Email Aggregation for Secure Local, Remote and Mobile Access

Frank Cox

October 27, 2013

Abstract

Revised October 28, 2013: *Minor corrections to the text and formatting, added more cross-references.*

This paper describes a method for aggregating email from several mailservers onto one computer, from where you can then access the email locally from the computer that stores your email, remotely from a laptop, or via your Android phone.

You could possibly accomplish something similar to this by forwarding all of your email to gmail or something like that, but this method allows you to keep all of your email on your own computer, where you can read it, sort it, archive it and delete it as you choose, without wondering where else your data may end up.

This method uses dovecot to provide an IMAP server on a Linux computer, fetchmail and procmail to poll mailservers and sort incoming email, postfix to send outbound email, and the Sylpheed email client for local and remote access, as well as K9Mail on Android for secure mobile access to email. Other email clients can easily be substituted for Sylpheed and/or K9Mail depending on your own preferences.

All remote and mobile user interaction with the email account(s) is done through a SSH tunnel, so no unencrypted data is ever exposed to the Internet at large or to whatever wifi network you happen to be connected to at the time.

Some mailservers require that outbound email appear to originate from certain IP addresses, which means that they would reject outbound email sent directly from a roving laptop or phone. The system described here routes outbound email from the remote and mobile email clients back to your main computer and it is forwarded to specified mailservers from there. Email sent from a remote or roving laptop or phone will look and act exactly the same as if it was sent directly from your local computer.

Note that security in this paper refers to the handling of email once it actually arrives on your computer. You don't have any control over the security of your incoming email before it gets to you, and you don't have any control over outbound email after it leaves your computer. You can, however, control the data that is actually present on your computer. For security of email in transit, you may wish to look at one of several available encryption options, the most popular of which is gpg.

Part I
System requirements

I currently use Centos 6 on both my main computer and my laptops, and an Android phone. Therefore, all of the setup details here relate directly to setting this up on Centos 6 and Android. You can certainly do this on other Linux distributions that aren't Centos, but some of the details may differ slightly.

Part II
Installing the software

If you're running Centos 6 or any other mainstream Linux distribution, most of the required software is probably installed on your computer already. What may not be installed is fetchmail, procmail and dovecot. "*yum install fetchmail procmail dovecot*" should get those for you.

1 Choice of email client

This document describes how to configure the Sylpheed email client for Linux and the K9Mail email client for Android. Other email clients will also work fine, of course.

You can download the latest stable version of Sylpheed for Centos 6 here `http://www.melvilletheatre.com/articles/el6`.

Part III
Getting incoming email onto the main computer

What we do here is use fetchmail to poll each of the mailservers for new email every X number of minutes, then procmail sorts incoming email and puts it into the appropriate mailbox.

2 Create Maildir directory

The first step is to create one main directory and three subdirectories in your home directory:

- mkdir ~/Maildir
- cd ~/Maildir
- mkdir tmp
- mkdir cur
- mkdir new

3 Fetchmail configuration

Fetchmail is the program that polls outside mailservers and downloads incoming mail to your computer.

Fetchmail reads a file named .fetchmailrc in your home directory. Since it probably doesn't exist yet, create it. Here is a sample ~/.fetchmailrc file:

```
set postmaster "myusername"
set logfile $HOME/Maildir/fetchmail.log
poll private.example.com
```

```
        proto POP3
        interval 50
        timeout 60
        user 'user1' there with password 'password1' is 'myusername' here
        ssl
        sslfingerprint "47:C6:CC:4C:A8:B8:22:75:DD:3E:E9:D2:50:EE:38:51"
        sslcommonname "imap.example.com"
        mda "/usr/bin/procmail -d %T"
        nokeep

#
poll mail.example.net

        proto POP3
        timeout 60
        user 'user' there with password 'password1' is 'myusername' here
        user 'anotheruser' there with password 'anotherpassword' is 'myusername' here
        mda "/usr/bin/procmail -d %T"
        nokeep

#
poll pop.gmail.com

        proto POP3
        timeout 60
        user 'example@gmail.com' there with password 'gmailpassword' is 'myusername' here
        ssl
        mda "/usr/bin/procmail -d %T"
        nokeep
```

First, we have two global settings: the "postmaster" is set to myusername, which is my user name on the local computer. This allows for any mail collected by fetchmail that would otherwise be undeliverable for whatever reason to be given to myusername instead of vanishing. Also, we set the fetchmail logfile to be /home/myusername/Maildir/fetchmail.log.

