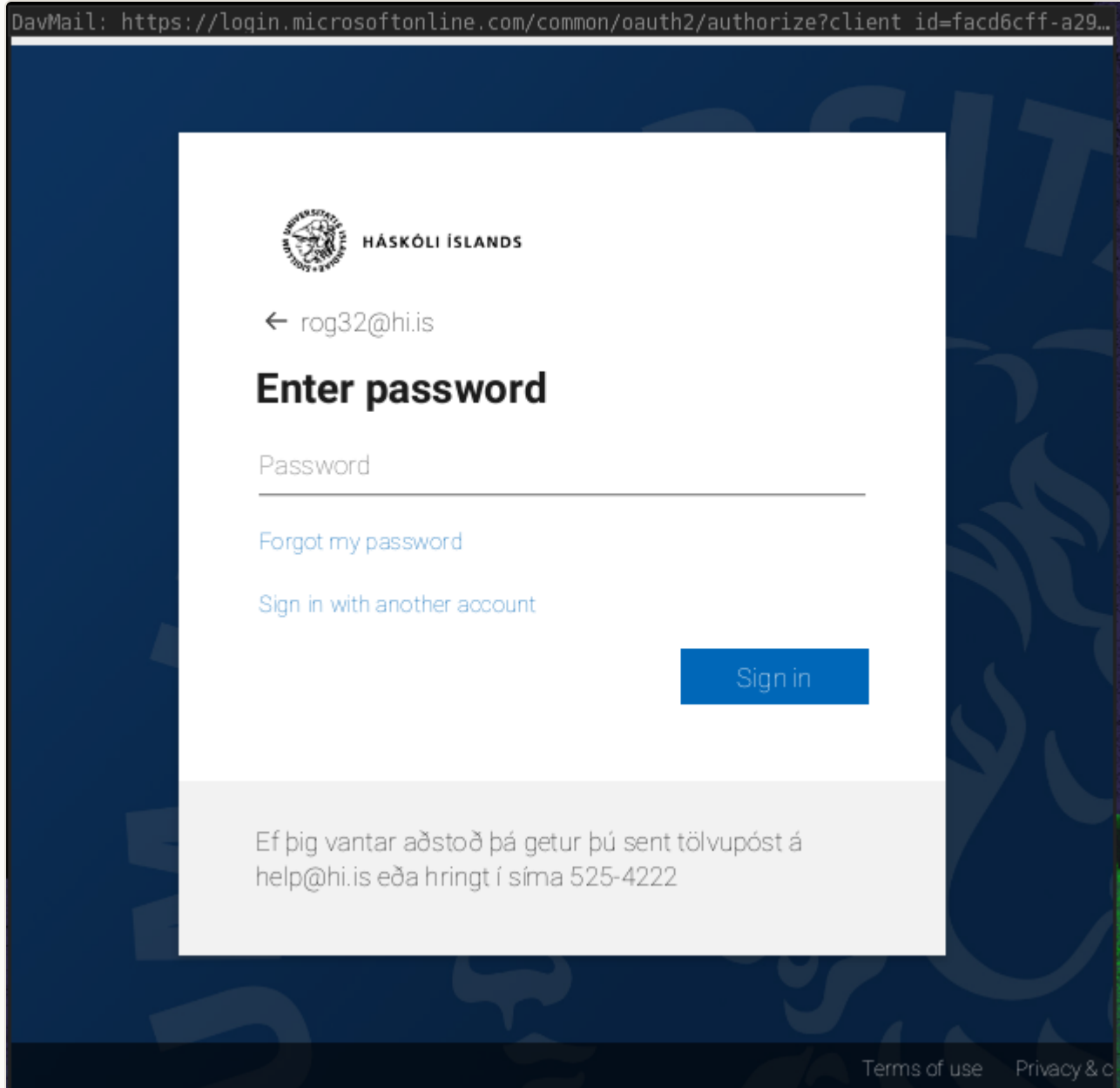


Figure 2: Interactive login workflow

Now we can configure the `.mbsyncrc` in a manner analogous to the standard IMAP setup, but with the port and host where `davmail` is running.

```
IMAPAccount rog32
Host localhost
User rog32@hi.is
Pass dummy
Port 1143
TLSType None
AuthMechs LOGIN

IMAPStore rog32-remote
Account rog32

MaildirStore rog32-local
SubFolders Verbatim
Path /media/mail/rog32/
Inbox /media/mail/rog32/Inbox

Channel rog32
Far :rog32-remote:
Near :rog32-local:
Patterns *
Create Both
SyncState *
```

