Install NextCloud on CentOS 8/RHEL 8 with Nginx (LEMP Stack)

This tutorial will be showing you how to install NextCloud on RHEL 8/CentOS 8 with Nginx web server.

What's NextCloud?

NextCloud is a free open-source self-hosted cloud storage solution. It's functionally similar to Dropbox. Proprietary cloud storage solutions (Dropbox, Google Drive, etc) are convenient, but at a price: they can be used to collect personal data because your files are stored on their computers. If you worried about privacy, you can switch to NextCloud, which you can install on your private home server or on a virtual private server (VPS). You can upload your files to your server via NextCloud and then sync those files to your desktop computer, laptop or smart phone. This way you have full control of your data.

NextCloud Features

- Free and open-source
- End-to-end encryption, meaning files can be encrypted on client device before uploaded to the server, so even if someone steals your server, they can not see your files.
- Can be integrated with an online office suite (Collobora, OnlyOffice) so you can create and edit your doc, ppt, xls files directly from NextCloud.
- The app store contains hundreds of apps to extend functionality (like calendar app, notes-taking app, video conferencing app, etc).
- The sync client are available on Linux, MacOS, Windows, iOS and android.

Prerequisites

NextCloud is written in PHP programing language. **To follow this tutorial, you first need to install LEMP stack on RHEL 8/CentOS 8**. If you haven't already done so, please check out the following tutorial.

- How to install LEMP stack on RHEL 8/CentOS 8

You can install NextCloud on your home server or a VPS (virtual private server). You also need a domain name. I registered my domain name from NameCheap because the price is low and they give whois privacy protection free for life. Nextcloud can be installed without a domain name, but it doesn't make sense if you don't encrypt the HTTP connection to prevent snooping. I recommend buying a domain name, if you really want to tinker with server software and use them to the fullest potential.

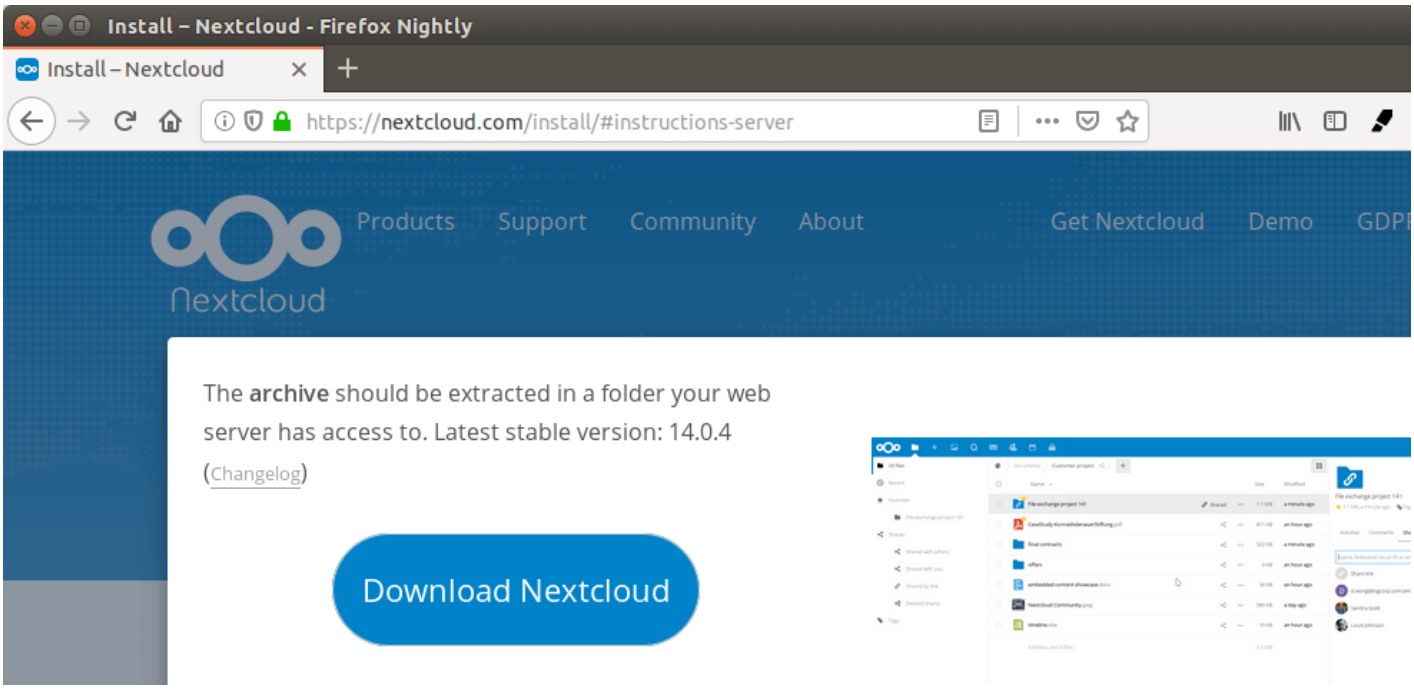
This tutorial uses root account to manage administration tasks. To switch to root, run the following command and enter root password.

```
su -
```

Now let's install NextCloud on the server.