Running "*fetchmail -V*" will give you a whole lot of information about the current configuration that fetchmail is using, either the default settings or those that you have changed in .fetchmailrc

There are three different types of mailserver listed here and each one requires a slightly different method for a POP3 login to be successful.

The timeout line sets the time that fetchmail will wait for a response from the mailserver to 60 seconds. The default timeout (if you don't set one yourself) is 300 seconds, which seems a bit excessive.

PRIVATE.EXAMPLE.COM: This is a private mailserver that has a self-signed ssl certificate. Therefore, to prevent fetchmail from complaining about an untrusted security certificate. You can get the ssl fingerprint from a mailserver by running this command:

```
fetchmail -v -p PROTOCOL -u USERNAME MAILSERVER
```

For example:

fetchmail -v -p POP3 -u user1 private.example.com

From the resulting output, key fingerprint and Issuer CommonName are the two fields that you are interested in.

Since in this example we don't get much mail from private.example.com, we use the interval keyword to tell fetchmail to check this mailbox only every fiftieth time that fetchmail is run.

MAIL.EXAMPLE.NET is a public mailserver where we have two usernames (mailboxes) that we want to pick up incoming email from

POP.GMAIL.COM is Google's gmail service; those are the settings that you need to pick up your email from gmail. Just substitute your username and password, and you're all set.

Incidentally, even though we're using ssl in the gmail settings, we don't need to specify a sslfingerprint because gmail uses a trusted ssl certificate. As long as the ca-certificates rpm is installed on your computer, which is probably is, you already have everything you need to authenticate your connection to gmail.

The mda line tells fetchmail that we're going to be using procmail to sort and deliver the incoming email, and "nokeep" tells fetchmail to delete incoming mail off of the outside mailserver after it has been picked up.

Set the permissions on your ~/.fetchmailrc file to 600, i.e run "*chmod 0600 ~/.fetchmailrc*"

4 Procmail configuration

As seen in the previous section, procmail is called by fetchmail to sort incoming mail and deliver it to the mailboxes on your computer.

I copied most of this file from `http://uce.uniovi.es/tips/Internet/Fetchmail.html`, and edited it very slightly. Assuming that you set up your system to match the description in this article, you don't need to change it at all for it to to work with your system. (You will, of course, want to add more recipes to it later to sort your email into submailboxes and discard spam and so on. That is beyond the scope of this article.)

This file should be named ~/.procmailrc

```
SHELL=/bin/bash
# Have to have this one (or whatever your shell is)
# Best bet is bash or sh.
LINEBUF=4096
# Magic. Apparently it burps on long lines if you don't
# put this in.
#PATH=/bin:/usr/bin:/usr/local/bin
# Where procmail looks for stuff. Works for practically
# every Linux distro I have seen.
VERBOSE=off
# Change to 'on' to get _long_ procmail log.
# NB: if this is short, I don't want to see long: I get
# a one-line summary for every email procmail looks at!
MAILDIR=$HOME/Maildir
# Not where your mail arrives on the machine. Where
# procmail will assume all the folders you mention in
# your recipes goes. Make sure your email-reading
# program also knows about it. (I understand $HOME/Mail
# is pretty standard, however.)
# Make sure that $MAILDIR exists and that it is a directory!
```

```
DEFAULT=$MAILDIR/
# See the slash!
# DEFAULT must end in a trailing / to inform Procmail
# to use maildir storage
LOGFILE=$HOME/Maildir/procmail.log
# I don't think this needs to be in your Mail folder,
# but my mail-reader (mutt) is great at different
# sorting, so I put the log into the mail directory :)
# Note learned through experience: if you leave this file
# too long, it will end up with tens of thousands of
# messages. Delete it regularly!
# To insert a blank line between each message's log entry in $LOGFILE,
# uncomment the next two lines (this is helpful for debugging)
LOG="
"
FORMAIL=/usr/bin/formail
# 'formail'. Part of the procmail package. Correct
# the path if this isn't where it lives for you.
# ('which formail' may well tell you.)
SENDMAIL=/usr/sbin/sendmail
# As with formail, tells procmail where to look for
# sendmail. If sendmail isn't there, mail transfer
# might be handled by a different program. Ask
# your sysadmin :) If you are your own sysadmin,
# then I hope you know.
# Subsequent to writing that, I have learned that this
# file is provided (with this name) by other MTAs too.
##############################
# The recipes
######
# Work around procmail bug: any output on stderr will cause the "F" in "From"
# to be dropped. This will re-add it.
:0 * ^^rom[ ]
{ LOG="*** Dropped F off From_ header! Fixing up. "
:0 fhw
| sed -e '1s/^/F/'
}
#
# Your recipes go here to sort your mail
#
#
#Save all remaining messages to default if not otherwise handled above
:0
$DEFAULT
```

Run the command *chmod 0600 ~/.procmailrc* to set the file permissions to what procmail expects to see.

