

Nextcloud 21 Installationsanleitung v. 3.0

von Carsten Rieger | Aktualisiert



Dieser Artikel beschreibt die Installation, Konfiguration und Härtung, das Monitoring sowie einige Erweiterungsmöglichkeiten von Nextcloud auf einem Ubuntu Server 20.04.2 LTS Focal Fossa bzw. Debian Server 10.9 Buster. Die Installation basiert dabei auf den Komponenten nginx 1.19.x mainline, Let's Encrypt TLS 1.3, MariaDB 10.5.x, PHP 8.x, Redis, Fail2ban, ufw sowie Netdata und erhält abschließend sowohl von Nextcloud, als auch von Qualys SSL Labs eine A+ Sicherheitsbewertung. Im Verlauf dieser Anleitung müssen Sie nur die **rot** markierten Werte wie bspw. **ihre.domain.de** oder **192.168.2.x** mit den entsprechenden Werten Ihres Systems ersetzen.

Möchten Sie lieber alles mit nur einem einzigen Skript installieren? Auch kein Problem, folgen Sie diesem [Link](#) und nutzen Sie mein Installationsskript für Nextcloud. Es enthält die Punkte 1 bis 6 dieser Installationsanleitung.

Letzte Aktualisierung: Samstag, 10. April 2021

Die ausführliche Aktualisierungshistorie finden Sie [hier](#).

Inhaltsverzeichnis

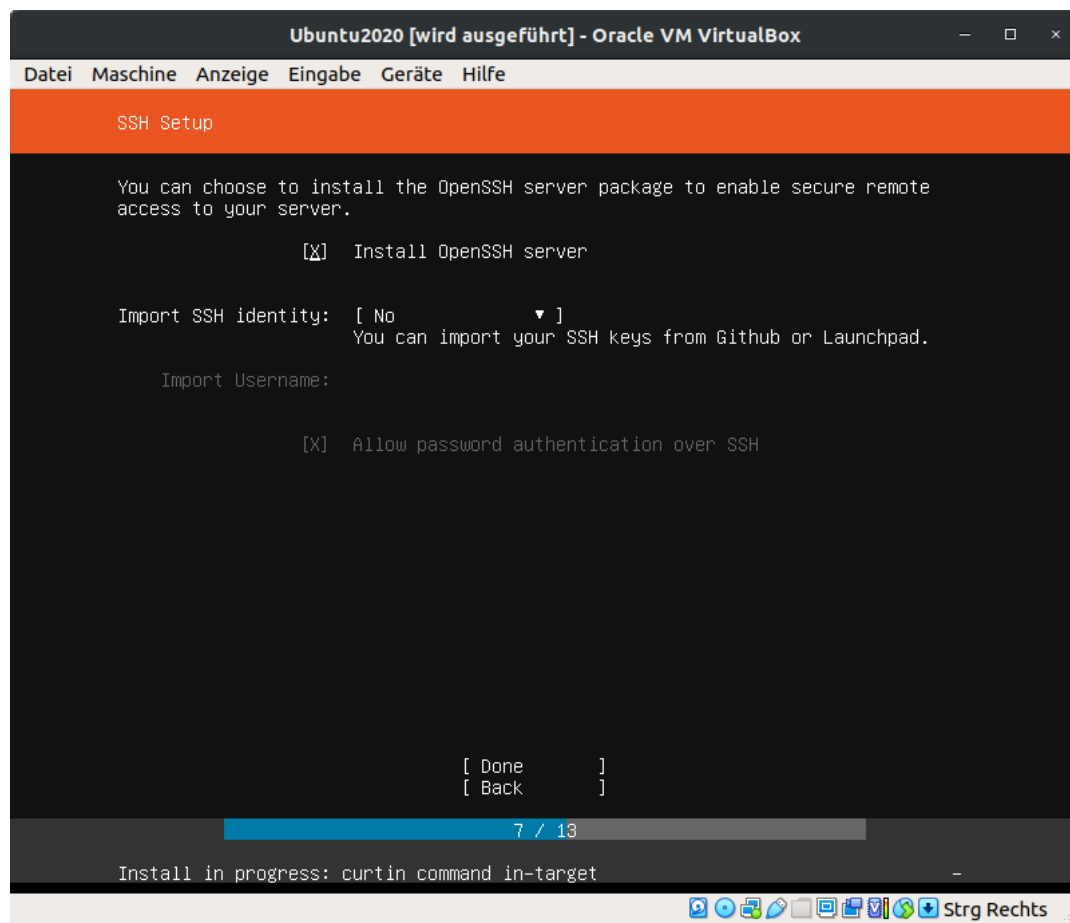
1. [Vorbereitungen und Installation des nginx Webservers](#)
2. [Installation und Konfiguration von PHP 8.0](#)
3. [Installation und Konfiguration des Datenbankservers MariaDB 10.5](#)
4. [Installation des Redis-server \(„in-memory-Datenbank“\)](#)
5. [Installation und Optimierung der Nextcloud \(inkl. SSL\)](#)
6. [Systemhärtung \(fail2ban und ufw\)](#)
7. [Systemmails per postfix](#)
 - [fail2ban Benachrichtigungen](#)
 - [apicron Benachrichtigungen](#)
 - [ssh Benachrichtigungen](#)
8. [Optimieren und aktualisieren der Nextcloud per Skript](#)
9. **optional:** [Systemüberwachung mit netdata](#)
10. **optional:** [Nextcloud Speicher erweitern/verschieben](#)
 - [mittels NAS](#)
 - [mittels einer weiteren HDD/SSD](#)
 - [mittels der Nextcloud external storage app](#)
11. **optional:** [Nextcloud High Performance Backend für Dateien](#)