Step 1: Download NextCloud on RHEL 8/CentOS 8 Server

Log into your RHEL 8/CentOS 8 server. Then download the NextCloud zip archive onto your server. The latest stable version is 14.0.4 at time of this writing. You may need to change the version number. Go to https://nextcloud.com/install and click the `download server` button to see the latest version.



You can use the wget tool to download it from command line. The download link is always available in the format below. If a new version comes out, simply replace 14.0.4 with the new version number.

```
yum install wget


wget https://download.nextcloud.com/server/releases/nextcloud-14.0.4.zip
```

Once downloaded, extract the archive with `unzip`.

```
yum install unzip


unzip nextcloud-14.0.4.zip -d /usr/share/nginx/
```

The `-d` option specifies the target directory. NextCloud web files will be extracted to `/usr/share/nginx/nextcloud/`. Then we need to change the owner of this directory to `nginx` so that Nginx web server can write to this directory.

```
chown nginx:nginx /usr/share/nginx/nextcloud/ -R
```

Step 2: Create a Database and User in MariaDB

Log into MariaDB database server with the following command. You will need to enter the MariaDB root password to login.

```
mysql -u root -p
```

Then create a database for Nextcloud. This tutorial name the database `nextcloud`. You can use whatever name you like.

```
CREATE DATABASE nextcloud DEFAULT CHARACTER SET utf8mb4 COLLATE utf8mb4_general_ci;
```

Create the database user. Again, you can use your preferred name for this user. Replace `your-password` with your preferred password.

```
CREATE USER nextclouduser@localhost IDENTIFIED BY 'your-password';
```

Grant this user all privileges on the `nextcloud` database.

```
GRANT ALL PRIVILEGES ON nextcloud.* TO nextclouduser@localhost;
```

Flush privileges and exit.

```
flush privileges;


exit;
```

Step 3: Create a Nginx Config File for Nextcloud

Create a `nextcloud.conf` file in `/etc/nginx/conf.d/` directory. I use the Nano command line text editor in this article.

```
nano /etc/nginx/conf.d/nextcloud.conf
```

Put the following text into the file. Replace the red-colored text with your actual data. In your DNS manager, create a sub-domain for your NextCloud server like `nextcloud.your-domain.com` and don't forget to set A record for the sub-domain.

```
server {
    listen 80;

