

# Installation Nextcloud

cloud, Nextcloud



Ce mode opératoire est en cours de rédaction et n'est pas finalisé.

Nextcloud est une application de stockage de fichiers en ligne.

## Installation Nginx

Nous installons Nginx

```
sudo aptitude install nginx
```

## Récupération de Nextcloud

Le dernier paquet Nexcloud se trouve à cette adresse :

<https://nextcloud.com/install/#instructions-server>

Nous récupérons l'archive Nextcloud

```
wget https://download.nextcloud.com/server/releases/latest.zip
```

Nous le décompressons

```
unzip latest.zip
```

Nous le renommons, lui donnons les bons droits et le plaçons dans /var/www

```
mv nextcloud cloud.grohub.org
sudo chown -R www-data: cloud.grohub.org
sudo mv cloud.grohub.org /var/www
```

## Création vHost Nginx

Nous créons le fichier vHost

```
sudo vi /etc/nginx/sites-available/cloud.grohub.org.conf
```

Nous y ajoutons ce contenu

```
upstream php-handler {
```

```
server unix:/var/run/php/nextcloud.sock;
}

# Set the `immutable` cache control options only for assets with a cache
busting `v` argument
map $arg_v $asset_immutable {
    "" "";
    default "immutable";
}

server {
    listen 80;
    server_name cloud.example.com;

    # Prevent nginx HTTP Server Detection
    server_tokens off;

    # Enforce HTTPS
    return 301 https://$server_name$request_uri;
}

server {
    listen 443 ssl http2;
    server_name cloud.grohub.org;

    # Path to the root of your installation
    root /var/www/cloud.grohub.org;

    # Logs
    access_log /var/log/nginx/cloud.grohub.org.access.log;
    error_log /var/log/nginx/cloud.grohub.org.error.log;

    # Use Mozilla's guidelines for SSL/TLS settings
    # https://mozilla.github.io/server-side-tls/ssl-config-generator/
    ssl_certificate /etc/nginx/ssl/cloud.grohub.org.crt;
    ssl_certificate_key /etc/nginx/ssl/cloud.grohub.org.key;

    # Prevent nginx HTTP Server Detection
    server_tokens off;

    # HSTS settings
    # WARNING: Only add the preload option once you read about
    # the consequences in https://hstspreload.org/. This option
    # will add the domain to a hardcoded list that is shipped
    # in all major browsers and getting removed from this list
    # could take several months.
    #add_header Strict-Transport-Security "max-age=15768000;
includeSubDomains; preload" always;
```

```
# set max upload size and increase upload timeout:
client_max_body_size 512M;
client_body_timeout 300s;
fastcgi_buffers 64 4K;

# Enable gzip but do not remove ETag headers
gzip on;
gzip_vary on;
gzip_comp_level 4;
gzip_min_length 256;
gzip_proxied expired no-cache no-store private no_last_modified no_etag
auth;
gzip_types application/atom+xml application/javascript application/json
application/ld+json application/manifest+json application/rss+xml
application/vnd.geo+json application/vnd.ms-fontobject application/wasm
application/x-font-ttf application/x-web-app-manifest+json
application/xhtml+xml application/xml font/opentype image/bmp image/svg+xml
image/x-icon text/cache-manifest text/css text/plain text/vcard
text/vnd.rim.location.xloc text/vtt text/x-component text/x-cross-domain-
policy;

# Pagespeed is not supported by Nextcloud, so if your server is built
# with the `ngx_pagespeed` module, uncomment this line to disable it.
#pagespeed off;

# The settings allows you to optimize the HTTP2 bandwidth.
# See
https://blog.cloudflare.com/delivering-http-2-upload-speed-improvements/
# for tuning hints
client_body_buffer_size 512k;

# HTTP response headers borrowed from Nextcloud `.htaccess`
add_header Referrer-Policy "no-referrer" always;
add_header X-Content-Type-Options "nosniff" always;
add_header X-Download-Options "noopen" always;
add_header X-Frame-Options "SAMEORIGIN" always;
add_header X-Permitted-Cross-Domain-Policies "none" always;
add_header X-Robots-Tag "noindex, nofollow" always;
add_header X-XSS-Protection "1; mode=block" always;

# Remove X-Powered-By, which is an information leak
fastcgi_hide_header X-Powered-By;

# Add .mjs as a file extension for javascript
# Either include it in the default mime.types list
# or include you can include that list explicitly and add the file
extension
# only for Nextcloud like below:
#include mime.types;
#types {
# application/javascript js mjs;
```

```
#}

# Specify how to handle directories -- specifying
`/index.php$request_uri`
# here as the fallback means that Nginx always exhibits the desired
behaviour
# when a client requests a path that corresponds to a directory that
exists
# on the server. In particular, if that directory contains an index.php
file,
# that file is correctly served; if it doesn't, then the request is
passed to
# the front-end controller. This consistent behaviour means that we
don't need
# to specify custom rules for certain paths (e.g. images and other
assets,
# `/updater`, `/ocm-provider`, `/ocs-provider`), and thus
# `try_files $uri $uri/ /index.php$request_uri`
# always provides the desired behaviour.
index index.php index.html /index.php$request_uri;

# Rule borrowed from `.htaccess` to handle Microsoft DAV clients
location = / {
    if ( $http_user_agent ~ ^DavClnt ) {
        return 302 /remote.php/webdav/$is_args$args;
    }
}

location = /robots.txt {
    allow all;
    log_not_found off;
    access_log off;
}

# Make a regex exception for `/.well-known` so that clients can still
# access it despite the existence of the regex rule
# `location ~ /\.(|autotest|...)` which would otherwise handle requests
# for `/.well-known`.
location ^~ /.well-known {
    # The rules in this block are an adaptation of the rules
    # in `.htaccess` that concern `/.well-known`.

    location = /.well-known/carddav { return 301 /remote.php/dav/; }
    location = /.well-known/caldav { return 301 /remote.php/dav/; }

    location /.well-known/acme-challenge { try_files $uri $uri/ =404;
}
    location /.well-known/pki-validation { try_files $uri $uri/ =404;
}
}
```

```
# Let Nextcloud's API for `/.well-known` URIs handle all other
# requests by passing them to the front-end controller.
return 301 /index.php$request_uri;
}

# Rules borrowed from `.htaccess` to hide certain paths from clients
location ~ ^/(?:(?:build|tests|config|lib|3rdparty|templates|data)?(?:$|/))
{ return 404; }
location ~ ^/(?!(?:\.|autotest|occ|issue|indie|db_|console))
{ return 404; }

# Ensure this block, which passes PHP files to the PHP process, is above
the blocks
# which handle static assets (as seen below). If this block is not
declared first,
# then Nginx will encounter an infinite rewriting loop when it prepends
`/index.php`
# to the URI, resulting in a HTTP 500 error response.
location ~ \.php(?:$|/) {
    # Required for legacy support
    rewrite
    ^/(?!(?:index|remote|public|cron|core/ajax/update|status|ocs/v[12]|updater/|
    .+|oc/ms-provider|.+|.+/richdocumentscode/proxy) /index.php$request_uri;

    fastcgi_split_path_info ^(.+?\.php)(/.*)$;
    set $path_info $fastcgi_path_info;

    try_files $fastcgi_script_name =404;

    include fastcgi_params;
    fastcgi_param SCRIPT_FILENAME $document_root$fastcgi_script_name;
    fastcgi_param PATH_INFO $path_info;
    fastcgi_param HTTPS on;

    fastcgi_param modHeadersAvailable true;           # Avoid sending the
security headers twice
    fastcgi_param front_controller_active true;       # Enable pretty urls
    fastcgi_pass php-handler;

    fastcgi_intercept_errors on;
    fastcgi_request_buffering off;

    fastcgi_max_temp_file_size 0;
}

location ~ \.(?:css|js|svg|gif|png|jpg|ico|wasm|tflite|map)$ {
    try_files $uri /index.php$request_uri;
    add_header Cache-Control "public, max-age=15778463,
$asset_immutable";
    access_log off;      # Optional: Don't log access to assets
```

```
location ~ /\.wasm$ {
    default_type application/wasm;
}

location ~ /\.woff2?$ {
    try_files $uri /index.php$request_uri;
    expires 7d;          # Cache-Control policy borrowed from `.htaccess`
    access_log off;     # Optional: Don't log access to assets
}

# Rule borrowed from `.htaccess`
location /remote {
    return 301 /remote.php$request_uri;
}

location / {
    try_files $uri $uri/ /index.php$request_uri;
}
}
```

