git-imap-send

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] [--[no-]draft] [(--folder|-f) <folder>] git imap-send --list

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.

`-f` `<folder>`

`--folder=<folder>`
Specify the folder in which the emails have to saved. For example: `--folder=[Gmail]/Drafts` or `-f` `INBOX/Drafts`.

`--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.

`--draft`

`--no-draft`
Mark uploaded messages with the IMAP `Draft` flag. The default is `--no-draft`.

With libcurl, `--draft` requires version 8.13.0 or later. Older libcurl still uploads the message but cannot set the flag.

`--list`
Run the IMAP LIST command to output a list of all the folders present.

CONFIGURATION ----------

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

Everything above this line in this section isn’t included from the linkgit:git-config[1] documentation. The content that follows is the same as what’s found there:

`imap.folder`
The folder to drop the mails into, which is typically the Drafts folder. For example: `INBOX.Drafts`, `INBOX/Drafts` or `[Gmail]/Drafts`. The IMAP folder to interact with MUST be specified; the value of this configuration variable is used as the fallback default value when the `--folder` option is not given.

`imap.tunnel`
Command used to set up 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 the authentication method for authenticating with the 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 methods are `PLAIN`, `CRAM-MD5`, `OAUTHBEARER` and `XOAUTH2`. If this is not set then `git` `imap-send` uses the basic IMAP plaintext `LOGIN` command.

GETTING A LIST OF AVAILABLE FOLDERS ----------

In order to send an email to a specific folder, you need to know the correct name of intended folder in your mailbox. The names like "Junk", "Trash" etc. displayed by various email clients need not be the actual names of the folders stored in the mail server of your email provider.

In order to get the correct folder name to be used with `git` `imap-send`, you can run `git` `imap-send` `--list`. This will display a list of valid folder names. An example of such an output when run on a Gmail account is:

```
* LIST (\HasNoChildren) "/" "INBOX"
* LIST (\HasChildren \Noselect) "/" "[Gmail]"
* LIST (\All \HasNoChildren) "/" "[Gmail]/All Mail"
* LIST (\Drafts \HasNoChildren) "/" "[Gmail]/Drafts"
* LIST (\HasNoChildren \Important) "/" "[Gmail]/Important"
* LIST (\HasNoChildren \Sent) "/" "[Gmail]/Sent Mail"
* LIST (\HasNoChildren \Junk) "/" "[Gmail]/Spam"
* LIST (\Flagged \HasNoChildren) "/" "[Gmail]/Starred"
* LIST (\HasNoChildren \Trash) "/" "[Gmail]/Trash"
```

Here, you can observe that the correct name for the "Junk" folder is `[Gmail]/Spam` and for the "Trash" folder is `[Gmail]/Trash`. Similar logic can be used to determine other folders as well.

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
```

You may want to use `sslVerify=false` while troubleshooting, if you suspect that the reason you are having trouble connecting is because the certificate you use at the private server `example.com` you are trying to set up (or have set up) may not be verified correctly.

Using Gmail's IMAP interface:

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

Gmail does not allow using your regular password for `git` `imap-send`. If you have multi-factor authentication set up on your Gmail account, you can generate an app-specific password for use with `git` `imap-send`. Visit https://security.google.com/settings/security/apppasswords to create it. Alternatively, use OAuth2.0 authentication as described below.

You might need to instead use: `folder` `=` `"[Google` `Mail]/Drafts"` if you get an error that the "Folder doesn't exist". You can also run `git` `imap-send` `--list` to get a list of available folders.

If your Gmail account is set to another language than English, the name of the "Drafts" folder will be localized.

If you want to use OAuth2.0 based authentication, you can specify `OAUTHBEARER` or `XOAUTH2` mechanism in your config. It is more secure than using app-specific passwords, and also does not

enforce the need of having multi-factor authentication. You will have to use an OAuth2.0 access token in place of your password when using this authentication.

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

Using Outlook's IMAP interface:

Unlike Gmail, Outlook only supports OAuth2.0 based authentication. Also, it supports only `XOAUTH2` as the mechanism.

```
[imap]
    folder = "Drafts"
    host = imaps://outlook.office365.com
    user = user@outlook.com
    port = 993
    authmethod = XOAUTH2
```

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).

In case you are using OAuth2.0 authentication, it is easier to use credential helpers to generate tokens. Credential helpers suggested in linkgit:git-send-email[1] can be used for `git imap-send` as well.

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

SEE ALSO -------- linkgit:git-format-patch[1], linkgit:git-send-email[1], `mbox`(5)

GIT --- Part of the linkgit:git[1] suite

□□□ #1
□ □□□ □□□ 2026-10-31 20:11:20 UTC
□ □□□ □□□ 2026-10-31 20:11:20 UTC