

Uploading big files > 512MB



- [System configuration](#)
 - [Configuring your Web server](#)
 - [Apache](#)
 - [Apache with mod_fcgid](#)
 - [Apache with mod_proxy_fcgi](#)
 - [nginx](#)
 - [Configuring PHP](#)
 - [Configuring Nextcloud](#)
 - [Adjust chunk size on Nextcloud side](#)
 - [Large file upload on object storage](#)
 - [Federated Cloud Sharing](#)
-

The default maximum file size for uploads is 512MB. You can increase this limit up to what your filesystem and operating system allows. There are certain hard limits that cannot be exceeded:

- < 2GB on 32Bit OS-architecture
- < 2GB with IE6 - IE8
- < 4GB with IE9 - IE11

64-bit filesystems have much higher limits; consult the documentation for your filesystem.



The Nextcloud sync client is not affected by these upload limits as it is uploading files in smaller chunks. See [Client documentation](#) for more information on configuration options.

System configuration

- Make sure that the latest version of PHP is installed
- Disable user quotas, which makes them unlimited
- Your temp file or partition has to be big enough to hold multiple parallel uploads from multiple users; e.g. if the max upload size is 10GB and the average number of users uploading at the same time is 100: temp space has to hold at least 10x100 GB

Configuring your Web server

“ ”

Nextcloud comes with its own `nextcloud/.htaccess` file. Because `php-fpm` can't read PHP settings in `.htaccess` these settings must be set in the `nextcloud/.user.ini` file.

Set the following two parameters inside the corresponding `php.ini` file (see the **Loaded Configuration File** section of [PHP version and information](#) to find your relevant `php.ini` files) :

```
php_value upload_max_filesize 16G
php_value post_max_size 16G
```

The `upload_max_filesize` and `post_max_size` settings may not apply to file uploads through WebDAV single file PUT requests or [Chunked file uploads](#) For those, PHP and webserver timeouts are the limiting factor on the upload size.

Adjust these values for your needs. If you see PHP timeouts in your logfiles, increase the timeout values, which are in seconds:

```
php_value max_input_time 3600
php_value max_execution_time 3600
```

The [mod_reqtimeout](#) Apache module could also stop large uploads from completing. If you're using this module and getting failed uploads of large files either disable it in your Apache config or raise the configured `RequestReadTimeout` timeouts.

There are also several other configuration options in your Web server config which could prevent the upload of larger files. Please see the manual of your Web server for how to configure those values correctly:

Apache

- [LimitRequestBody](#) (In Apache HTTP Server <=2.4.53 this defaulted to unlimited, but now defaults to 1 GiB. The new default limits uploads from non-chunking clients to 1 GiB. If this is a concern in your environment, override the new default by either manually setting it to `0` or to a value similar to that used for your local environment's PHP `upload_max_filesize` / `post_max_size` / `memory_limit` parameters.)
- [SSLRenegBufferSize](#)
- [Timeout](#)

Apache with mod_fcgid

- [FcgidMaxRequestInMem](#)
- [FcgidMaxRequestLen](#)

“ ”

If you are using Apache/2.4 with mod_fcgid, as of February/March 2016, `FcgidMaxRequestInMem` still needs to be significantly increased from its default value to avoid the occurrence of segmentation faults when uploading big files. This is not a regular setting but serves as a workaround for [Apache with mod_fcgid bug #51747](#).

Setting `FcgidMaxRequestInMem` significantly higher than normal may no longer be necessary, once bug #51747 is fixed.

Apache with mod_proxy_fcgi

- [ProxyTimeout](#)

nginx

- [client_max_body_size](#)
- [fastcgi_read_timeout](#) [often the solution to 504 timeouts during `MOVE` transactions that occur even when using chunking]
- [client_body_temp_path](#)



Make sure that `client_body_temp_path` points to a partition with adequate space for your upload file size, and on the same partition as the `upload_tmp_dir` or `tempdirectory` (see below). For optimal performance, place these on a separate hard drive that is dedicated to swap and temp storage.

If your site is behind a nginx frontend (for example a loadbalancer):

By default, downloads will be limited to 1GB due to `proxy_buffering` and `proxy_max_temp_file_size` on the frontend.

- If you can access the frontend's configuration, disable [proxy buffering](#) or increase [proxy_max_temp_file_size](#) from the default 1GB.
- If you do not have access to the frontend, set the [X-Accel-Buffering](#) header to `add_header X-Accel-Buffering no;` on your backend server.

Configuring PHP

If you don't want to use the Nextcloud `.htaccess` or `.user.ini` file, you may configure PHP instead. Make sure to comment out any lines `.htaccess` pertaining to upload size, if you entered any.

If you are running Nextcloud on a 32-bit system, any `open_basedir` directive in your `php.ini` file needs to be commented out.

Set the following two parameters inside `php.ini`, using your own desired file size values:

```
upload_max_filesize = 16G
post_max_size = 16G
```

Tell PHP which temp directory you want it to use:

```
upload_tmp_dir = /var/big_temp_file/
```

Output Buffering must be turned off in `.htaccess` or `.user.ini` or `php.ini`, or PHP will return memory-related errors:

- `output_buffering = 0`

Configuring Nextcloud

As an alternative to the `upload_tmp_dir` of PHP (e.g. if you don't have access to your `php.ini`) you can also configure a temporary location for uploaded files by using the `tempdirectory` setting in your `config.php` (See [Configuration Parameters](#)).

If you have configured the `session_lifetime` setting in your `config.php` (See [Configuration Parameters](#)) file then make sure it is not too low. This setting needs to be configured to at least the time (in seconds) that the longest upload will take. If unsure remove this completely from your configuration to reset it to the default shown in the `config.sample.php`.

Adjust chunk size on Nextcloud side

For upload performance improvements in environments with high upload bandwidth, the server's upload chunk size may be adjusted:

```
sudo -E -u www-data php occ config:system:set --type int --value 20971520
files.chunked_upload.max_size
```

Put in a value in bytes (in this example, 20MB). Set `--value 0` for no chunking at all.

Default is `104857600` (100 MiB).

Large file upload on object storage

[Chunked file uploads](#) do have a larger space consumption on the temporary folder when processing those uploads on object storage as the individual chunks get downloaded from the storage and will be assembled to the actual file on the Nextcloud servers temporary directory. It is recommended to increase the size of your temp directory accordingly and also ensure that request timeouts are high enough for PHP, webserver or any load balancers involved.

“ ”

In more recent versions of Nextcloud Server, when uploading to S3 in *Primary Storage* mode, we use S3 MultipartUpload. This allows chunked upload streaming of the chunks directly to S3 so that the final MOVE request no longer needs to assemble the final file on the Nextcloud server. This requires your `memcache.distributed` to be set to use Redis (or Memcached), otherwise we fall back on the prior behavior which consumes space on the Nextcloud Server for file assembly (as described above).

Federated Cloud Sharing

If you are using [Federated Cloud Sharing](#) and want to share large files, you can increase the timeout values for requests to the federated servers. Therefore, you can set `davstorage.request_timeout` in your `config.php`. The default value is 30 seconds.

📄 #3

📄 cocobook-sync 📄 2026-10-28 16:27:43 UTC

📄 cocobook-sync 📄 2026-10-29 01:09:19 UTC