FAQ

Release v0.11

The Syncthing Authors

Jan 30, 2022

CONTENTS

GENERAL

1.1 What is Syncthing?

Syncthing is an application that lets you synchronize your files across multiple devices. This means the creation, modification or deletion of files on one machine will automatically be replicated to your other devices. We believe your data is your data alone and you deserve to choose where it is stored. Therefore Syncthing does not upload your data to the cloud but exchanges your data across your machines as soon as they are online at the same time.

1.2 Is it "syncthing", "Syncthing" or "SyncThing"?

It's **Syncthing**, although the command and source repository is spelled `syncthing` so it may be referred to in that way as well. It's definitely not SyncThing, even though the abbreviation `st` is used in some circumstances and file names.

1.3 How does Syncthing differ from BitTorrent Sync?

The two are different and not related. Syncthing and BitTorrent Sync accomplish some of the same things, namely syncing files between two or more computers.

BitTorrent Sync by BitTorrent, Inc is a proprietary peer-to-peer file synchronization tool available for Windows, Mac, Linux, Android, iOS, Windows Phone, Amazon Kindle Fire and BSD.[1] Syncthing is an open source file synchronization tool.

Syncthing uses an open and documented protocol, and likewise the security mechanisms in use are well defined and visible in the source code. BitTorrent Sync uses an undocumented, closed protocol with unknown security properties.

---

[1] http://en.wikipedia.org/wiki/BitTorrent_Sync

TWO

USAGE

2.1 What things are synced?

The following things are *always* synchronized:

- File Contents
- File Modification Times

The following may be synchronized or not, depending:

- File Permissions (When supported by file system. On Windows, only the read only bit is synchronized.)
- Symbolic Links (When supported by the OS. On Windows Vista and up, requires administrator privileges. Links are synced as is and are not followed.)

The following is *not* synchronized;

- File or Directory Owners and Groups (not preserved)
- Directory Modification Times (not preserved)
- Hard Links (followed, not preserved)
- Extended Attributes, Resource Forks (not preserved)
- Windows, POSIX or NFS ACLs (not preserved)
- Devices, FIFOs, and Other Specials (ignored)
- Sparse file sparseness (will become unsparse)

2.2 Is synchronization fast?

Syncthing segments files into pieces, called blocks, to transfer data from one device to another. Therefore, multiple devices can share the synchronization load, in a similar way as the torrent protocol. The more devices you have online (and synchronized), the faster an additional device will receive the data because small blocks will be fetched from all devices in parallel.

Syncthing handles renaming files and updating their metadata in an efficient manner. This means that renaming a large file will not cause a retransmission of that file. Additionally, appending data to existing large files should be handled efficiently as well.

Temporary files are used to store partial data downloaded from other devices. They are automatically removed whenever a file transfer has been completed or after the configured amount of time which is set in the configuration file (24 hours by default).

2.3 Should I keep my device IDs secret?

No. The IDs are not sensitive. Given a device ID it's possible to find the IP address for that node, if global discovery is enabled on it. Knowing the device ID doesn't help you actually establish a connection to that node or get a list of files, etc.

For a connection to be established, both nodes need to know about the other's device ID. It's not possible (in practice) to forge a device ID. (To forge a device ID you need to create a TLS certificate with that specific SHA-256 hash. If you can do that, you can spoof any TLS certificate. The world is your oyster!)

See also:

device-ids

2.4 What if there is a conflict?

Syncthing does recognize conflicts. When a file has been modified on two devices simultaneously, one of the files will be renamed to `<filename>.sync- conflict-<date>-<time>.<ext>`. The device which has the larger value of the first 63 bits for his device ID will have his file marked as the conflicting file. Note that we only create `sync-conflict` files when the actual content differs.

Beware that the `<filename>.sync-conflict-<date>-<time>.<ext>` files are treated as normal files after they are created, so they are propagated between devices. We do this because the conflict is detected and resolved on one device, creating the `sync-conflict` file, but it's just as much of a conflict everywhere else and we don't know which of the conflicting files is the "best" from the user point of view. Moreover, if there's something that automatically causes a conflict on change you'll end up with `sync-conflict-...sync-conflict -...-sync-conflict` files.