```

```
    listen [::]:80;
    server_name nextcloud.your-domain.com;

    # Add headers to serve security related headers
    add_header X-Content-Type-Options nosniff;
    add_header X-XSS-Protection "1; mode=block";
    add_header X-Robots-Tag none;
    add_header X-Download-Options noopen;
    add_header X-Permitted-Cross-Domain-Policies none;
    add_header Referrer-Policy no-referrer;

    #I found this header is needed on Debian/Ubuntu/CentOS/RHEL, but not on Arch Linux.
    add_header X-Frame-Options "SAMEORIGIN";

    # Path to the root of your installation
    root /usr/share/nginx/nextcloud/;

    access_log /var/log/nginx/nextcloud.access;
    error_log /var/log/nginx/nextcloud.error;

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

    # The following 2 rules are only needed for the user_webfinger app.
    # Uncomment it if you're planning to use this app.
    #rewrite ^/.well-known/host-meta /public.php?service=host-meta last;
    #rewrite ^/.well-known/host-meta.json /public.php?service=host-meta-json
    # last;

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

    location ~ /.well-known/acme-challenge {
```

```
      allow all;
    }

    # set max upload size
    client_max_body_size 512M;
    fastcgi_buffers 64 4K;

    # Disable gzip to avoid the removal of the ETag header
    gzip off;

    # Uncomment if your server is build with the ngx_pagespeed module
    # This module is currently not supported.
    #pagespeed off;

    error_page 403 /core/templates/403.php;
    error_page 404 /core/templates/404.php;

    location / {
       rewrite ^ /index.php;
    }

    location ~ ^/(?:build|tests|config|lib|3rdparty|templates|data)/ {
       deny all;
    }
    location ~ ^/(?:\.|autotest|occ|issue|indie|db_|console) {
       deny all;
     }

    location ~ ^/(?:index|remote|public|cron|core/ajax/update|status|ocs/v[12]|updater/.+|ocs-
provider/.+|core/templates/40[34])\.php(?:$|/) {
       include fastcgi_params;
       fastcgi_split_path_info ^(.+\.php)(/.*)$;
       try_files $fastcgi_script_name =404;
       fastcgi_param SCRIPT_FILENAME $document_root$fastcgi_script_name;
       fastcgi_param PATH_INFO $fastcgi_path_info;
       #Avoid sending the security headers twice
       fastcgi_param modHeadersAvailable true;
       fastcgi_param front_controller_active true;
       fastcgi_pass unix:/run/php-fpm/www.sock;
       fastcgi_intercept_errors on;
```

```
        fastcgi_request_buffering off;
    }

    location ~ ^/(?:updater|ocs-provider)(?:$|/) {
        try_files $uri/ =404;
        index index.php;
    }

    # Adding the cache control header for js and css files
    # Make sure it is BELOW the PHP block
    location ~* \.(?:css|js)$ {
        try_files $uri /index.php$uri$is_args$args;
        add_header Cache-Control "public, max-age=7200";
        # Add headers to serve security related headers (It is intended to
        # have those duplicated to the ones above)
        add_header X-Content-Type-Options nosniff;
        add_header X-XSS-Protection "1; mode=block";
        add_header X-Robots-Tag none;
        add_header X-Download-Options noopen;
        add_header X-Permitted-Cross-Domain-Policies none;
        # Optional: Don't log access to assets
        access_log off;
    }

    location ~* \.(?:svg|gif|png|html|ttf|woff|ico|jpg|jpeg)$ {
        try_files $uri /index.php$uri$is_args$args;
        # Optional: Don't log access to other assets
        access_log off;
    }
}
```

In nano text editor, press `Ctrl+O` to save the file. Then press Enter to confirm. Press `Ctrl+X` to exit. Then test Nginx configuration.

```
nginx -t
```

If the test is successful, reload Nginx for the changes to take effect.

```
systemctl reload nginx
```

Step 4: Install and Enable PHP Modules

Run the following commands to install PHP modules required or recommended by NextCloud.

```
yum install php-common php-gd php-json php-curl php-zip php-xml php-mbstring php-bz2 php-intl
```

We also need to tell SELinux (Security Enhenced Linux) to allow PHP-FPM to use execmem.

```
setsebool -P httpd_execmem 1
```

Then reload PHP-FPM

```
systemctl reload php-fpm
```

Step 5: Setting up Permissions

First, tell SELinux to allow Nginx and PHP-FPM to read and write to the `/usr/share/nginx/nextcloud/` directory.

```
chcon -t httpd_sys_rw_content_t /usr/share/nginx/nextcloud/ -R
```

By default, SELinux forbids Nginx to make network requests to other servers, but later Nginx needs to request TLS certificate status from Let’s Encrypt CA server, so we need to tell SELinux to allow Nginx with the following command.

```
setsebool -P httpd_can_network_connect 1
```