Nous créons le lien symbolique dans `/etc/nginx/sites-enabled`

```
cd /etc/nginx/sites-enabled
sudo ln -s /etc/nginx/sites-available/cloud.grohub.org.conf
```

Nous testons et rechargeons la configuration

```
sudo nginx -t
sudo systemctl reload nginx
```

## Installation PHP

Nous installon PHP-fpm et les différents modules dont nous aurons besoin

```
sudo aptitude install php-common php-imagick php8.2-apcu php8.2-bcmath
php8.2-bz2 php8.2-cli php8.2-common php8.2-curl php8.2-fpm php8.2-gd php8.2-
gmp php8.2-igbinary php8.2-imagick php8.2-imap php8.2-intl php8.2-mbstring
php8.2-opcache php8.2-pgsql php8.2-readline php8.2-redis php8.2-xml php8.2-
zip
```

## Tuning

## php.ini

Nous réglons le temps maximal d'exécution pour éviter les timeout dans le fichier /etc/php/7.3/fpm/php.ini

```
max_execution_time = 300
```

Nous activons opcode. Toujours dans le fichier /etc/php/7.3/fpm/php.ini, nous recherchons ces clés et enlevons les commentaires ou les activer

```
opcache.enable=1
opcache.interned_strings_buffer=8
opcache.max_accelerated_files=10000
opcache.memory_consumption=128
opcache.save_comments=1
opcache.revalidate_freq=1
```

## PHP pool

Dans le fichier /etc/php/7.3/fpm/pool.d/nextcloud.conf nous saisissons ces éléments

```
[nextcloud]
listen = /run/php/nextcloud.sock

listen.owner = www-data
listen.group = www-data

user = www-data
group = www-data

pm = dynamic
pm.max_children = 120
pm.start_servers = 12
pm.min_spare_servers = 6
pm.max_spare_servers = 18

env[HOSTNAME] = $HOSTNAME
env[PATH] = /usr/local/bin:/usr/bin:/bin
env[TMP] = /tmp
env[TMPDIR] = /tmp
env[TEMP] = /tmp
```

## Liens

- [documentation Nextcloud](#)

Last update: 27/04/2023 08:40 infrastructure:services:cloud:nextcloud:installation <https://wiki.grohub.org/infrastructure/services/cloud/nextcloud/installation?rev=1682584819>

---

From:  
<https://wiki.grohub.org/> - **Grohub wiki**

Permanent link:  
<https://wiki.grohub.org/infrastructure/services/cloud/nextcloud/installation?rev=1682584819>

Last update: **27/04/2023 08:40**