The `TLSType` and `AuthMechs` are important parameters. Note that the `Pass` is truly not important since we an OAuth token is stored in the `properties` file.

```
> mbsync -V rog32
Reading configuration file /home/haozeke/.mbsyncrc
C: 0/1 B: 0/0 M: +0/0 *0/0 #0/0 S: +0/0 *0/0 #0/0
Channel rog32
Opening master store rog32-remote...
Resolving localhost... ok
Connecting to localhost (:::1):1143)...
Opening slave store rog32-local...
Logging in...
*** IMAP Warning *** Password is being sent in the clear
C: 0/1 B: 0/20 M: +0/0 *0/0 #0/0 S: +0/0 *0/0 #0/0
Opening master box INBOX...
Opening slave box INBOX...
Maildir notice: no UIDVALIDITY, creating new.
Loading master...
Loading slave...
slave: 0 messages, 0 recent
master: 6983 messages, 0 recent
Synchronizing...
C: 0/1 B: 0/20 M: +0/0 *0/0 #0/0 S: +3041/6983 *0/0 #0/0
```

Figure 3: First run of a large inbox with `davmail` and `isync`

Gmaileer

The general setup `gmi init blah@gmail.com` works well for personal accounts. Basically:

```
cd /media/mail
mkdir personalGmail
cd personalGmail
gmi init personal@gmail.com
```

However, there was an additional step for the IEEE account.

IEEE

Some issues with tags led to the following changes in the `.gmaileer.json` file

```
{"replace_slash_with_dot": false, "account": "rgoswami@ieee.org", "timeout": 600, "drop_non_existing_label":
false, "ignore_empty_history": false, "ignore_tags": ["TODO", "new"], "ignore_remote_labels":
["CATEGORY_SOCIAL", "CATEGORY_FORUMS", "CATEGORY_PROMOTIONS", "CATEGORY_PERSONAL", "CATEGORY_UPDATES"],
"remove_local_messages": true, "file_extension": ""}
```

Essentially just the addition of two new `ignore_tags`.

Notmuch

At this point, we have all mail synced into local directories, but we have no access to view or interact with them. We will start by setting up `notmuch` to search and index our mail. This is pretty basic for now, in `$HOME/.notmuch-config`.

```
[database]
path=/run/media/Storage/.mail/
[user]
name=Person
primary_email=primary@domain.com
other_email=one@domain.com;two@domain.com
[new]
tags=new;unread;inbox;TODO;
ignore=/. *[\.](json|lock|bak|osync_workdir|mbsyncstate|uidvalidity|log)$/
[search]
exclude_tags=deleted;spam;trash;
[maildir]
synchronize_flags=true
```

There are a lot more options which may be configured, but this is enough to get started.

```
notmuch setup
notmuch new
```

Sending Email

At this stage we need a way to actually send email. A simple configuration can be setup in

`/.config/msmtp/config` is given below (where we use `pass` again):

```
defaults
port 587
tls on
auth on
logfile ~/.config/msmtp/.msmtp.log
tls_trust_file /etc/ssl/certs/ca-certificates.crt

account r95g10
host smtp.gmail.com
from r95g10@gmail.com
user r95g10@gmail.com
tls_starttls on
tls on
auth on
port 587
password pass show mail/r95g10.gmail

account rog32
host localhost
from rog32@hi.is
user rog32@hi.is
tls_starttls off
tls off
auth plain
port 1025
password dummy
```

Note that the exchange account again uses a dummy password, and the real password will be prompted for on the first run. The permissions of the file above should be `600`. It is prudent to test at-least the exchange section to login.

```
echo "Hello world" | msmtp --account=rog32 r95g10@gmail.com
```

Astroid

The bulk of this setup is [by the numbers](#), with the exception of the poll script. However, minimally configure `astroid` with the location of our `notmuch` configuration.

```
astroid --new-config
vim ~/.config/astroid/config
```

The relevant sections are:

```
"notmuch_config": "\\home\\whoever\\.notmuch-config",
```

The accounts section is fairly self explanatory, but we will need to use the following `sendmail` line as well:

```
"sendmail": "msmtp --read-envelope-from -i -t",
```

With that, usage of `astroid` may commence.

