Rocky Enterprise Linux 9.2 Manual Pages on command 'git-imap-send.1'

C:\>man git-imap-send.1

GIT-IMAP-SEND(1) Git Manual GIT-IMAP-SEND(1)

NAME

git-imap-send - Send a collection of patches from stdin to an IMAP folder

SYNOPSIS

git imap-send [-v] [-q] [--[no-]curl]

DESCRIPTION

This command uploads a mailbox generated with git format-patch into an IMAP drafts folder. This allows patches to be sent as other email is when using mail clients that cannot read mailbox files directly. The command also works with any general mailbox in which emails have the fields "From", "Date", and "Subject" in that order.

Typical usage is something like:

git format-patch --signoff --stdout --attach origin | git imap-send

OPTIONS

-v, --verbose

Be verbose.

-q, --quiet

Be quiet.

--curl

Use libcurl to communicate with the IMAP server, unless tunneling into it.

Ignored if Git was built without the USE_CURL_FOR_IMAP_SEND option set.

--no-curl

Talk to the IMAP server using git?s own IMAP routines instead of using libcurl. Ignored if Git was built with the NO_OPENSSL option set.

CONFIGURATION

To use the tool, imap.folder and either imap.tunnel or imap.host must be set to appropriate values.

Variables

imap.folder

The folder to drop the mails into, which is typically the Drafts folder. For example: "INBOX.Drafts", "INBOX/Drafts" or "[Gmail]/Drafts". Required.

imap.tunnel

Command used to setup a tunnel to the IMAP server through which commands will be piped instead of using a direct network connection to the server. Required when imap.host is not set.

imap.host

A URL identifying the server. Use an imap:// prefix for non-secure connections and an imaps:// prefix for secure connections. Ignored when imap.tunnel is set, but required otherwise.

imap.user

The username to use when logging in to the server.

imap.pass

The password to use when logging in to the server.

imap.port

An integer port number to connect to on the server. Defaults to 143 for imap:// hosts and 993 for imaps:// hosts. Ignored when imap.tunnel is set.

imap.sslverify

A boolean to enable/disable verification of the server certificate used by the SSL/TLS connection. Default is true. Ignored when imap.tunnel is set.

imap.preformattedHTML

A boolean to enable/disable the use of html encoding when sending a patch. An html encoded patch will be bracketed with <pre> and have a content type of text/html. Ironically, enabling this option causes Thunderbird to send the patch as a plain/text, format=fixed email. Default is false.

imap.authMethod

Specify authenticate method for authentication with IMAP server. If Git was built with the NO_CURL option, or if your curl version is older than 7.34.0, or if you?re running git-imap-send with the --no-curl option, the only supported method is CRAM-MD5. If this is not set then git imap-send uses the basic IMAP plaintext LOGIN command.

Examples

Using tunnel mode:

```
[imap]
    folder = "INBOX.Drafts"
    tunnel = "ssh -q -C user@example.com /usr/bin/imapd ./Maildir 2> /dev/null"
```

Using direct mode:

```
[imap]
    folder = "INBOX.Drafts"
    host = imap://imap.example.com
    user = bob
    pass = p4ssw0rd
```

Using direct mode with SSL:

```
[imap]
    folder = "INBOX.Drafts"
    host = imaps://imap.example.com
    user = bob
    pass = p4ssw0rd
    port = 123
    sslverify = false
```

EXAMPLES

To submit patches using GMail?s IMAP interface, first, edit your ~/.gitconfig to specify your account settings:

```
[imap]
    folder = "[Gmail]/Drafts"
    host = imaps://imap.gmail.com
    user = user@gmail.com
    port = 993
    sslverify = false
```

You might need to instead use: folder = "[Google Mail]/Drafts" if you get an error that the "Folder doesn?t exist".

Once the commits are ready to be sent, run the following command:

```
$ git format-patch --cover-letter -M --stdout origin/master | git imap-send
```

Just make sure to disable line wrapping in the email client (GMail?s web interface will wrap lines no matter what, so you need to use a real IMAP client).

CAUTION

It is still your responsibility to make sure that the email message sent by your email program meets the standards of your project. Many projects do not like patches to be attached. Some mail agents will transform patches (e.g. wrap lines, send them as format=flowed) in ways that make them fail. You will get angry flames ridiculing you if you don?t check this.

Thunderbird in particular is known to be problematic. Thunderbird users may wish to visit this web page for more information:

http://kb.mozillazine.org/Plain_text_e-mail_-_Thunderbird#Completely_plain_email

SEE ALSO

git-format-patch(1), git-send-email(1), mbox(5)

GIT

Part of the git(1) suite