

# Nextcloud

- Nextcloud @Nginx @Oracle Linux v9.5 with self-signed cert (for virtualization)
- Big-Files upload
- Replicate Nextcloud instance to another server
- Encryption

# Nextcloud @Nginx @Oracle Linux v9.5 with self-signed cert (for virtualization) preparations

disable IPv6

```
TODO: sysctl ipv6...
```

```
dnf update
dnf install \
  mariadb \
  wget \
  unzip
```

## Create database

Connect to database server, create database and user for application from wherever it is possible. When safe post-script installation been executed, most probably remote root access is not permitted. Login locally to create new database and user.

```
ssh anton@lt58ncp1dbn1
sudo su
mariadb -u maxscale -p -h 10.120.12.xxx
```

```
CREATE DATABASE ncp1 CHARACTER SET utf8mb4;
CREATE USER 'ncp1rw'@'%' IDENTIFIED BY 'superpass';
GRANT ALL ON ncp1.* TO 'ncp1rw'@'%';
```

```
FLUSH PRIVILEGES;
SHOW GRANTS FOR ncp1rw;
```

```
+-----+
| Grants for ncp1rw@%          |
+-----+
| GRANT USAGE ON *.* TO `ncp1rw`@`%` IDENTIFIED BY PASSWORD
'|BD9925F1D4C650B93F105762F0FC7F494AD66AC8' |
| GRANT ALL PRIVILEGES ON `ncp1`.* TO `ncp1rw`@`%`          |
+-----+
2 rows in set (0.009 sec)
```

Check connectivity to database server from application server

```
[root@lt58ncp1app1 anton]#
mariadb -h lt58ncp1dbn1 -u ncp1rw -p
```

# Generate certificates (self-signed)

Prepare storage

```
export fqdn="host"
export dir="/data/certs/"
mkdir -p ${dir}/CA
mkdir -p ${dir}/${fqdn}
cd ${dir}/CA
pwd
```

Create CSR and certificate

```
TODO: add echoing pass to file and from ${fqdn}.pass
```

create key for CA, give passphrase minimum of four(4) symbols (you want complicated for stronger setups)

```
openssl genrsa -out ca.key 2048
```

create request for CA certificate

```
openssl req -new -sha256 -key ca.key -out ca.csr
```

create a CA certificate

```
openssl x509 -req -days 3600 -in ca.csr -out ca.crt -signkey ca.key
```

generate a key for new certificate for server (modern systems accept key size minimum 2048)

```
openssl genrsa -out server.key 2048
```

create request for the new certificate

```
openssl req -new -sha256 -key server.key -out server.csr
```

sign request for new certificate with the CA

```
openssl x509 -req -sha256 -days 3600 -in server.csr -signkey server.key -out server.crt
```

Now we have pair of certificate and key, we can rename them

```
ls -la

mv ${dir}/CA/server.* ${dir}/${fqdn}/
cd ${dir}/${fqdn}/
mv server.csr ${fqdn}.csr
mv server.crt ${fqdn}.crt
mv server.key ${fqdn}.key
```

Fix SELinux contexts for certificates

```
setenforce 0
```

# Install Nginx webserver

```
dnf install nginx
systemctl enable nginx
systemctl start nginx
```

```
systemctl status nginx
ss -ntap | grep nginx
```

```
LISTEN 0      511      0.0.0.0:80      0.0.0.0:*    users:(("nginx",pid=3459,fd=6),("nginx",pid=3457,fd=6))
LISTEN 0      511      [::]:80        [::]:*      users:(("nginx",pid=3459,fd=7),("nginx",pid=3457,fd=7))
```

### Create firewall rules for webserver

```
firewall-cmd --remove-service=http --permanent
firewall-cmd --add-service=https --permanent
systemctl restart firewalld
firewall-cmd --list-all
```

### Prepare local storage for application (not for data)

```
sudo su
export dir="/var/www/"
mkdir -p ${dir}
cd $dir
pwd
```

Download nextcloud and check integrity. Decide which version is going to be deployed. Rule of thumb is to go current major version minus one.

```
# export v=21
export v=29
wget https://download.nextcloud.com/server/releases/latest-$v.zip
curl https://download.nextcloud.com/server/releases/latest-$v.zip.sha256
sha256sum latest-$v.zip
unzip latest-$v.zip
```