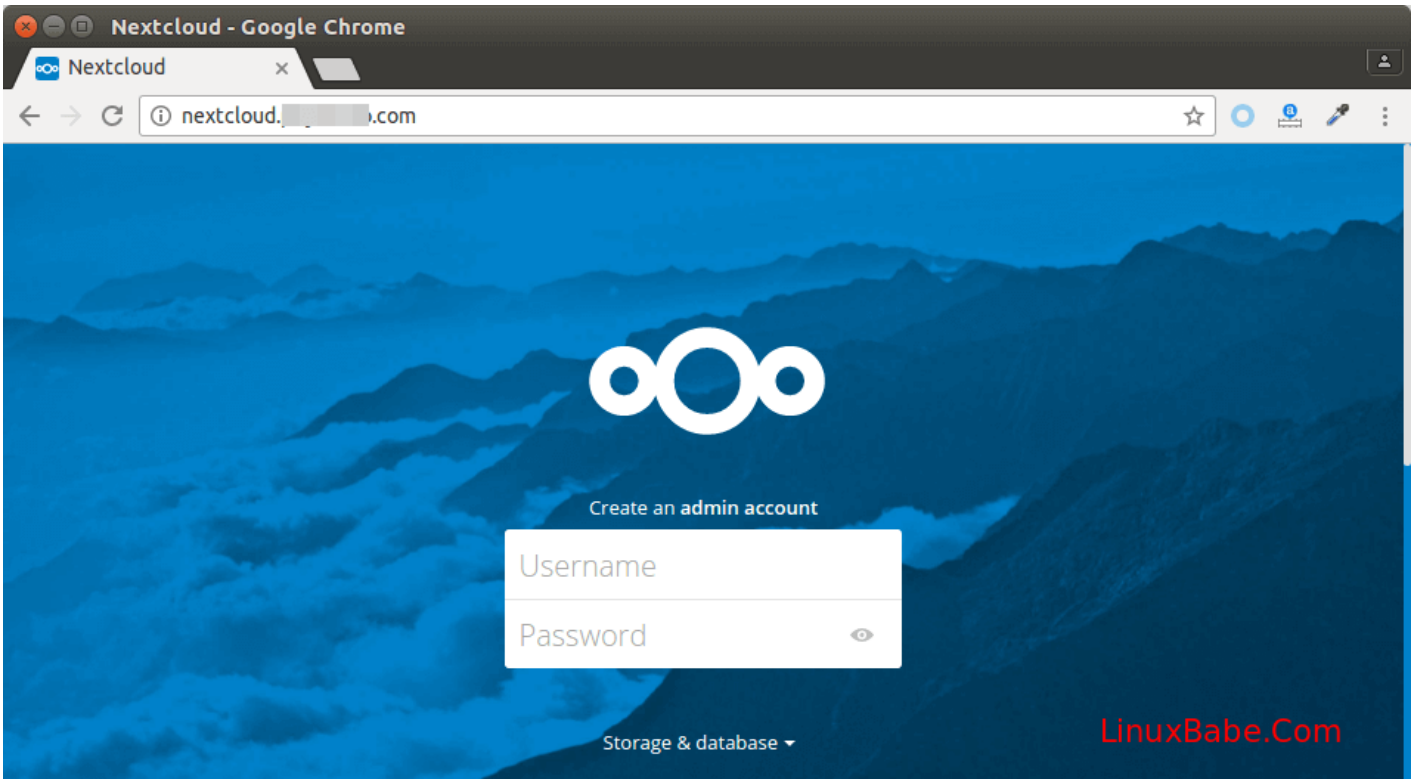
By default, there are 3 files in `/var/lib/php/` directory whose group owner are set to `apache`, but we are using Nginx. So we need to give the nginx user permissions to read and write to the 3 directories with `setfacl`.

```
setfacl -R -m u:nginx:rwx /var/lib/php/opcache/
setfacl -R -m u:nginx:rwx /var/lib/php/session/
setfacl -R -m u:nginx:rwx /var/lib/php/wsdlcache/
```

Step 6: Enable HTTPS

Now you can access the Nextcloud web install wizard in your browser by entering the domain name for your Nextcloud installation.

```
nextcloud.your-domain.com
```



If the web page can't load, you probably need to open port 80 in firewall.

```
firewall-cmd --permanent --zone=public --add-service=http
```

And port 443 as well.

```
firewall-cmd --permanent --zone=public --add-service=https
```

The `--permanent` option will make this firewall rule persistent across system reboots. Next, reload the firewall daemon for the change to take effect.

```
systemctl reload firewalld
```

Now the NextCloud install wizard should be loaded successfully. Before entering any sensitive information, we should enable secure HTTPS connection on Nextcloud. We can obtain a free TLS certificate from Let's Encrypt.

Download Let's Encrypt client `certbot-auto` from EFF website.

```
wget https://dl.eff.org/certbot-auto
```

Give execute permission.

```
chmod a+x certbot-auto
```

Move it to user's PATH, like `/usr/local/bin/` and rename it to `certbot`.

```
sudo mv certbot-auto /usr/local/bin/certbot
```

Set root as the owner and change the permission to 0755.

```
sudo chown root /usr/local/bin/certbot


sudo chmod 0755 /usr/local/bin/certbot
```

Now we can use `certbot` command to obtain a free TLS certificate using the Nginx plugin.

```
sudo /usr/local/bin/certbot --nginx --agree-tos --redirect --hsts --staple-ocsp --email your-email-address -d nextcloud.your-domain.com
```

Where:

- **–nginx**: Use the Nginx authenticator and installer
- **–agree-tos**: Agree to Let's Encrypt terms of service
- **–redirect**: Add 301 redirect so that HTTP requests will be redirected to HTTPS.
- **–hsts**: Add the Strict-Transport-Security header to every HTTP response.
- **–staple-ocsp**: Enables OCSP Stapling to improve performance and user privacy.
- **-d** flag is followed by a list of domain names, separated by comma. You can add up to 100 domain names.
- **–email**: Email used for registration and recovery contact.