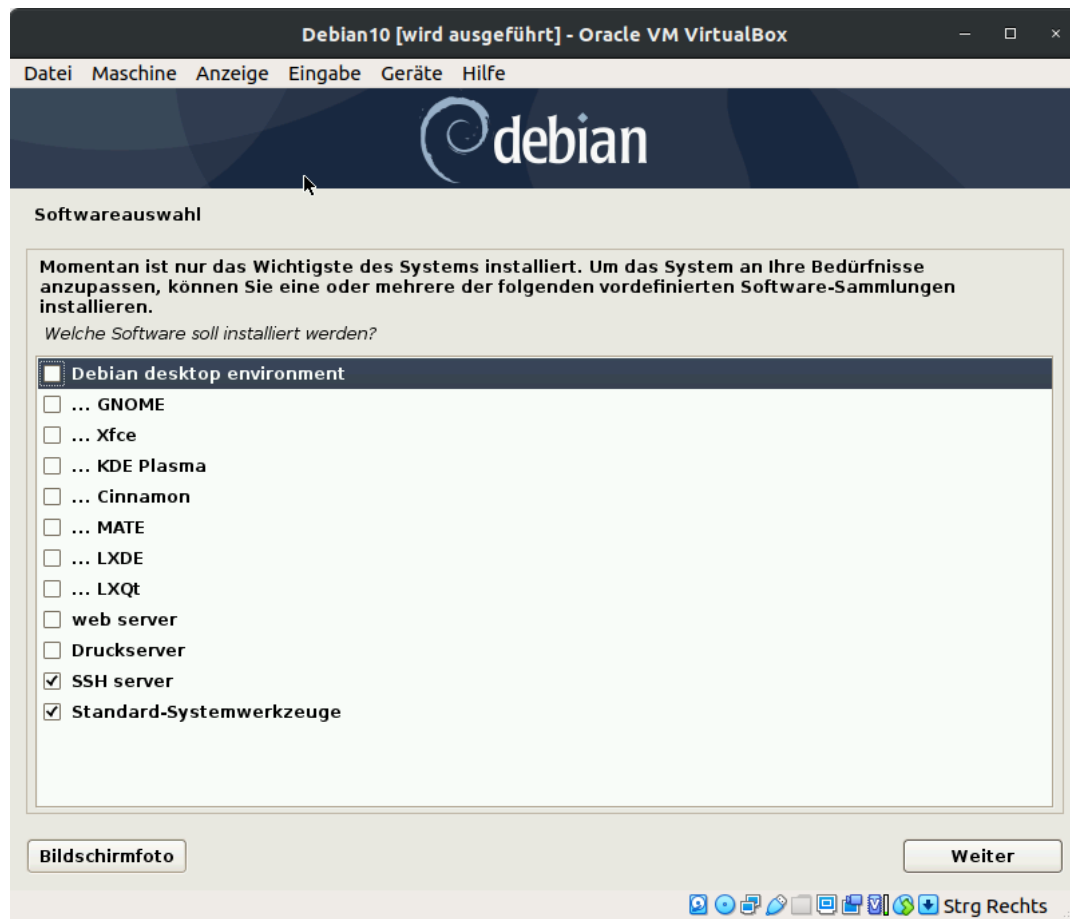
Systemvoraussetzungen seitens Nextcloud

https://docs.nextcloud.com/server/latest/admin_manual/installation/system_requirements.html#server

1. Vorbereitungen und Installation des nginx Webserver

Die Installationsmedien für den zugrundeliegenden Linux-Server erhalten Sie hier:





Ubuntu 20.04.x LTS: Voraussetzungen

[Download-Installationsmedium](#) - SSH wird vorausgesetzt (s. Bild zuvor)