### Change permission to nginx's configuration and applications directories

```
cat /etc/nginx/nginx.conf | grep user
mkdir -p ${dir}/nextcloud/data/
chown -R nginx:nginx ${dir}/nextcloud/data/
chown -R nginx:nginx ${dir}/nextcloud/config/
chown -R nginx:nginx ${dir}/nextcloud/apps/
namei -mo ${dir}/nextcloud/config
```

Disable default config by commenting or deleting the lines

```
vi /etc/nginx/nginx.conf
```

```
# Load modular configuration files from the /etc/nginx/conf.d directory.
# See http://nginx.org/en/docs/nginx_core_module.html#include
# for more information.
include /etc/nginx/conf.d/*.conf;

server {
    listen      80;
    listen     [::]:80;
    server_name _;
    root       /usr/share/nginx/html;

    # Load configuration files for the default server block.
    include /etc/nginx/default.d/*.conf;

    error_page 404 /404.html;
    location = /404.html {
    }

    error_page 500 502 503 504 /50x.html;
    location = /50x.html {
    }
}

# Settings for a TLS enabled server.
#
#   server {
#       listen      443 ssl http2;
#       listen     [::]:443 ssl http2;
#       server_name _;
#       root       /usr/share/nginx/html;
#
#       ssl_certificate "/etc/pki/nginx/server.crt";
#       ssl_certificate_key "/etc/pki/nginx/private/server.key";
#       ssl_session_cache shared:SSL:1m;
#       ssl_session_timeout 10m;
#       ssl_ciphers PROFILE=SYSTEM;
#       ssl_prefer_server_ciphers on;
#
#       # Load configuration files for the default server block.
#       include /etc/nginx/default.d/*.conf;
#
#       error_page 404 /404.html;
#           location = /40x.html {
#       }
#
#       error_page 500 502 503 504 /50x.html;
#           location = /50x.html {
#       }
#   }
}
```

Create nginx configuration file

```
vi /etc/nginx/conf.d/server.conf
```

[https://docs.nextcloud.com/server/latest/admin\\_manual/installation/nginx.html](https://docs.nextcloud.com/server/latest/admin_manual/installation/nginx.html)

Change certificates pointing to the created ones

```
ssl_certificate /data/certs/lt58nct1app21/lt58nct1app21.crt;  
ssl_certificate_key /data/certs/lt58nct1app21/lt58nct1app21.key;
```

Check and reload config

```
nginx -t  
nginx -s reload
```

```
[root@lt58nct1app21 lt58nct1app21]# nginx -t  
nginx: the configuration file /etc/nginx/nginx.conf syntax is ok  
nginx: configuration file /etc/nginx/nginx.conf test is successful  
[root@lt58nct1app21 lt58nct1app21]# nginx -s reload  
[root@lt58nct1app21 lt58nct1app21]# systemctl status nginx  
● nginx.service - The nginx HTTP and reverse proxy server  
   Loaded: loaded (/usr/lib/systemd/system/nginx.service; enabled; preset: disabled)  
   Active: active (running) since Mon 2025-07-21 07:00:24 EEST; 9min ago  
 Main PID: 14512 (nginx)  
   Tasks: 2 (limit: 5835)  
  Memory: 2.9M  
     CPU: 47ms  
   CGroup: /system.slice/nginx.service  
           └─14512 "nginx: master process /usr/sbin/nginx"  
             └─14661 "nginx: worker process"  
  
Jul 21 07:00:24 lt58nct1app21 systemd[1]: Starting The nginx HTTP and reverse proxy server...  
Jul 21 07:00:24 lt58nct1app21 nginx[14510]: nginx: the configuration file /etc/nginx/nginx.conf syntax is ok  
Jul 21 07:00:24 lt58nct1app21 nginx[14510]: nginx: configuration file /etc/nginx/nginx.conf test is successful  
Jul 21 07:00:24 lt58nct1app21 systemd[1]: Started The nginx HTTP and reverse proxy server.
```

Enable latest supported PHP module:

```
dnf module list php  
dnf module enable php:8.3
```

```
[root@lt58nct1app21 lt58nct1app21]# dnf module list php  
Last metadata expiration check: 0:44:11 ago on Mon 21 Jul 2025 06:26:23 AM EEST.  
Oracle Linux 9 Application Stream Packages
```