5 Dovecot configuration

Dovecot is the IMAP server that the email clients (Sylpheed and K9Mail) will talk to so you can read your email.

On Centos 6, there are only two changes needed in the default dovecot configuration.

In */etc/dovecot/dovecot.conf* change the line that reads *#protocols = imap pop3 lmtp* to read *protocols = imap* instead.

In */etc/dovecot/conf.d/10-mail.conf* change the line that reads *# mail_location = maildir:~/Maildir* to read *mail_location = maildir:~/Maildir* instead.

That's it. Now you can start dovecot by issuing the command */sbin/service dovecot start*

You will probably want to have dovecot start automatically every time you boot your computer. *chkconfig dovecot on* will do that for you.

6 Basic Sylpheed configuration on your main computer

The easiest way to create new sub-mailboxes is to do it through Sylpheed. That way all of the necessary subdirectories get created automatically and the program does all of the heavy lifting for you.

Load Sylpheed. The first window asks you to create a mailbox, but you don't need to do that since we already have one (the ~/Maildir directory). Therefore, click Cancel on that window and confirm that you really don't want to create a mailbox.

Here we are on "New account setup". Select IMAP4 as the account type, then input your display name and your primary email address. On the next screen enter your username (what you used for "myusername" in the ~/.fetchmailrc configuration file) into the User ID field, and "localhost" (without the quotes) into the other two text fields (IMAP4 server and SMTP server). Leave the other options at their default values, i.e. un-checked.

Sylpheed will then ask you to enter your password. Do that, and you will be presented with a screen showing your first two mailboxes, INBOX and Trash.

6.1 Clearing Sylpheed's IMAP cache

Sylpheed automatically populates a cache of messages as you access them. This is not particularly useful if you're using an IMAP server hosted on your own local computer, and it can add up to quite a bit of disk space being used on a remote computer as well (though the cache is actually useful in that situation).

If you want Sylpheed to clear its imap message cache whenever you exit the program, go to Edit Accounts - Advanced and check "Clear all message caches on exit".

6.2 Creating sub-folders under Maildir

To create additional folders and sub-folders under Maildir, right-click on on the folder or sub-folder where you want the new sub-folder, and select Create New Folder. Name your new folder, click OK and that's there is to that. You can then use procmail recipes to automatically sort incoming email into the appropriate location if desired.

You can also create special folders such as Drafts, Queue, Sent and Junk by right-clicking on a folder, clicking on Properties, and selecting the appropriate Type from the drop-down list.

If you don't want Sylpheed to ask you to enter your password every time you start it up, you can go to Configuration - Preferences for current account, enter your password in the Password field, and save that setting.

7 Testing your incoming email

Now would be a good time to try this out and make sure you can receive and read incoming email. Send yourself an email (not with Sylpheed since we haven't set that part up yet) or do something to insure that you actually have some email to pick up. Then type this into a terminal window: *fetchmail -v private.example.com*

The -v option causes all control messages passed between fetchmail and the mailserver to be echoed to stdout so you can be sure that everything is working. You can omit the -v later on after you know that it's working if you wish.

Click the Get button at the top left of the Sylpheed window or re-load Sylpheed to make the program check your Maildir for new email. And if all went as it should, you have one (or more) new messages in INBOX/default, which you can click on and read.

8 Adding additional email accounts

You can add the rest of your email accounts to Sylpheed any time by going to Configuration - Add new account.

Set each account up using the same information as in section 6 above, changing only the display name and email address that you want to associate with your other account(s), and setting the account type to POP3. We don't actually intend to do any POP3 mail transfers with this account, but the local/none protocol choice seems to have disappeared from Sylpheed. If you are using a different mail client you can choose the local/none option if it is present.