Debian 10.9.x: Voraussetzungen

[Download-Installationsmedium](#) - Standardsystemwerkzeuge u. SSH werden vorausgesetzt (s. Bild zuvor)

su -

apt install -y sudo

usermod -aG sudo <aktueller Benutzer>

exit

Debian und Ubuntu Server:

Wechseln Sie in den privilegierten Benutzermodus

sudo -s

um die folgenden Softwarepakete, als notwendige Grundlage des Serverbetriebs, zu installieren:

```
apt install -y curl gnupg2 git lsb-release ssl-cert ca-certificates apt-transport-https tree locate software-properties-common  
dirmngr screen htop net-tools zip unzip bzip2 ffmpeg ghostscript libfile-fcntllock-perl libfontconfig1 libfuse2 socat
```

Tragen Sie den zukünftigen Servernamen sowohl in die hosts-Datei, als auch in die hostname-Datei ein:

```
vi /etc/hosts
```

Passen Sie die roten Werte an Ihre Umgebung an:

```
127.0.0.1 localhost
```

```
127.0.1.1 ihre.domain.de domain
```

```
# The following lines are desirable for IPv6 capable hosts
```

```
::1 ip6-localhost ip6-loopback
```

```
fe00::0 ip6-localnet
```

```
ff00::0 ip6-mcastprefix
```

```
ff02::1 ip6-allnodes
```

```
ff02::2 ip6-allrouters
```

<externe IP> ihre.domain.de

Geben Sie den korrekten Servername in der hostname-Datei an und ersetzen den roten Wert durch Ihren:

```
vi /etc/hostname
```

Der Servername muss als FQDN, also vollqualifiziert angegeben werden:

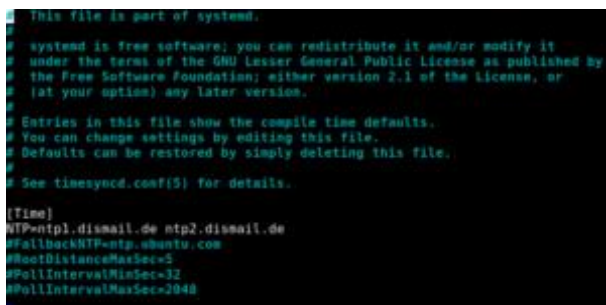
ihre.domain.de

Überprüfen Sie, ob der Zeitserverdienst mit mindestens einem Endpunkt konfiguriert ist.

```
nano /etc/systemd/timesyncd.conf
```

Ist die Zeile NTP auskommentiert (#NTP=), so entfernen Sie das '#'-Zeichen vor NTP und fügen Sie bspw. diese zwei Zeitserver hinzu:

```
NTP=ntp1.dismail.de ntp2.dismail.de
```



```
This file is part of systemd.
#
# systemd is free software; you can redistribute it and/or modify it
# under the terms of the GNU Lesser General Public License as published by
# the Free Software Foundation; either version 2.1 of the license, or
# (at your option) any later version.
#
# Entries in this file show the compile time defaults.
# You can change settings by editing this file.
# Defaults can be restored by simply deleting this file.
# See timesyncd.conf(5) for details.
#
[Time]
NTP=ntp1.dismail.de ntp2.dismail.de
FallbackNTP=ntp.ubuntu.com
RootDistanceMaxSec=5
PollIntervalMinSec=32
PollIntervalMaxSec=2048
```

Speichern Sie diese Datei und starten den Zeitserver neu:

```
systemctl restart systemd-timesyncd
```

Starten Sie den Server neu

```
reboot now
```

und melden sich dann erneut mit privilegierten Benutzerrechten am Server an:

```
sudo -s
```

Fügen Sie dem System weitere Software-Repositories (Softwarequellen) hinzu, um die **aktuellen** Releases der jeweiligen Pakete installieren zu können. Wechseln Sie in das folgende Verzeichnis:

```
cd /etc/apt/sources.list.d
```

Nur Ubuntu Server (AMD64 und ARM64):

Korrigieren Sie, sofern notwendig, zuerst die DNS-Auflösung:

```
rm -f /etc/resolv.conf
```

```
ln -sf /run/systemd/resolve/resolv.conf /etc/resolv.conf
```

```
systemctl restart systemd-resolved.service
```

Fügen Sie dann die Softwarequellen für nginx, PHP und MariaDB hinzu:

```
echo "deb http://nginx.org/packages/mainline/ubuntu $(lsb_release -cs) nginx" | tee nginx.list
```

```
echo "deb http://ppa.launchpad.net/ondrej/php/ubuntu $(lsb_release -cs) main" | tee php.list
echo "deb http://ftp.hosteurope.de/mirror/mariadb.org/repo/10.5/ubuntu $(lsb_release -cs) main" | tee mariadb.list
```