If this is a first run on RHEL 8/CentOS 8 system, you may be asked to install some dependency packages. Press `y` to continue.

```
File  Edit  View  Search  Terminal  Help
 libsepol-devel
              x86_64 2.8-1.el8     rhel-8-for-x86_64-baseos-beta-rpms     85 k
 libcom_err-devel
              x86_64 1.44.3-1.el8  rhel-8-for-x86_64-baseos-beta-rpms     37 k
 pcre2-devel   x86_64 10.31-11.el8  rhel-8-for-x86_64-baseos-beta-rpms    591 k
 libkadm5      x86_64 1.16.1-19.el8 rhel-8-for-x86_64-baseos-beta-rpms    184 k
 keyutils-libs-devel
              x86_64 1.5.10-6.el8  rhel-8-for-x86_64-baseos-beta-rpms     48 k
 libselinux-devel
              x86_64 2.8-5.el8     rhel-8-for-x86_64-baseos-beta-rpms    199 k
 zlib-devel    x86_64 1.2.11-10.el8 rhel-8-for-x86_64-baseos-beta-rpms     56 k
 libverto-devel
              x86_64 0.3.0-5.el8   rhel-8-for-x86_64-baseos-beta-rpms     18 k
 pcre2-utf32   x86_64 10.31-11.el8  rhel-8-for-x86_64-baseos-beta-rpms    215 k
Enabling module streams:
 python27             2.7

Transaction Summary
================================================================================
Install  24 Packages

Total download size: 16 M
Installed size: 51 M
Is this ok [y/N]: y
```

You will be asked if you want to receive emails from EFF(Electronic Frontier Foundation). After choosing Y or N, your TLS certificate will be automatically obtained and configured for you, which is indicated by the message below.

```
IMPORTANT NOTES:
 - Congratulations! Your certificate and chain have been saved at:
   /etc/letsencrypt/live/nextcloud.linuxbabe.com/fullchain.pem
   Your key file has been saved at:
   /etc/letsencrypt/live/nextcloud.linuxbabe.com/privkey.pem
   Your cert will expire on 2019-03-06. To obtain a new or tweaked
   version of this certificate in the future, simply run certbot again
   with the "certonly" option. To non-interactively renew *all* of
   your certificates, run "certbot renew"
 - If you like Certbot, please consider supporting our work by:

   Donating to ISRG / Let's Encrypt:   https://letsencrypt.org/donate
   Donating to EFF:                    https://eff.org/donate-le

[root@rhel8 ~]#
```