2.5 How to configure multiple users on a single machine?

Each user should run their own Syncthing instance. Be aware that you might need to configure ports such that they do not overlap (see the config.xml).

2.6 Is Syncthing my ideal backup application?

No, Syncthing is not a backup application because all changes to your files (modification, deletion, etc) will be propagated to all your devices. You can enable versioning, but we encourage the use of other tools to keep your data safe from your (or our) mistakes.

2.7 Why is there no iOS client?

Alternative implementation Syncthing (using the Syncthing protocol) are being developed at this point in time to enable iOS support. Additionally, it seems that the next version of Go will support the darwin-arm architecture such that we can compile the mainstream code for the iOS platform.

2.8 Why does it use so much CPU?

1. When new or changed files are detected, or Syncthing starts for the first time, your files are hashed using SHA-256.

2. Data that is sent over the network is first compressed and then encrypted using AES-128. When receiving data, it must be decrypted and decompressed.

Hashing, compression and encryption cost CPU time. Also, using the GUI causes a certain amount of CPU usage. Note however that once things are *in sync* CPU usage should be negligible.

2.9 How can I exclude files with brackets ([]) in the name?

The patterns in .stignore are glob patterns, where brackets are used to denote character ranges. That is, the pattern q[abc]x will match the files qax, qbx and qcx.

To match an actual file *called* q[abc]x the pattern needs to "escape" the brackets, like so: q\[abc\]x.

2.10 Why is the setup more complicated than BTSync?

Security over convenience. In Syncthing you have to setup both sides to connect two nodes. An attacker can't do much with a stolen node ID, because you have to add the node on the other side too. You have better control where your files are transferred.

2.11 How do I access the web GUI from another computer?

The default listening address is 127.0.0.1:8384, so you can only access the GUI from the same machine. Change the GUI listen address through the web UI from 127.0.0.1:8384 to 0.0.0.0:8384 or change the config.xml:

```
<gui enabled="true" tls="false">
  <address>127.0.0.1:8384</address>
```

to

```
<gui enabled="true" tls="false">
  <address>0.0.0.0:8384</address>
```

Then the GUI is accessible from everywhere. You should most likely set a password and enable HTTPS now. You can do this from inside the GUI.

If both your computers are Unixy (Linux, Mac, etc) You can also leave the GUI settings at default and use an ssh port forward to access it. For example,

```
$ ssh -L 9090:127.0.0.1:8384 user@othercomputer.example.com
```

will log you into othercomputer.example.com, and present the *remote* Syncthing GUI on http://localhost:9090 on your *local* computer. You should not open more than one Syncthing GUI in a single browser due to conflicting X-CSRFTokens. Any modification will be rejected. See issue #720 to work around this limitation.

The CSRF tokens are stored using cookies. Therefore, if you get the message Syncthing seems to be experiencing a problem processing your request, you should verify the cookie settings of your browser.

2.12 Why do I see Syncthing twice in task manager?

One process manages the other, to capture logs and manage restarts. This makes it easier to handle upgrades from within Syncthing itself, and also ensures that we get a nice log file to help us narrow down the cause for crashes and other bugs.

2.13 Where do Syncthing logs go to?

Syncthing logs to stdout by default. On Windows Syncthing by default also creates `syncthing.log` in Syncthing's home directory (check `-help` to see where that is). Command line option `-logfile` can be used to specify user-defined logfile.

2.14 How do I upgrade Syncthing?

- If automatic upgrades is enabled (which is the default), Syncthing will upgrade itself automatically within 24 hours of a new release.

- The upgrade button appears in the web GUI when a new version has been released. Pressing it will perform an upgrade.

- To force an upgrade from the command line, run `syncthing -upgrade`.

Note that your system should have CA certificates installed which allow a secure connection to GitHub (e.g. FreeBSD requires `sudo pkg install ca_root_nss`). If `curl` or `wget` works with normal HTTPS sites, then so should Syncthing.

2.15 Where do I find the latest release?

We release new versions through GitHub. The latest release is always found on the release page. Unfortunately GitHub does not provide a single URL to automatically download the latest version. We suggest to use the GitHub API at https://api.github.com/repos/syncthing/syncthing/releases/latest and parsing the JSON response.