Um diesen Quellen vertrauen zu können nutzen wir die entsprechenden Schlüssel:

PHP-Key:

```
apt-key adv --recv-keys --keyserver hks://keyserver.ubuntu.com:443 4F4EA0AAE5267A6C
```

Nur Debian Server (AMD64):

Fügen Sie die Softwarequellen für nginx, PHP und MariaDB hinzu:

```
echo "deb [arch=amd64] http://nginx.org/packages/mainline/debian $(lsb_release -cs) nginx" | tee nginx.list
echo "deb [arch=amd64] https://packages.sury.org/php/ $(lsb_release -cs) main" | tee php.list
echo "deb [arch=amd64] http://mirror2.hs-esslingen.de/mariadb/repo/10.5/debian $(lsb_release -cs) main" | tee mariadb.list
```

Um den jeweiligen Quellen vertrauen zu können nutzen wir die entsprechenden Schlüssel:

PHP-Key:

```
wget -q https://packages.sury.org/php/apt.gpg -O- | apt-key add -
```

Ab hier geht es wieder für beide Server-Betriebssysteme (Ubuntu und Debian) weiter:

Wir ergänzen die noch fehlenden Schlüssel für nginx und MariaDB, aktualisieren das System und generieren im Anschluß daran sogenannte self-signed-Zertifikate, die im weiteren Verlauf durch vollwertige Zertifikate von Let's Encrypt ersetzt werden.

NGINX-Key:

```
curl -fsSL https://nginx.org/keys/nginx_signing.key | apt-key add -
```

MariaDB-Key:

```
apt-key adv --recv-keys --keyserver hks://keyserver.ubuntu.com:443 0xF1656F24C74CD1D8
apt update && apt upgrade -y
make-ssl-cert generate-default-snakeoil -y
```

Um sicherzustellen, dass keine Relikte früherer Installationen den Betrieb des Webserver stören, entfernen wir diese:

```
apt remove nginx nginx-extras nginx-common nginx-full -y --allow-change-held-packages
```

Zudem stellen wir sicher, dass das Pendant (Apache2) zum nginx Webserver weder aktiv noch installiert ist.

```
systemctl stop apache2.service && systemctl disable apache2.service
```

Nun sind die Vorbereitungen zur Installation des Webserver abgeschlossen und wir können diesen mit dem nachfolgenden Befehl installieren

```
apt install nginx -y
```

und den Dienst zum automatischen Start nach einem Systemneustart mittels

```
systemctl enable nginx.service
```

einrichten. Mit Blick auf die späteren Anpassungen wird die Standardkonfiguration gesichert und eine neue Konfigurationsdatei geöffnet:

```
mv /etc/nginx/nginx.conf /etc/nginx/nginx.conf.bak && touch /etc/nginx/nginx.conf
nano /etc/nginx/nginx.conf
```

Kopieren Sie den gesamten nachfolgenden Inhalt in die Datei und ersetzen die **rot** markierten Werte entsprechend Ihres Systems:

```
user www-data;
worker_processes auto;
pid /var/run/nginx.pid;
events {
worker_connections 1024;
multi_accept on; use epoll;
}
http {
server_names_hash_bucket_size 64;
access_log /var/log/nginx/access.log;
error_log /var/log/nginx/error.log warn;
set_real_ip_from 127.0.0.1;
#optional, Sie können das eigene Subnetz ergänzen, bspw.:
# set_real_ip_from 192.168.2.0/24;
real_ip_header X-Forwarded-For;
real_ip_recursive on;
include /etc/nginx/mime.types;
default_type application/octet-stream;
sendfile on;
send_timeout 3600;
tcp_nopush on;
tcp_nodelay on;
open_file_cache max=500 inactive=10m;
open_file_cache_errors on;
keepalive_timeout 65;
reset_timedout_connection on;
server_tokens off;
resolver 127.0.0.53 valid=30s;
resolver_timeout 5s;
include /etc/nginx/conf.d/*.conf;
}
```

Speichern Sie die Datei und schließen Sie diese, um im Anschluß den Webserver neu zu starten:

```
service nginx restart
```

Vorbereitend für die SSL Zertifikate und die Webverzeichnisse legen wir vier Ordner an und setzen die korrekten Berechtigungen:

```
mkdir -p /var/nc_data /var/www/letsencrypt/.well-known/acme-challenge /etc/letsencrypt/rsa-certs /etc/letsencrypt/ecc-certs
```

```
chown -R www-data:www-data /var/nc_data /var/www
```

Die Installation des Webserver ist somit bereits abgeschlossen und wir fahren mit der Installation und den Anpassungen von PHP fort.