Finish the Installation in your Web Browser

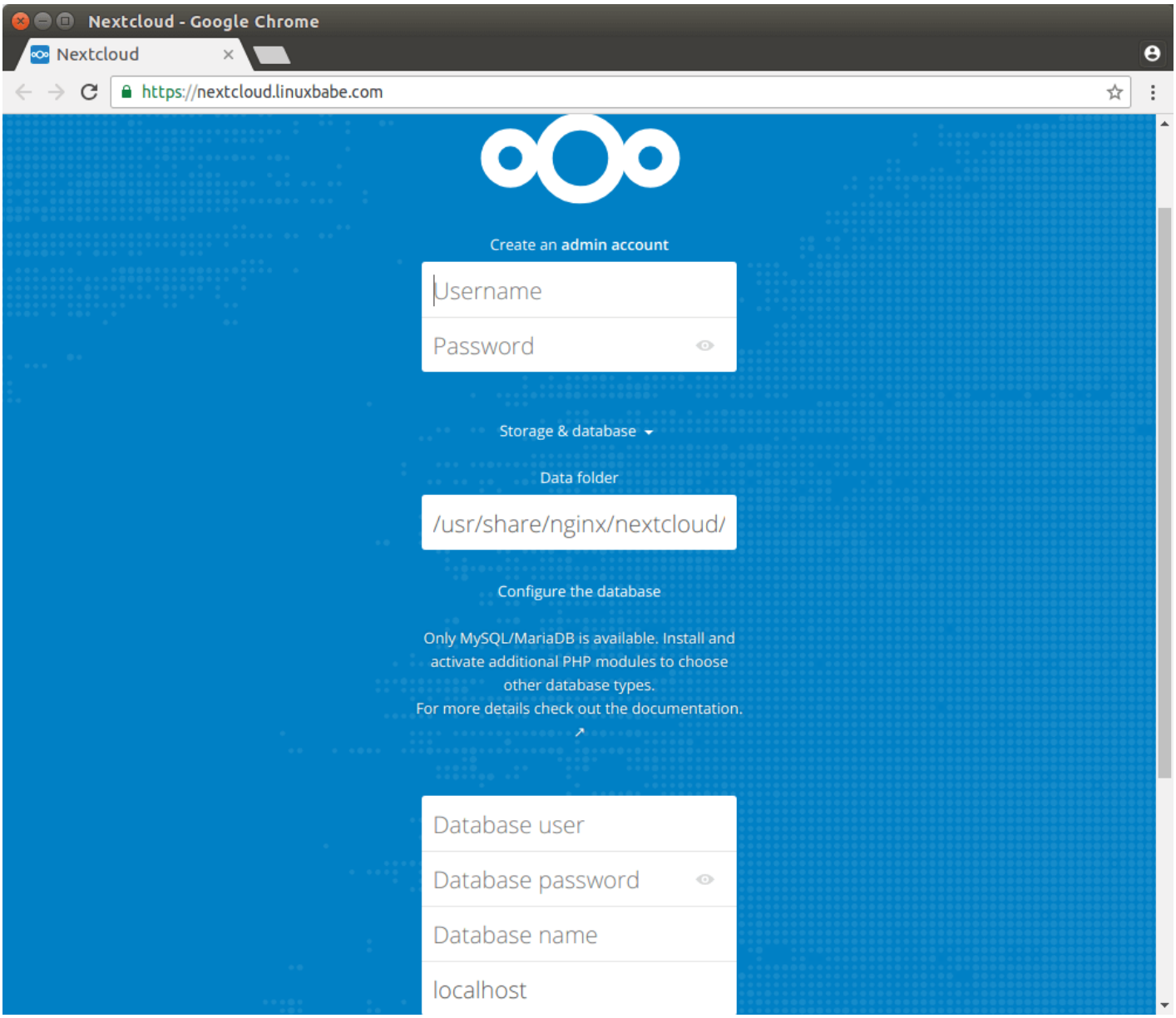
Now you can access the Nextcloud web install wizard using HTTPS connection. To complete the installation, you need to create an admin account, enter the path of Nextcloud data folder, enter database details created earlier. You can use the default `localhost` as host address, or you can enter `localhost:3306`, as MariaDB listens on port 3306.

The data folder is where users' files are stored. For security, it's best to place the data directory outside of Nextcloud web root. So instead of storing users' files under `/usr/share/nginx/nextcloud/data/`, we can change it to **/usr/share/nginx/nextcloud-data**. which can be created with the following command:

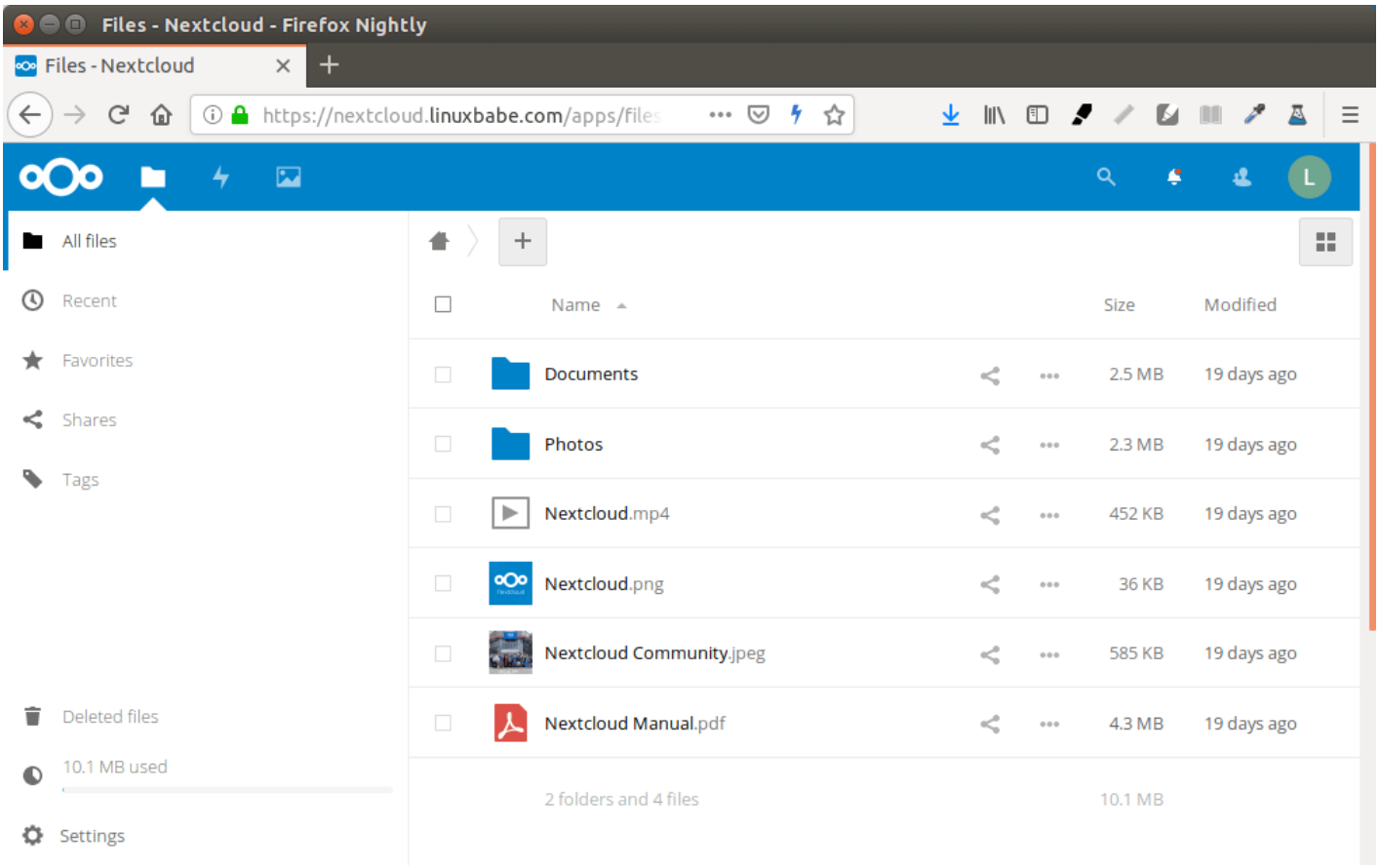
```
mkdir /usr/share/nginx/nextcloud-data
```

Then make sure Nginx user has write permission to the data directory.

```
chown nginx:nginx /usr/share/nginx/nextcloud-data -R
chcon -t httpd_sys_rw_content_t /usr/share/nginx/nextcloud-data/ -R
```



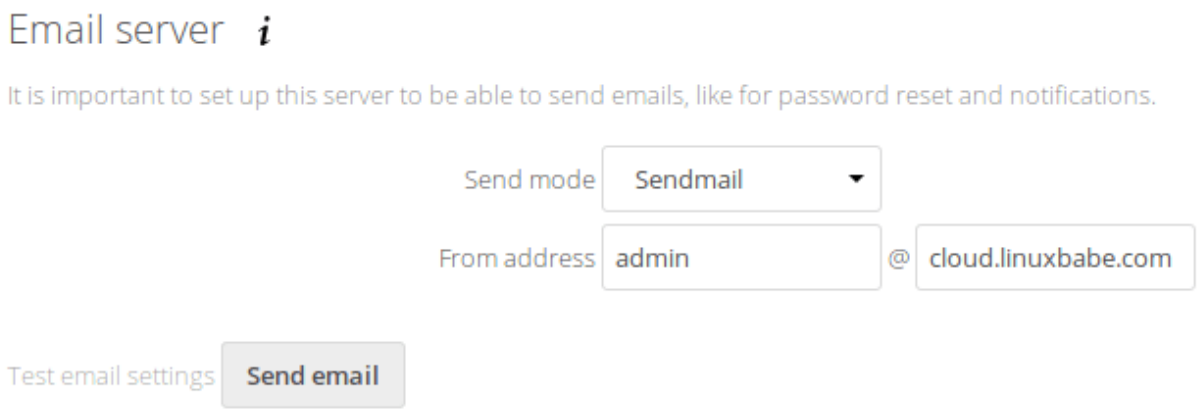
Once it's done, you will see the Web interface of Nextcloud. Congrats! You can start using it as your private cloud storage.