The only thing that additional accounts here are good for is changing the From: line on outbound email, since fetchmail already picks up inbound email for all of your accounts. Therefore, if you never actually send email from any particular account, there is no need to add it to Sylpheed; incoming email to those accounts will still show up in your mailbox as usual.

Go to Configuration - Edit Accounts and un-check the G (Get all) box beside your newly created account since we don't actually want that account to check for new mail, and you will have an error message pop up if it does.

8.1 Saving sent email

The easiest way to save sent email, i.e. outbound email that you sent to other people, is to create a Sent folder using the procedure shown in subsection 6.2above, then go to Configuration - Common Preferences - Send - General, and check the box that says "Save sent messages to outbox".

Another method for saving sent email is described in section 11.1.

9 Moving existing email into IMAP

By default, Sylpheed saves email in MH format. If you already have a quantity of previously received or sent email in MH format that you want to move into your new IMAP setup, you can simply create your IMAP account as described in section 6. This will append the IMAP folders to the bottom of the list of mail folders. Now you can select the email that you want to move (Select All may come in handy here), then right-click on the messages, select Move and place the messages in your new IMAP folders.

After moving all of your email into the IMAP folders, you can delete the old MH mailboxes by left-clicking on the MH mailbox at the top of the list, then selecting File - Mailbox - Remove Mailbox from the menu at the top of the window. Note that this does not actually delete the ~/Mail directory; if you want to delete that you have to do that manually.

10 Setting up Fetchmail to retrieve email automatically

One way to have fetchmail continuously retrieve all incoming email for you at specified intervals is to run it in daemon mode. However, depending on your needs, this may be overkill.

When you use fetchmail to retrieve email from a number of mailservers, some may be more important than others, therefore you may wish to retrieve the email at various intervals depending on the mailserver.

Fetchmail offers an *interval* keyword that tells it to check a particular mailserver only every N poll cycles (shown in the example .fetchmailrc file provided in section 3.

Another method that doesn't require running fetchmail in daemon mode or use of the *interval* keyword is to simply set it up to run as a cron job. Instead of using the *poll* keyword in .fetchmailrc, you can use *skip* which tells fetchmail to skip that mailserver unless it is explicitly called on the commandline. Therefore, a mailhost's entry in your ~/.fetchmailrc file might look something like this:

```
skip mail.example.com

    proto POP3
    user 'user1' there with password 'password1' is 'myusername' here
    mda "/usr/bin/procmail -d %T"
    nokeep
```

Now you can set up your crontab to poll each mailserver at exactly the intervals you require:

```
# poll every five minutes
*/5 * * * * /usr/bin/fetchmail mail.example.com &> /dev/null
# poll every ten minutes
*/10 * * * * /usr/bin/fetchmail example.net &> /dev/null
# poll once per day at 11am
00 11 * * * /usr/bin/fetchmail pop.example.com &> /dev/null
# poll every half hour
*/30 * * * * /usr/bin/fetchmail pop.example.net &> /dev/null
```

This method allows you to easily determine exactly what times and exactly what intervals you want your email picked up for each individual mailserver in your ~/.fetchmailrc file.

11 Procmail recipes

That's all there is to receiving email using fetchmail and procmail. Now all we need to do is automatically sort email into submailboxes based on things like subject or sender's address. However, since everyone's needs in this regard are going to be completely different, a complete discussion of how to create procmail recipes is beyond the scope of this document. There are numerous good tutorials on how to make this work available, though. A quick web search for "procmail recipes" will give you more information than you can use.

11.1 Alternate method for saving sent email

You can configure postfix to recognize a + sign as a separator in email addresses, such that email sent to joeblow@example.com and joeblow+whatever@example.com are both delivered to joeblow@example.com. How to configure postfix to recognize this addressing scheme is addressed below in section 23.

Once postfix has been configured to recognize a + sign, you can simply tell your email client to automatically Bcc: a copy of all outbound email to you. With Sylpheed, the Compose tab on the Account Configuration window has a section at the bottom, "Automatically set the following addresses".

Add your local address to the Bcc field. Note that this is not your public email address. For example, if my username on my local computer is joeblow, then my local address is *joeblow@localhost*. Add this to the Bcc field: *joeblow+sent@localhost*.

Now all email sent by you through Sylpheed will be automatically copied to your local address.

Assuming that your username is joeblow and that a Sent folder exists under under Maildir, here is the procmail recipe that will put incoming email to *joeblow+sent@localhos*t into the Sent folder:

```
# Sent
:0
* ^To.*joeblow\+sent@localhost
.Sent/
```

The method shown in section 8.1 is simpler than this and gets you much the same result, as long as your email clients support it and both Sylpheed and K9Mail do.