2. Installation und Konfiguration von PHP 8.0 (fpm)

Das PHP Repository wurde bereits im vorherigen Kapitel eingerichtet und aktiviert, so dass wir direkt mit der Installation beginnen können. Haben Sie Interesse an PHP 8.x, so lesen Sie bitte [diesen Artikel](#). Für größere, produktive Umgebungen empfehlen wir derzeit noch PHP 8.0:

```
apt update && apt install -y php8.0-fpm php8.0-gd php8.0-mysql php8.0-curl php8.0-xml php8.0-zip php8.0-intl php8.0-mbstring php8.0-bz2 php8.0-ldap php8.0-apcu php8.0-bcmath php8.0-gmp php8.0-imagick php8.0-igbinary php8.0-redis php8.0-smbclient php8.0-cli php8.0-common php8.0-opcache php8.0-readline imagemagick
```

Optional (bei einem geplanten Einsatz von Samba- und/oder CIFS-Shares oder einer LDAP(s)-Anbindung):

```
apt install ldap-utils nfs-common cifs-utils
```

Setzen Sie das richtige Datumsformat, um auch ein korrektes Logging zu ermöglichen:

```
timedatectl set-timezone Europe/Berlin
```

Bevor wir mit den Optimierungen von PHP beginnen sichern wir die Konfigurationsdateien:

```
cp /etc/php/8.0/fpm/pool.d/www.conf /etc/php/8.0/fpm/pool.d/www.conf.bak
cp /etc/php/8.0/cli/php.ini /etc/php/8.0/cli/php.ini.bak
cp /etc/php/8.0/fpm/php.ini /etc/php/8.0/fpm/php.ini.bak
cp /etc/php/8.0/fpm/php-fpm.conf /etc/php/8.0/fpm/php-fpm.conf.bak
cp /etc/php/8.0/mods-available/apcu.ini /etc/php/8.0/mods-available/apcu.ini.bak
cp /etc/ImageMagick-6/policy.xml /etc/ImageMagick-6/policy.xml.bak
```

Führen Sie nun alle nachfolgenden Optimierungen durch:

```
sed -i "s;/env\[HOSTNAME\] = /env\[HOSTNAME\] = /" /etc/php/8.0/fpm/pool.d/www.conf
sed -i "s;/env\[TMP\] = /env\[TMP\] = /" /etc/php/8.0/fpm/pool.d/www.conf
sed -i "s;/env\[TMPDIR\] = /env\[TMPDIR\] = /" /etc/php/8.0/fpm/pool.d/www.conf
sed -i "s;/env\[TEMP\] = /env\[TEMP\] = /" /etc/php/8.0/fpm/pool.d/www.conf
sed -i "s;/env\[PATH\] = /env\[PATH\] = /" /etc/php/8.0/fpm/pool.d/www.conf
sed -i "s/pm.max_children = ./pm.max_children = 120/" /etc/php/8.0/fpm/pool.d/www.conf
sed -i "s/pm.start_servers = ./pm.start_servers = 12/" /etc/php/8.0/fpm/pool.d/www.conf
sed -i "s/pm.min_spare_servers = ./pm.min_spare_servers = 6/" /etc/php/8.0/fpm/pool.d/www.conf
sed -i "s/pm.max_spare_servers = ./pm.max_spare_servers = 18/" /etc/php/8.0/fpm/pool.d/www.conf
sed -i "s/pm.max_requests = ./pm.max_requests = 1000/" /etc/php/8.0/fpm/pool.d/www.conf
sed -i "s/allow_url_fopen = ./allow_url_fopen = 1/" /etc/php/8.0/fpm/php.ini
sed -i "s/output_buffering = ./output_buffering = 'Off'/" /etc/php/8.0/cli/php.ini
sed -i "s/max_execution_time = ./max_execution_time = 3600/" /etc/php/8.0/cli/php.ini
sed -i "s/max_input_time = ./max_input_time = 3600/" /etc/php/8.0/cli/php.ini
sed -i "s/post_max_size = ./post_max_size = 10240M/" /etc/php/8.0/cli/php.ini
sed -i "s/upload_max_filesize = ./upload_max_filesize = 10240M/" /etc/php/8.0/cli/php.ini
```

```

sed -i "s;/date.timezone.*/date.timezone = Europe\\Berlin/" /etc/php/8.0/cli/php.ini
sed -i "s;/memory_limit = 128M/memory_limit = 1024M/" /etc/php/8.0/fpm/php.ini
sed -i "s;/output_buffering = */output_buffering = 'Off'/" /etc/php/8.0/fpm/php.ini
sed -i "s;/max_execution_time = */max_execution_time = 3600/" /etc/php/8.0/fpm/php.ini
sed -i "s;/max_input_time = */max_input_time = 3600/" /etc/php/8.0/fpm/php.ini
sed -i "s;/post_max_size = */post_max_size = 10240M/" /etc/php/8.0/fpm/php.ini
sed -i "s;/upload_max_filesize = */upload_max_filesize = 10240M/" /etc/php/8.0/fpm/php.ini
sed -i "s;/date.timezone.*/date.timezone = Europe\\Berlin/" /etc/php/8.0/fpm/php.ini
sed -i "s;/session.cookie_secure.*/session.cookie_secure = True/" /etc/php/8.0/fpm/php.ini
sed -i "s;/opcache.enable= */opcache.enable=1/" /etc/php/8.0/fpm/php.ini
sed -i "s;/opcache.enable_cli= */opcache.enable_cli=1/" /etc/php/8.0/fpm/php.ini
sed -i "s;/opcache.memory_consumption= */opcache.memory_consumption=128/" /etc/php/8.0/fpm/php.ini
sed -i "s;/opcache.interned_strings_buffer= */opcache.interned_strings_buffer=8/" /etc/php/8.0/fpm/php.ini
sed -i "s;/opcache.max_accelerated_files= */opcache.max_accelerated_files=10000/" /etc/php/8.0/fpm/php.ini
sed -i "s;/opcache.revalidate_freq= */opcache.revalidate_freq=1/" /etc/php/8.0/fpm/php.ini
sed -i "s;/opcache.save_comments= */opcache.save_comments=1/" /etc/php/8.0/fpm/php.ini
sed -i '$apc.enable_cli=1' /etc/php/8.0/mods-available/apcu.ini
sed -i "s;/rights=\"none\" pattern=\"PS\"/rights=\"read|write\" pattern=\"PS\"/" /etc/ImageMagick-6/policy.xml
sed -i "s;/rights=\"none\" pattern=\"EPS\"/rights=\"read|write\" pattern=\"EPS\"/" /etc/ImageMagick-6/policy.xml
sed -i "s;/rights=\"none\" pattern=\"PDF\"/rights=\"read|write\" pattern=\"PDF\"/" /etc/ImageMagick-6/policy.xml
sed -i "s;/rights=\"none\" pattern=\"XPS\"/rights=\"read|write\" pattern=\"XPS\"/" /etc/ImageMagick-6/policy.xml