Nix Python Poll Script

Natively only bash appears to be supported. However, with `nix`, we can use a reproducible python script with [the sh library](#) to call system functions instead.

```
from pathlib import Path
import sh

# For generic IMAP maildirs

ISYNC_LABELS = ["rgoswami.iitk", "rog32"]

for isync in ISYNC_LABELS:
    sh.mbsync("-V", isync, _bg=True)

# Gmaileer
GMAIL_IDENTIFIERS = ["gmail", "univ", "ieee"]

path = Path(r"/run/media/haozeke/Storage/.mail/")

for dirs in path.iterdir():
    if dirs.is_dir():
        for gmi in GMAIL_IDENTIFIERS:
            if gmi in dirs.name:
                print(f"Syncing {dirs.name}")
                sh.gmi("sync", _cwd=dirs, _fg=True)
```

This needs to be coupled with the standard `nix` shebang in the `poll.sh` file:

```
#!/usr/bin/env nix-shell
#!nix-shell -i python3 -p "python38.withPackages(ps: [ ps.numpy ps.sh ])"
```

Navigation

For working with HTML emails, we need to highlight the notice about potentially sketchy HTML using (defaults) `j` or `k` and then hit `enter` or `o`.



From: **AAAI-21 Conference** <aaai@memberclicks-mail.net>
To: rog32@hi.is <rog32@hi.is>
Date: December 29, 2020 11:40 pm (Yesterday)
Subject: **NEW!!** AAAI-21 New Faculty Highlights CFP: January 5
Tags: **inbox** **new** **TODO**

Alternative part (type: text/html) - potentially sketchy.

Figure 4: Unhelpful default

inbox (25991/39997) NEW!! AAAI-21 New Faculty Highli...

Dear AAAI Members and Affiliates,

This year, AAAI is launching a new invited speaker program highlighting AI researchers who have just begun careers as new faculty members or the equivalent in industry. Applications will be adjudicated by a committee consisting of the AAAI-21 chairs and a diverse group of AAAI Fellows.

New Faculty Highlight talks will be allotted 30 minutes each; the aim is for these talks to broadly survey the candidate's research to date. Several talks will be released online and publicized each day of the AAAI conference, following which they will be available archivally as part of the conference program. Invited speakers will be further invited to contribute an article to a corresponding series in AI Magazine.

We wish to establish the talk series as a regular program in future AAAI conferences with the eligibility dates set to one year before the conference dates, but considering the pandemic situation this year, we set the eligibility dates for AAAI-21 to be between July 1, 2019 and December 31, 2020.

To qualify applicants must:

1. Have taken up a faculty position at a research-intensive university or a broadly equivalent position in industry (e.g., a position in a research lab that involves independent research and publication at AI venues), with a start date between July 1, 2019 and December 31, 2020.
2. Have at least six papers published or accepted for publication in top conferences/journals OR have received a cumulative 500 citations OR have received a best paper award at one of the top conferences OR have transferred research into a deployed system that touches at least 1,000 people.

We invite applications from qualified researchers. Applications should consist of a cover letter explaining why the candidate meets each of the qualification criteria (with final adjudication of the criteria at the discretion of the committee), the candidate's CV, a link to the candidate's Google Scholar profile or equivalent, and a one-page statement describing the work that would be surveyed by a talk.

Applications may be submitted at

<https://www.surveymonkey.com/r/aaai21-new-faculty-highlights> and are due on or before January 5, 2021. Selected applicants will be required to provide a talk video for the committee to approve.

This email was sent to rog32@hi.is by aaai21@aaai.org

Association for the Advancement of Artificial Intelligence · 2275 East Bayshore Road, Suite 160, Palo Alto, California 94303, United States

[Remove My Email or Manage Preferences](#) · [Privacy Policy](#)

powered by MemberClicks

Figure 5: After viewing the sketchy bit

Note that since most email is actually sent with `text/html` it might make more sense to configure the `thread_view` in the configuration file.

```
```json { hL_lines=["3"] } "thread_view": { "open_html_part_external": "false", "preferred_type": "html",  
"preferred_html_only": "false", "allow_remote_when_encrypted": "false", "open_external_link": "xdg-open",
"default_save_directory": "~", "indent_messages": "false", "gravatar": { "enable": "true" },
```

“mark\_unread\_delay”: “0.5”, “expand\_flagged”: “true” },

```
Deletion {#deletion}
```

There are again, two different approaches to deletion.

Gmail

: For `gmail` accounts it is simple, just adding a `trash` tag will do the trick, so we can select multiple emails with `t` and then hit `+` to add `trash` and everything works out

Other Accounts

: We can **delete mail** forever (from the server as well), by using a safe-tag and then using `notmuch`

The generic non-`lieer` method requires:

```
```bash
```

```
notmuch search --output=files --format=text0 tag:killed | xargs -r0 rm
```

A `pre-new` hook will work.

Composition

Thankfully, `astroid` supports GPG encryption as well as markdown support. This makes for simple integration with any popular editor.