Part IV
Postfix setup for sending outbound email

12 The simplest possible relay

In its default configuration, postfix will send outbound email directly from your computer to any other computer.

In most cases, that's not what you want. You want to forward outbound email to a mailserver and so it can be passed on from there. Accordingly, you need to tell postfix what mailserver to relay your email through.

The simplest case is where you are relaying email through a mailserver that doesn't require authentication. In that case, simply uncomment the line in /etc/postfix/main.cf that says

```
# relayhost = $mydomain
```

Remove the # and change $mydomain to the name of your outbound mailserver:

```
relayhost = mail.example.com
```

Run the *postfix reload* command.

Now all email sent from your computer via postfix will be relayed through your mailserver.

13 Relaying using SMTP Authentication

If you want to relay your outbound email through a mailserver that requires SMTP Authentication, as most public mailservers do, you need to insure that the cyrus-sasl-plain is installed on your computer:

```
rpm -q cyrus-sasl-plain
```

It's probably installed. If it isn't, get it.

Add a few more lines to /etc/postfix/main.cf, as follows:

smtp_sasl_auth_enable = yes
relayhost = mail..example.com
smtp_sasl_password_maps = hash:/etc/postfix/sasl_passwd

Create the file /etc/postfix/sasl_password; enter the name of the mailserver followed by white space (space or tab) followed by your username on the mailserver, a colon, and your password on the mailserver.

mail..example.com username:password

It is optional, but wise, to make the sasl_passwd file read+write only for root to protect the username/password combinations from prying eyes. The Postfix SMTP client will still be able to read the SASL client passwords.

Run the *postmap /etc/postfix/sasl_passwd* command. This must be done after making any changes the sasl_passwd file.

Run the *postfix reload* command.

If you can't send email through the target mailserver, check /var/log/maillog and find out why. One common error is this:

SASL authentication failure: No worthy mechs found

This error occurs because the Postfix SMTP client does not support plain text auth mechanisms by default; if you get this error the solution is to add this directive to /etc/postfix/main.cf:

smtp_sasl_security_options = noanonymous

14 Sender-dependent relaying

You can use forward email to different mailservers depending on the email address that you used to send by adding a few more lines to /etc/postfix/main.cf, as follows:

smtp_sender_dependent_authentication = yes
sender_dependent_relayhost_maps = hash:/etc/postfix/sender_relay

Now your /etc/postfix/main.cf file may look something like this:

smtp_sender_dependent_authentication = yes
sender_dependent_relayhost_maps = hash:/etc/postfix/sender_relay
smtp_sasl_auth_enable = yes
relayhost = mail..example.com
smtp_sasl_security_options = noanonymous
smtp_sasl_password_maps = hash:/etc/postfix/sasl_passwd
relayhost = mail.example.com

The next step is to create the table that allows postfix to associate an outbound email address with a mailserver. If postfix finds the sender's address in the table, it will use that mailserver. If the sender's address is not in the table, postfix will use the relayhost specified in /etc/postfix/main.conf, as described above.

Create the file /etc/postfix/sender_relay, as follows:

user@example.com mail2.example.com
user2@example.net mail.example.net
user3@example.com mail3.example.com

Run the *postmap /etc/postfix/sender_relay* command. This must be done after making any changes the sender_relay file.

Run the *postfix reload* command.

Some mailservers accept email only from specific IP ranges or addresses. This situation generally arises only when you have more than one gateway on your network, and wish to use mailservers provided by different ISP's. If this applies to you, see section 26.

Part V
Configure Sylpheed and ssh tunnelling on a roving computer

This is where the secure part of secure remote and mobile access to your email comes in. All access to your IMAP mailbox from anywhere other than your own local computer will be done through a ssh tunnel. No unencrypted data will be exposed.

Setting up a ssh login to your computer is beyond the scope of this document. We assume that you can log into your computer from somewhere else using ssh; if not there is plenty of documentation available to help you get that going. For maximum convenience and security you will probably want to disable remote password logins and use only public key authentication. You may also want to set up ssh to listen on a port other than the default port 22; while this doesn't give you a huge increase in security, it doesn't hurt anything either.