```

Starten Sie nun beide Dienste, nginx und PHP, neu:

```
service php8.0-fpm restart
```

```
service nginx restart
```

Auch PHP ist nun bereits installiert und für Nextcloud optimiert. Für weitere PHP-Optimierungen finden Sie in [diesem Artikel](#) weitere Tuningmöglichkeiten. Starten wir nun mit der Installation und Konfiguration des Datenbankserver MariaDB.

3. Installation und Konfiguration von MariaDB 10.5

Die Installation von MariaDB erfolgt mit diesem Befehl:

```
apt update && apt install mariadb-server -y
```

Wie erfolgt ein Upgrade von MariaDB v. 10.4 zu v. 10.5

Härten wir nun den Datenbankserver mittels des mitgelieferten Tools „mysql_secure_installation“. Bei einer Erstinstallation besteht kein Rootpasswort, so dass Sie die Abfrage mit ENTER bestätigen könne. Es wird empfohlen, ein Passwort direkt zu setzen, der entsprechende Dialog erscheint automatisch:

```
mysql_secure_installation
```

```
Enter current password for root (enter for none): <ENTER> or type the password
```

```
Switch to unix_socket authentication [Y/n] Y
```

```
Set root password? [Y/n] Y
```

Remove anonymous users? [Y/n] **Y**

Disallow root login remotely? [Y/n] **Y**

Remove test database and access to it? [Y/n] **Y**

Reload privilege tables now? [Y/n] **Y**

Stoppen Sie nun den Datenbankserver und sichern dann die Standardkonfiguration, um unmittelbar danach Anpassungen vornehmen zu können:

```
service mysql stop
```

```
mv /etc/mysql/my.cnf /etc/mysql/my.cnf.bak
```

```
nano /etc/mysql/my.cnf
```

