Mail
archiveIMAP Instructions

First, this is NOT a replacement for imapsync (http://imapsync.lamiral.info/). imapsync is designed for a sync between two IMAP servers. It can be made to do this, but then it is not free, is it. This is strictly designed to move (**move**, not sync) e-mail from an active account to an archive IMAP account. Why would we do this? Read on.

I ran into a problem with one of my clients where their mailbox was getting very full, yet they felt they needed the historical data for a reason.

Some mail readers have a lot of problems with large message stores. For example, many mail readers will download and store all information in your IMAP account on your local machine, even though that is not the reason for IMAP in the first place. And, a few people do not organize their mail into folders (mailboxes), resulting in thousands of e-mails in the INBOX. Some e-mail programs (Outlook for one) indexes every e-mail in the INBOX and keeps the index in memory at all times.

To help with this situation, I wrote archiveIMAP which is designed to be run as a cron job (daily, weekly, monthly, whatever), moving e-mail out of one IMAP account and into another, while maintaining the hierarchy of the folders. This helps in two distinct situations:

A Lot of Mail

Several of my users use an email service that limits the amount of mail which can be stored there. One of them has a 2G limit, with the ability to increase to 10G for additional costs. The 10G is a hard limit.

However, they keep **everything**, so they always scare that limit. The only way (aside from actually **deleting** old mail, gasp) is to move the mail off that server "somewhere else." Fortunately, they have an internal Linux box, so I was able to set up Postfix+Dovecot+roundcube fairly easily. Once that was done, we install this script on that server and set up a cron job to archive mail from the mail service to the archive machine.

Too many messages in the Inbox

Again, a few of my clients do not categorize their mail, ie place it in folders to organize it. The problems with this are two-fold.

1. Some e-mail programs like Outlook index everything in your INBOX and keep that index in memory while Outlook is on. As such, you can see a major difference in the usefulness of your workstation if you have, say, 1000 message in your inbox vs having 100. In some cases, clients have kept up to 8000 messages in their inbox.
2. Disk access on the server is much higher for large numbers of messages. When you start your mail client, the first thing it does is requests a list of all messages, which it then uses to display your mail list. If you have a lot of messages, each time a request like this takes place, the server is having to

(c) 2020 Rod <wiki@linuxservertech.com> | 2020-12-01 22:22
URL: http://wiki.linuxservertech.com//index.php?action=artikel&cat=16&id=216&artlang=en

scan through 1000, or 8000 files instead of 100. Thus, the server is working harder and is less responsive.

So, massive numbers of e-mail messages in the INBOX result in slower responses from the mail server and slower responses from your computer.

Solution

I wrote archiveIMAP for this purpose. archiveIMAP can connect to any IMAP capable server for both its source and destination. In extreme cases, you can build a cheap, slow machine which can contain your archived e-mail, while maintaining a higher end machine for your normal day-to-day.

archiveIMAP is written in Perl and has been tested with dovecot. I intend to test it with courier and Microsoft Exchange (as source only), but that has not happened yet.

In theory, the program complies with RFC3501 (http://tools.ietf.org/html/rfc3501) which is the defining document covering IMAP e-mail protocol.

archiveIMAP requilres the following Perl libraries. In theory, Net::IMAP::Simple (debian package libnet-imap-simple-ssl-perl) is really the only one that is needed. Date::Manip is used by IMAP::Simple to validate the dates, but since those are calculated it should not be needed.

There are two files needed; archiveIMAP (the Perl script) and archiveIMAP.cfg which the script looks for in its own directory. You can download archiveIMAP and a sample configuration file below.

archiveIMAP does not take any parameters from the command line. It simply reads the configuration file, then follows those directions. There are some global variables defined at the top of the program which can be overridden in the configuration file. The configuration file is a Perl sniplett which is executed at run time, after the globals have been initialized. The following globals are defined.

```
# globals
my $CONFIG_FILE_NAME = 'archiveIMAP.cfg';
my $server = 'localhost'; # used if not specified in definition
# added to ignore in definition
my @ignore = ( 'Deleted Messages','Drafts','Junk E-
mail','Junk','Trash' );
# these are folders we may want to copy, but we never want to delete
my @systemFolders = ('Outbox','Sent Items','INBOX','Inbox', 'Sent');
my $deleteEmptyFolders = 1;
my $deleteOnSuccess = 1;

my %accounts;
```


(c) 2020 Rod <wiki@linuxservertech.com> | 2020-12-01 22:22
URL: http://wiki.linuxservertech.com//index.php?action=artikel&cat=16&id=216&artlang=en

Note that, with the exception of @ignore and @systemFolders, these values are used to "fill in the blanks" of any items not configured in the %accounts hash, which is the main driver for the program. @ignore and @systemFolders are appended to the corresponding lists in the individual %accounts entries.