Using this method, it is possible to set up Sylpheed on a roving computer in such a way that the ssh tunnelling is both automatic and completely transparent to the user. In other words, loading and using Sylpheed on a roving computer loads, looks and operates exactly the same as it does on the local computer.

15 Configure Sylpheed on roving computer

This part is easy – configure Sylpheed in exactly the same way as you configured it to work on your local computer, as described in section6.

There are only two changes to make from that configuration.

Click on Configuration - Edit accounts, and select your IMAP account. Click on Edit, then the Advanced tab.

Check off Specify SMTP port, and change the number to 2025. Check off Specify IMAP port, and change the number to 2143.

That's it. Sylpheed is now configured on your roving computer.

TIME SAVER: You can copy everything in the ~/.sylpheed-2.0 directory on your local computer to your roving computer, then load sylpheed and make the above changes.

16 Configure the ssh tunnel on roving computer

There are two options for configuring the ssh tunnel on a roving computer – the choice depends on the stability of the Internet connection involved and the term that you intend to leave the roving computer connected to your IMAP server.

16.1 The simple way to configure the ssh tunnel and run Sylpheed

This is the commandline to run on your roving computer:

ssh -C -f -L 2025:localhost:25 -L 2143:localhost:143 username@IMAP_server sleep 10; sylpheed

This command does the following things:

-C Compress the data transferred between the IMAP server and the roving computer. Since a majority of email usually consists of text, this option will make a noticeable difference in performance.

-f Requests ssh to go to background just before command execution.

-L Specifies the ports that we want to forward. In this case we aer forwarding port 2025 on the roving computer to port 25 on the IMAP server, and port 2143 on the roving computer to port 143 on the IMAP server. Note how these match the Sylpheed settings that we changed in section 15; port 25 is the standard SMTP port, used here to send outgoing email, and port 143 is the standard IMAP port, used here to talk to the IMAP server and read email.

IMAP_server This is the name or IP address of your IMAP server, i.e. your local computer. This is the only part of this commandline that you don't use literally as it is written here. EXAMPLE: If your IMAP server is located at IP address 203.0.113.105, the commandline that you want to use is this: *ssh -C -f -L 2025:localhost:25 -L 2143:localhost:143 joeblow@203.0.113.105 sleep 10; sylpheed*

sleep As soon as we log into the IMAP server, we do nothing. This holds the ssh tunnel open for ten seconds.

sylpheed This runs Sylpheed.

What this commandline does is log into your IMAP server and forward two ports between the computers. It then loads and runs the email client which immediately starts communicating with the IMAP server.

The interesting part of this commandline is that the sleep command holds the tunnel open for ten seconds, then exits. Since Sylpheed is now using the tunnel, it can't be closed until Sylpheed exits. When that happens the tunnel will be automatically closed.

In this way, the operation of Sylpheed on a roving computer can be made absolutely transparent. The user simply loads Sylpheed using the above commandline, the ssh tunnel is established and held until Sylpheed exits, and the ssh tunnel is closed.

16.2 The bulletproof way to configure the ssh tunnel and run Sylpheed

The method described in section 16.1 works great if you're planning to connect to the IMAP server, check your mail and then move on. However, if you wish to maintain the ssh connection for hours, days or weeks at a time so you can continuously check your email, the ssh tunnel may unexpectedly disappear due to timeouts or network problems or who knows what else. If that happens, it's not the end of the world – you can just close Sylpheed and re-open it and the ssh tunnel will be re-established and everything will start working again.

However, it would be better to simply avoid that occurrence completely, and the solution is called *autossh*, a utility that can automatically restart ssh tunnels. If a ssh connection is unexpectedly dropped, autossh will automatically restart it with no user interaction required. As the whole thing runs in the background, the user may never be aware that the connection had disappeared.

Note that this method requires that you use public key authentication to your IMAP server; it won't work with password authentication. If you can use only password authentication for some reason, you must use the method described in section 16.1.

The homepage for autossh is located at `http://www.harding.motd.ca/autossh/index.html`, and if you are using Centos 6 and have the epel repository set up, you can install it on your roving computer with a simple *yum install autossh* command. (You don't need to install it on the IMAP server.)

After installing autossh, this is the commandline to run on your roving computer:

```
autossh -M 0 -q -f -o "ServerAliveInterval 60" -o "ServerAliveCountMax 3" -L 2025:localhost:25
-L 2143:localhost:143 username@IMAP_server sleep 10; sylpheed
```

While it is somewhat longer, you will note that this commandline is very similar to the commandline in section 16.1. It uses exactly the same technique to hold the ssh tunnel open for as long as you have your email client running, and as soon as you close your email client the tunnel will be automatically closed.