Kopieren Sie alle nachfolgenden Zeilen in die leere Datei:

```
[client]
```

```
default-character-set = utf8mb4
```

```
port = 3306
```

```
socket = /var/run/mysqld/mysqld.sock
```

```
[mysqld_safe]
```

```
log_error=/var/log/mysql/mysql_error.log
```

```
nice = 0
```

```
socket = /var/run/mysqld/mysqld.sock
```

```
[mysqld]
```

```
basedir = /usr
```

```
bind-address = 127.0.0.1
```

```
binlog_format = ROW
```

```
bulk_insert_buffer_size = 16M
```

```
character-set-server = utf8mb4
```

```
collation-server = utf8mb4_general_ci
```

```
concurrent_insert = 2
```

```
connect_timeout = 5
```

```
datadir = /var/lib/mysql
```

```
default_storage_engine = InnoDB
```

```
expire_logs_days = 2
```

```
general_log_file = /var/log/mysql/mysql.log
```

```
general_log = 0
```

```
innodb_buffer_pool_size = 1024M
```

```
innodb_buffer_pool_instances = 1
```

```
innodb_flush_log_at_trx_commit = 2
```

```
innodb_log_buffer_size = 32M
```

```
innodb_max_dirty_pages_pct = 90
```

```
innodb_file_per_table = 1
```

```
innodb_open_files = 400
```

```
innodb_io_capacity = 4000
innodb_flush_method = O_DIRECT
key_buffer_size = 128M
lc_messages_dir = /usr/share/mysql
lc_messages = en_US
log_bin = /var/log/mysql/mariadb-bin
log_bin_index = /var/log/mysql/mariadb-bin.index
log_error = /var/log/mysql/mysql_error.log
log_slow_verbosity = query_plan
log_warnings = 2
long_query_time = 1
max_allowed_packet = 16M
max_binlog_size = 100M
max_connections = 200
max_heap_table_size = 64M
myisam_recover_options = BACKUP
myisam_sort_buffer_size = 512M
port = 3306
pid-file = /var/run/mysqld/mysqld.pid
query_cache_limit = 2M
query_cache_size = 64M
query_cache_type = 1
query_cache_min_res_unit = 2k
read_buffer_size = 2M
read_rnd_buffer_size = 1M
skip-external-locking
skip-name-resolve
slow_query_log_file = /var/log/mysql/mariadb-slow.log
slow-query-log = 1
socket = /var/run/mysqld/mysqld.sock
sort_buffer_size = 4M
table_open_cache = 400
thread_cache_size = 128
tmp_table_size = 64M
tmpdir = /tmp
transaction_isolation = READ-COMMITTED
#unix_socket=OFF
user = mysql
wait_timeout = 600
```

```
[mysqldump]
max_allowed_packet = 16M
quick
quote-names
[isamchk]
key_buffer = 16M
```

Speichern und schließen Sie die Datei und starten dann den Datenbankserver neu, um die Nextcloud-Datenbank, den Nextcloud-Benutzer und sein Passwort einzurichten:

```
service mysql restart
mysql -uroot -p
```

Erläuterung:

Datenbankname: **nextcloud**

Datenbankbenutzer: **nextcloud**

Datenbankbenutzerpaßwort: **nextcloud**

```
CREATE DATABASE nextcloud CHARACTER SET utf8mb4 COLLATE utf8mb4_general_ci; CREATE USER nextcloud@localhost
identified by 'nextcloud'; GRANT ALL PRIVILEGES on nextcloud.* to nextcloud@localhost; FLUSH privileges; quit;
```

Überprüfen Sie, ob das Isolation-Level (read commit) und das Charset (utf8mb4) korrekt gesetzt wurden:

```
mysql -h localhost -uroot -p -e "SELECT @@TX_ISOLATION; SELECT SCHEMA_NAME 'database', default_character_set_name
'charset', DEFAULT_COLLATION_NAME 'collation' FROM information_schema.SCHEMATA WHERE
SCHEMA_NAME='nextcloud'"
```

Erscheint in der Ausgabe (resultset) „READ-COMMITTED“ und „utf8mb4_general_ci“ wurde alles korrekt eingerichtet und wir können mit der Installation von Redis fortfahren.

4. Installation und Konfiguration von Redis

Wir installieren den Redis-Server um die Nextcloudperformance zu steigern, da durch Redis die Last auf der MariaDB-Nextclouddatenbank reduziert wird:

```
apt update && apt install redis-server
```

Passen Sie die Redisconfiguration durch das Sichern und Anpassen der Konfiguration mittels Ausführen der nachfolgenden Befehle an:

```
cp /etc/redis/redis.conf /etc/redis/redis.conf.bak
sed -i "s/port 6379/port 0/" /etc/redis/redis.conf
sed -i s/\#\ unixsocket/\unixsocket/g /etc/redis/redis.conf
sed -i "s/unixsocketperm 700/unixsocketperm 770/" /etc/redis/redis.conf
sed -i "s/# maxclients 10000/maxclients 512/" /etc/redis/redis.conf
usermod -aG redis www-data
cp /etc/sysctl.conf /etc/sysctl.conf.bak
sed -i '$avm.overcommit_memory = 1' /etc/sysctl.conf
```

Aus hinreichender Installationserfahrung heraus empfehle ich Ihnen, den gesamten Server einmalig neu zu starten:

reboot now

Gratulation, der Server ist bereits Installiert und eingerichtet, so dass nun mit der Einrichtung der Nextcloud begonnen werden kann.