Name	Stream	Profiles	Summary
php	8.1	common [d], devel, minimal	PHP scripting language
php	8.2	common [d], devel, minimal	PHP scripting language
php	8.3	common [d], devel, minimal	PHP scripting language

Install prerequisites

```
dnf install \  
php \  
php-fpm \  

```

```
php-mysqlnd \  
php-zip \  
php-xml \  
php-mbstring \  
php-curl \  
php-gd
```

## Enable php-fpm service

```
systemctl enable php-fpm  
systemctl start php-fpm  
systemctl status php-fpm
```

Modify webserver config to forward requests to correct socket. Config file comes with php-fpm package, but check and adjust configs:

```
cat /etc/nginx/conf.d/php-fpm.conf
```

## Determine where PHP socket is listening

```
fgrep -irn www.sock /etc/php-fpm.d/
```

```
/etc/php-fpm.d/www.conf:38:listen = /run/php-fpm/www.sock
```

## Reconfigure config

```
vi /etc/nginx/conf.d/host.conf
```

```
upstream php-handler {  
    # server 127.0.0.1:9000;  
    # server unix:/run/php/php8.2-fpm.sock;  
    server unix:/run/php-fpm/www.sock;  
}
```

```
upstream php-handler {  
    # server 127.0.0.1:9000;  
    server unix:/run/php-fpm/www.sock;  
}
```

Enable logging: check log file is created after reloading nextcloud page and it can be tail'ed

```
mkdir -p /var/www/nextcloud/log
chown -R www-data:www-data /var/www/nextcloud/log/
vi /var/www/nextcloud/config/config.php
ls -la /var/www/nextcloud/log/

tail -f /var/www/nextcloud/log/nextcloud.log
```

# Big-Files upload

To enable big files management, there are several components need to be fine-tuned.

Good practise is to backup configuration file by copying and renaming it by adding timestamp prefix. Also, duplicate the line to be modified and add timestamped and signed comment.

## Prepare location for temporary storage

Ensure, disk used has enough storage to allocate for temporary file (if limit of the big file is 16GB, it must be 16GB (per one upload process, if two users upload two files, it should be double).

```
export host="$(hostname)"
export dir="/mnt/${host}-data/ncp1/tmp/"
mkdir -p ${dir}/php-fpm/
mkdir -p ${dir}/nginx/
df -h ${dir}
chmod 775 ${dir}/php-fpm/
chmod 775 ${dir}/nginx/
chown -R www-data:www-data ${dir}

ls -latr ${dir}
```

First, let's see current settings, which will be modified, note them, adjust values and extract them again. By doing so, we ensure that new values are applied. This facilitate troubleshooting.

## PHP (php-fpm)

Initially, it is important to understand that 'php' and 'php-fpm' are to different packages AND configuration files are different for them as well.

To show PHP related info is to enable show php button in the admin page. The reason to use it that exactly like that PHP modules and versions are seen from application perspective. System might have packages installed, but for some reason are not seen by application. But..

```
root@(host):/var/www/nextcloud# sudo -u www-data php ./occ config:app:set --value=yes serverinfo phpinfo
```

Config value 'phpinfo' for app 'serverinfo' is now set to 'yes', stored as mixed in fast cache

The PHP info will be displayed on admin page, navigate to:

```
https://(nextcloud)/settings/admin/serverinfo
```

Click on "Show phpinfo()". Search for (CTRL+F or CMD+F in Firefox) "configuration file", which will indicate which file to modify. In my case, it is

```
Configuration File (php.ini) Path /etc/php/8.3/fpm
Loaded Configuration File      /etc/php/8.3/fpm/php.ini
```

We are interested in values defining max filesize and timeouts (in seconds)

Let's modify these values. Backup first, then load editor

```
file="/etc/php/8.3/fpm/php.ini"
cp ${file} ${file}.${date +"%Y-%m-%d.%H%M"}
ls -latr ${file}*

vi ${file}
```

Duplicate, modify, comment and save file. Final result and documenting style should be as below

```
# max_execution_time = 30
# 2025-03-23 * for big files /A
max_execution_time = 3600

# max_input_time = 60
# 2025-03-23 * for big files /A
max_input_time = 60

# memory_limit = 128M
# 2025-03-23 * for big files /A
memory_limit = 1G
```

```
# post_max_size = 8M
# 2025-03-23 * for big files /A
post_max_size = 16G

# a;upload_tmp_dir =
# 2025-03-23 * for big files /A
upload_tmp_dir = /mnt/gcp1ncp1app1-data/ncp1/tmp/php-fpm/

# upload_max_filesize = 2M
# 2025-03-23 * for big files /A
upload_max_filesize = 16G
```