When using this method, the operation of Sylpheed on a roving computer is still absolutely transparent. The user simply loads Sylpheed using the above commandline, the ssh tunnel is established, held and automatically restarted the connection drops unexpectedly, and when Sylpheed exits, the ssh tunnel is closed.

Part VI
Configure VX Connectbot and K9Mail on Android

Unfortunately, there doesn't appear to be a way to set this stuff up on Android to be as completely transparent as it can be on a roving Linux computer. Two separate actions are required. First, establish the ssh tunnel. Then load K9Mail. When finished reviewing your email, close the ssh tunnel again.

17 How to establish a ssh tunnel on Android

Download and install VX Connectbot on your Android device. Configure VX Connectbot to connect with your IMAP server, and under the Port Forwards section, make the following entries:

Nickname: smtp

Type: local

Source port: 2025

Destination: localhost:25

Nickname: imap

Type: local

Source port: 2143

Destination: localhost:143

You can configure VX Connectbot to provide you with a command line on the IMAP server or not, at your option.

18 How to set up K9Mail on Android

The setting that you use with K9Mail on Android are almost identical to the settings you use with Sylpheed on Linux. Set up a new email account, enter your name and primary email address, then tell K9Mail that this is an IMAP account and enter your username and password on the IMAP server.

The IMAP server name is localhost, Security is none, Authentication is Plain, port is 2143.

The SMTP server name is localhost, Security is none, port is 2025. Don't check the Required sign-in box.

K9Mail is a bit more convenient than Sylpheed when it comes to setting up alternate identities for your outbound email. You don't have to create a separate "dummy" email account to use different email addresses on your outbound email. Under the Sending Mail option on the Setting menu, there is a Manage Identities section. You can just enter your alternate names and/or email addresses there.

19 Using VX Connectbot and K9Mail on Android

Whenever you want to check your email on your Android device, you must first load VX Connectbot and establish the ssh tunnel. When you are finished, you may disconnect the ssh tunnel unless you want to leave it running, which you can do if desired. See what I mean about this not being quite as transparent as it can be on a Linux computer?

- Load VX Connectbot, click on the menu item to connect to your IMAP server.
- Load K9Mail.
- Review your mail.
- When finished, load VX Connectbot again and click on Disconnect to close the ssh tunnel if desired.

Part VII
Summary

20 Summary: Maildir, fetchmail, procmail, dovecot, Sylpheed

20.1 Required maildir directories 2

Create one main directory and three subdirectories in your home directory:

1. mkdir ~/Maildir
2. cd ~/Maildir
3. mkdir tmp
4. mkdir cur
5. mkdir new

20.2 Fetchmail and procmail 3 4

On your local computer that will aggregate the mail and act as the IMAP server, you must set up .fetchmailrc and .procmailrc. An example .fetchmailrc is provide in section 3. An example .procmailrc is provided in section 4.

1. chmod 0600 ~/.fetchmailrc
2. chmod 0600 ~/.procmailrc

20.3 Dovecot 5

1. In */etc/dovecot/dovecot.conf* change the line that reads *#protocols = imap pop3 lmtp* to read *protocols = imap*.
2. In */etc/dovecot/conf.d/10-mail.conf* change the line that reads *# mail_location = maildir:~/Maildir* to read *mail_location = maildir:~/Maildir*.

20.4 Sylpheed 6

1. Cancel Sylpheed Create Mailbox window.
2. Select IMAP4 as account type.
3. Enter localhost into IMAP4 server and SMTP server fields.

21 Summary: Postfix IV

1. edit /etc/aliases to change *root marc* to *root yourusername*
2. edit /etc/postfix/main.cf, add this section:

```
smtp_sender_dependent_authentication = yes
sender_dependent_relayhost_maps = hash:/etc/postfix/sender_relay
smtp_sasl_auth_enable = yes
relayhost = mail..example.com
smtp_sasl_security_options = noanonymous
smtp_sasl_password_maps = hash:/etc/postfix/sasl_passwd
relayhost = mail.example.com
user@example.com mail2.example.com
user2@example.net mail.example.net
user3@example.com mail3.example.com
```

3. Create the file /etc/postfix/sasl_password, and make it read-write only for root:

```
mail..example.com username:password
```

4. Create the file /etc/postfix/sender_relay:

```
user@example.com mail2.example.com
user2@example.net mail.example.net
user3@example.com mail3.example.com
```

5. Execute the following commands:

   (a) postmap /etc/postfix/sasl_passwd

   (b) postmap /etc/postfix/sender_relay

   (c) postfix reload

22 Summary: Remote email clients

22.1 Roving computer 15

Sylpheed configuration

Click on Configuration - Edit accounts, and select your IMAP account. Click on Edit, then the Advanced tab.