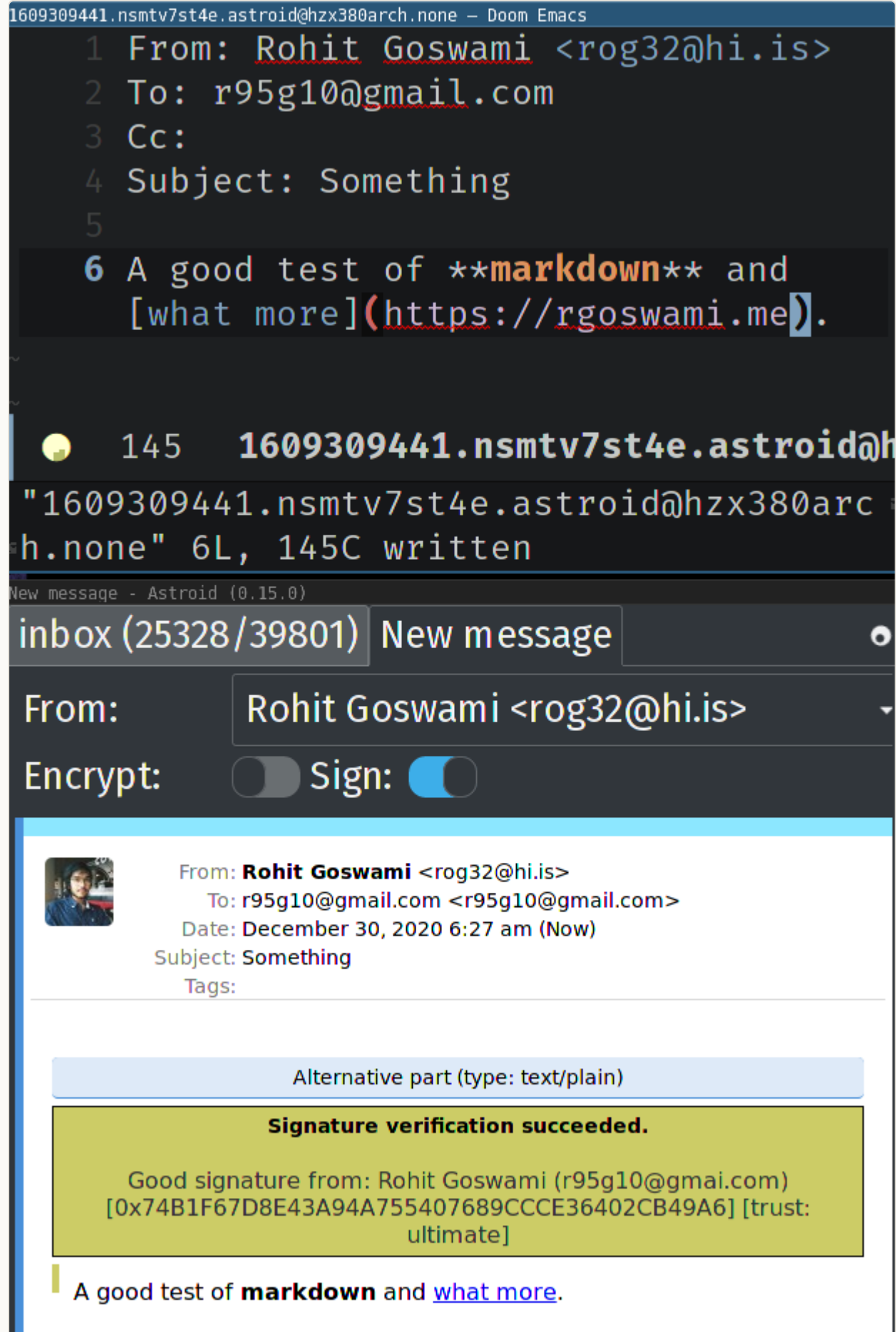


Figure 6: Composition with 'emacs' and 'astroid'

Conclusions

This is enough to get started for a while, but it isn't yet at the stage where I can replace `thunderbird` unfortunately. However, there are several pain points to be addressed, which will be covered in a future post. Some of these are essentially related to network fluctuations, and the awkward deletion setup.

Update

- `astroid` appears to be having a [bit of a development crisis](#)
 - The [sequel \(ragnarok\)](#) unfortunately seems to have gone towards a [browser frontend](#), which is quite unacceptable to me
 - This makes it far too similar to (in principle) Mailspring (though with less problematic connectivity issues)
 - This means I won't be moving forward with the next set of posts regarding `astroid` and will move towards `emacs` again
- [Max Mehl](#) has a [good collection of scripts](#) for `astroid`