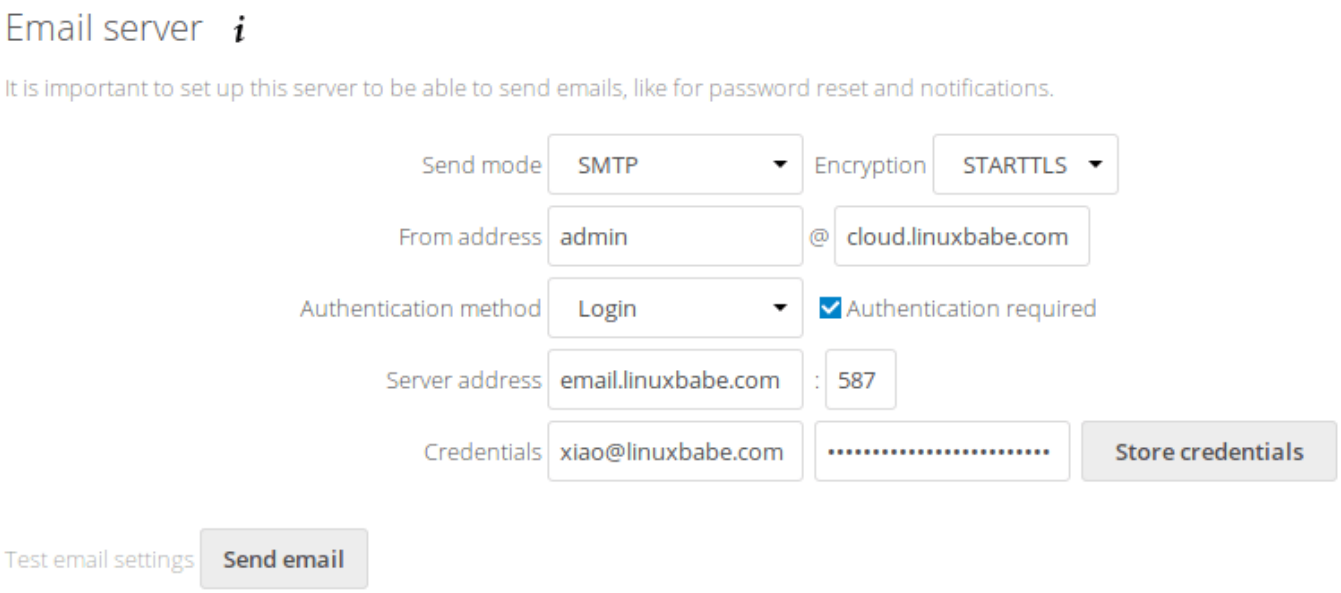
How to Set up NextCloud Email Notification

If your NextCloud instance will be used by more than one person, it's important that your NextCloud server can send transactional emails, such as password-resetting email. To configure email settings, go to **Settings** -> **Basic settings**. You will find the email server settings.

There are two send modes: `sendmail` and `smtp`. The `sendmail` mode is available if your NextCloud host has a SMTP server running.



If you would like to use a SMTP server running on another host, then choose `smtp` mode and enter the login credentials like below.



You also need to tell SELinux to allow Nginx to send mail with the following command.

```
setsebool -P httpd_can_sendmail on
```

For how to set up your own email server, read the following tutorial:

- How to quickly set up a mail server on CentOS with Modoboa

Increase Upload File Size Limit

The default maximum upload file size limit set by Nginx is 1MB. To allow uploading large files to

your NextCloud server, edit the Nginx configuration file for NextCloud.

```
nano /etc/nginx/conf.d/nextcloud.conf
```

We have already set the maximum file size in this file, as indicated by

```
client_max_body_size 512M;
```

You can change it if you prefer, like 1G.

```
client_max_body_size 1024M;
```

Save and close the file. Then reload Nginx for the changes to take effect.

```
systemctl reload nginx
```

PHP also sets a limit of upload file size. The default maximum file size for uploading is 2MB. To increase the upload size limit, edit the PHP configuration file.

```
nano /etc/php.ini
```

Find the following line (line 827).

```
upload_max_filesize = 2M
```

Change the value like below:

```
upload_max_filesize = 1024M
```

Save and close the file. Alternatively, you can run the following command to change the value without manually opening the file.

```
sed -i 's/upload_max_filesize = 2M/upload_max_filesize = 1024M/g'  /etc/php.ini
```

Then restart PHP-FPM.

```
systemctl restart php-fpm
```

Auto-Renew Let’s Encrypt

Certificate

Edit root user’s crontab file.

```
sudo crontab -e
```

Add the following line at the end of the file to run the Cron job daily. If the certificate is going to expire in 30 days, certbot will try to renew the certificate. It’s necessary to reload the Nginx service to pick up new certificate and key file.

```
@daily certbot renew --quiet && systemctl reload nginx
```

Revision #1
Created 11 July 2021 18:48:10 by Admin
Updated 11 July 2021 18:49:01 by Admin