Check off Specify SMTP port, and change the number to 2025. Check off Specify IMAP port, and change the number to 2143.

Simple ssh tunnelling command line

ssh -C -f -L 2025:localhost:25 -L 2143:localhost:143 username@IMAP_server sleep 10; sylpheed

Autossh commandline

autossh -M 0 -q -f -o "ServerAliveInterval 60" -o "ServerAliveCountMax 3" -L 2025:localhost:25 -L 2143:localhost:143 username@IMAP_server sleep 10; sylpheed

22.2 Android VI

VX Connectbot port forwards

Nickname: smtp

Type: local

Source port: 2025

Destination: localhost:25

Nickname: imap

Type: local

Source port: 2143

Destination: localhost:143

K9Mail configuration

IMAP server localhost

Security none

Authentication Plain

Port 2143

SMTP server localhost

Security none

Port 2025

Part VIII
Miscellaneous

23 Postfix support for the + sign address delimiter

By default, postfix doesn't recognize a + sign as an address delimiter. Therefore, if your username is joeblow, your localhost address is *joeblow@localhos*t, but *joeblow+sent@localhos*t is invalid because that's not your username.

Add this line to the bottom of /etc/postfix/main.cf:

recipient_delimiter = +

Run the *postfix reload* command and now postfix will recognize any valid username followed by a + sign, followed by any text you wish.

This is useful if you want to use the scheme for keeping a copy of your sent email described in section 8.1.

24 Telling postfix to use procmail as the local delivery agent

By default, postfix delivers local mail by itself. However, if you want to be able to have email sent to localhost (such as status and error messages discussed in section 25) sorted into a submailbox for you, postfix can be told to use procmail as the local delivery agent, and then you can create a procmail recipe to sort the localhost email in any way you choose.

IMPORTANT: If you do this, you MUST set up an alias to forward root's email to a real user as described in section 25.

Telling postfix to use procmail as the local delivery agent requires one line in /etc/postfix/main.cf:

mailbox_command = /usr/bin/procmail -a $DOMAIN

Note: Copy this line just as written above. You don't have to substitute anything for the $DOMAIN variable; what it says here is exactly what you want.

25 Reading root's email

It is a good idea for someone to read email sent to root. Email to root consists of automatic status messages and error messages and the like, and when your machine is interacting with others on the Internet it is good to know that it's not misbehaving.

To start receiving email sent to root on your local machine, un-comment the following line in */etc/aliases*:

```
# root: marc
```

Remove the # and change "marc" to your username.

Run the *newaliases* command to update the data table that postfix uses.

Run the *postfix reload* command.

You can test this by sending an email to root from the commandline, as follows:

```
echo message | sendmail root
```

This command sends a message containing the text "message" to root which is automatically forwarded to the username that you specified in /etc/aliases.

26 Custom routing to different mailservers

Do you have more than one ISP providing service to your network, and more than one gateway? For example, you may have a cable ISP providing service to a router on 192.168.0.1 and a DSL ISP providing service to a router on 192.168.0.254.

Assume that the network card in your computer is eth0, the default gateway on your computer is set to use the cable ISP on 192.168.0.1, and you want to route email to a mailserver located at ip address 203.0.113.105 via the DSL connection.

Edit /etc/rc.local on your computer and add this line:

```
route add -net 203.0.113.105 netmask 255.255.255.255 gw 192.168.0.254 dev eth0
```

You can have as many route add lines as you need, and you can specify specific routing like this for any number of networks cards as well.

27 How to list all open ssh connections

This command, which must be run as root, will list all of the currently open ssh connections on the computer that you run it on:

```
lsof -i -n | egrep '\<ssh\>'
```

Part IX
Copyright

Other articles written by Frank Cox can be found here.

Frank Cox owns and operates the Melville Theatre in Melville, Saskatchewan, Canada, and has been playing with computers for over 30 years.