Warning: Do not set deleteOnSuccess to 0 (false) without a good reason. If this is turned off, the message will be copied the first time. When you run the script again, the message will be copied a second time, so you will now have two copies of the message. I had this turned on for testing, but decided someone, somewhere might actually need it so I left it as an option.

Warning: the accounts are stored in a Perl hash. That means the processing order is not defined. The %accounts hash will be processed in whatever order the computer feels like doing at that time. One reason to (theoretically) set deleteOnSuccess off would be in anticipation that a later entry in the hash would not do that. THIS WILL NOT WORK under the current setup. If you need to do something like this, you wold want to change the line (line 271 in v1.0)

foreach my $account (keys %accounts) {

to do some logical order, such as

foreach my $account (sort keys %accounts) {

or something so it would always be run in a specific order.

The %accounts hash is what drives the program, as stated earlier. Create one entry for each account you want to process. archiveIMAP will loop through each element of the %accounts hash and process them. Here is an example of the %accounts hash, taken from the archiveIMAP.cfg.sample file below.

```
%accounts = (
   'unique name for display' => {
                                'source' => { # login information for so
urce
                                   'server'    => 'Source Server Name',
                                   'username'  => 'Source Login Name',
                                   'password'  => 'Source Password',
                                   # you can explicitly tell it what the
 folder separator is
                                   'separator' => '.',
                                   # these folders are NEVER deleted
                                   'systemFolders' => ['Outbox','Sent It
ems','INBOX','Inbox'],

                                },
                                'target' => { # login information for ta
rget
                                   'server'    => 'Target Server Name',
```


(c) 2020 Rod <wiki@linuxservertech.com> | 2020-12-01 22:22

URL: http://wiki.linuxservertech.com//index.php?action=artikel&cat=16&id=216&artlang=en

```
                                'username'  => 'Target Login Name',
                                'password'  => 'Target Password',
                             },
                             'age'    => ceil(365/2), # number of day
s
                             # following is a list of folders (mailbo
xes) to be ignored
                             # is is APPENDED to the global @ignore
                             'ignore' => [ 'ThisIsSpam','ThisIsNOTSpa
m', 'ToProcess' ],
                             'deleteOnSuccess' => 1, # RTFM. Very dan
gerous to say no
                             'deleteEmptyFolders' => 0, # removes any
 totally empty mailboxes
                           },
               );
```

the first line, %accounts = (, and the last line,);, are used to contain the hash. the other lines are one account, and you can have as many of these as you like.

The most basic account entry would be:

```
%accounts = (
   'unique name for display' => {
                             'source' => {
                                'username'  => 'Source Login Name',
                                'password'  => 'Source Password',
                             },
                             'target' => {
                                'username'  => 'Target Login Name',
                                'password'  => 'Target Password',
                             },
                             'age'    => floor(365*8.5), # number of
days
                           },
               );
```

In this case, the source and destination server names would come from the global $server (defaults to localhost), the ignore list would come from @ignore ('Deleted Messages','Drafts','Junk E-mail','Junk','Trash') and we would deleteOnSuccess and deleteEmptyFolders as per the globals (defaults to true for both).

The 'unique name for display' is used just for that; when the program is running, it

(c) 2020 Rod <wiki@linuxservertech.com> | 2020-12-01 22:22
URL: http://wiki.linuxservertech.com//index.php?action=artikel&cat=16&id=216&artlang=en

will display a message to the console telling you what it is working on.

Anyway, if you find it useful, download and use. It is open source, and free to use for commercial or private purposes. Feel free to drop me a line here (in the comments) or by going to http://www.dailydata.net and submitting a message through the interface (for obvious reasons, I do NOT put my e-mail address on public web sites . . . .)

The attached version has been upgraded to v1.01, which contains some bug fixes, a little more error checking, and has been tested against Exchange (you should really explicitly set the separator on that). **Note**: it has been tested against Exchange as a source, not as as target.

Unique solution ID: #1215
Author: Rod
Last update: 2014-08-21 02:49

(c) 2020 Rod <wiki@linuxservertech.com> | 2020-12-01 22:22
URL: http://wiki.linuxservertech.com//index.php?action=artikel&cat=16&id=216&artlang=en