Restart php-fpm

```
systemctl restart php8.3-fpm
```

# Webserver (nginx):

```
root@(host):/home/anton# vi /etc/nginx/sites-enabled/(host).conf
```

```
# 2025-03-23 * for big files /A
# client_max_body_size 512M;
client_max_body_size 16G;
# client_body_timeout 300s;
client_body_timeout 3600s;

# 2025-03-23 * for big files /A
client_body_temp_path /mnt/gcp1ncp1app1-data/ncp1/tmp/nginx/;
```

# Application

Adjusting chunk size (use smaller chunks for higher bandwidth). In this example it will be set to 20 MB, default is 100 MB. `--value 0` for no chunking.

```
cd /var/www/nextcloud/
sudo -u www-data php occ config:system:get files.chunked_upload.max_size
```

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

System config value files.chunked\_upload.max\_size set to integer 20971520

```
sudo -u www-data php occ config:system:get files.chunked_upload.max_size
```

20971520

ref.

[https://docs.nextcloud.com/server/latest/admin\\_manual/configuration\\_files/big\\_file\\_upload\\_configuration.html](https://docs.nextcloud.com/server/latest/admin_manual/configuration_files/big_file_upload_configuration.html)

# Replicate Nextcloud instance to another server

## Destination:

Prepare

```
export user="user"
export src="(source host)"
export dst="(destination host)"
export path="/home/${user}/delme/${src}--${dst}/"
mkdir -p ${path}
chown -R ${user}:${user} /home/${user}/delme/
ls -la ${path}
namei -mo ${path}
```

Clean before bringing new stuff here (carefully here)

```
echo ${path}
ls -la ${path}
read -p "Correct path to clean?"
rm -i ${path}/*
ls -la ${path}
```

## Source

Prepare

```
sudo su
export user="user"
export src="(source host)"
export dst="(destination host)"
```

```
export path="/home/${user}/delme/${src}--${dst}/"
export file="nextcloud.tgz.${date +"%Y-%m-%d.%H%M"}"
mkdir -p ${path}
ls -lahtr ${path}
```

Compress application directory (takes little time, coffee time)

```
tar -czvf ${path}/${file} /var/www/nextcloud/
ls -lahtr ${path}
```

Transfer file to the second instance

```
scp -v ${path}/${file} ${user}@${dst}:delme/${src}--${dst}/
```

## Destination (continues):

Observe, that has been transferred correctly

```
ls -lahtr ${path}
```

Move aside current Nextcloud application instance before bring new one.

```
export dir_current="nextcloud.${date +"%Y-%m-%d.%H%M"}"
echo ${dir_current}
mv /var/www/nextcloud /var/www/${dir_current}
ls -latr /var/www
```

Extract it

```
# that is number one, not L letter
export file=$(ls -l ${path}/nextcloud*)
echo ${file}
tar -xvzf ${file} -C /
```

After successful testing, remove old version, as they might occupy disk space.

```
ls -dlr /var/www/nextcloud*
du -h --max-depth=1 /var/www
```

# Encryption

Enable encryption app (functionality)

```
sudo -u www-data php /var/www/nextcloud/occ app:enable encryption
```

Enable encryption

```
sudo -u www-data php /var/www/nextcloud/occ encryption:enable
```

Enable master key mode

```
sudo -u www-data php /var/www/nextcloud/occ encryption:enable-master-key
```

Initialize encryption

```
sudo -u www-data php /var/www/nextcloud/occ encryption:init
```

Verify status

```
sudo -u www-data php /var/www/nextcloud/occ encryption:status
```

Enable a recovery key

```
sudo -u www-data php /var/www/nextcloud/occ encryption:enable-recovery
```

Enable encryption for group folders

By default group folders are excluded from server-side encryption. To enable encryption for group folders, execute command

```
./occ config:app:set groupfolders enable_encryption --value='true'
```

Restart scheduled jobs

```
systemctl restart nextcloud-cron  
systemctl restart cron
```

Verify

```
ls -lh /var/www/nextcloud/data/<username>/files/  
cat /var/www/nextcloud/data/<username>/files/<file>
```