5. Installation und Optimierung der Nextcloud (inkl. SSL)

Wir richten nun verschiedene vhost, also Serverkonfigurationsdateien, ein und modifizieren die Standard vhost-Datei persistent. Da das System zuvor neu gestartet wurde wechseln wir erneut in den privilegierten Benutzermodus, sichern die Standard vhost-Datei namen default.conf und legen leere vHost-Dateien zum Konfigurieren an.

```
sudo -s
```

```
[ -f /etc/nginx/conf.d/default.conf ] && mv /etc/nginx/conf.d/default.conf /etc/nginx/conf.d/default.conf.bak  
touch /etc/nginx/conf.d/default.conf  
touch /etc/nginx/conf.d/http.conf  
touch /etc/nginx/conf.d/nextcloud.conf
```

Somit ist durch die leere „default.conf“ Datei auch bei späteren Aktualisierungen des Webservers sichergestellt, dass diese Standardkonfiguration den Nextcloudbetrieb nicht beeinflusst.

Erstellen Sie die globale vhost-Datei, um die http-Standardanfragen permanent auf https umzuleiten und zudem die SSL-Zertifikatskommunikation mit Let'sEncrypt zu ermöglichen.

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

Kopieren Sie alle nachfolgenden Zeilen in die Datei **http.conf** und passen die **rot** markierten Werte entsprechend Ihres Systems an:

```
upstream php-handler {  
    server unix:/run/php/php8.0-fpm.sock;  
}  
  
server {  
    listen 80 default_server;  
    listen [::]:80 default_server;  
    server_name ihre.domain.de;  
    root /var/www;  
    location ^~ /.well-known/acme-challenge {  
        default_type text/plain;  
        root /var/www/letsencrypt;  
    }  
    location / {  
        return 301 https://$host$request_uri;  
    }  
}
```

Speichern und schließen Sie diese Datei. Bearbeiten Sie nun die eigentliche Nextcloud vHost-Datei **nextcloud.conf**, die sämtliche Konfigurationen für den Betrieb der Nextcloud enthält.

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

Kopieren Sie alle nachfolgenden Zeilen in die Datei **nextcloud.conf** und passen den **rot** markierten Werte entsprechend Ihres Systems an:

```
server {  
listen 443    ssl http2;  
listen [::]:443 ssl http2;  
server_name ihre.domain.de;  
ssl_certificate /etc/ssl/certs/ssl-cert-snakeoil.pem;  
ssl_certificate_key /etc/ssl/private/ssl-cert-snakeoil.key;  
ssl_trusted_certificate /etc/ssl/certs/ssl-cert-snakeoil.pem;  
#ssl_certificate /etc/letsencrypt/rsa-certs/fullchain.pem;  
#ssl_certificate_key /etc/letsencrypt/rsa-certs/privkey.pem;  
#ssl_certificate /etc/letsencrypt/ecc-certs/fullchain.pem;  
#ssl_certificate_key /etc/letsencrypt/ecc-certs/privkey.pem;  
#ssl_trusted_certificate /etc/letsencrypt/ecc-certs/chain.pem;  
ssl_dhparam /etc/ssl/certs/dhparam.pem;  
ssl_session_timeout 1d;  
ssl_session_cache shared:SSL:50m;  
ssl_session_tickets off;  
ssl_protocols TLSv1.3 TLSv1.2;  
ssl_ciphers 'TLS-CHACHA20-POLY1305-SHA256:TLS-AES-256-GCM-SHA384:ECDHE-RSA-AES256-GCM-SHA512:DHE-RSA-AES256-GCM-SHA512:ECDHE-RSA-AES256-GCM-SHA384:DHE-RSA-AES256-GCM-SHA384';  
ssl_ecdh_curve X448:secp521r1:secp384r1;  
ssl_prefer_server_ciphers on;  
ssl_stapling on;  
ssl_stapling_verify on;  
add_header Strict-Transport-Security "max-age=15768000; includeSubDomains; preload;" always;  
client_max_body_size 5120M;  
fastcgi_buffers 64 4K;  
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/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;  
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 "none" always;
add_header X-XSS-Protection "1; mode=block" always;
fastcgi_hide_header X-Powered-By;
fastcgi_read_timeout 3600;
fastcgi_send_timeout 3600;
fastcgi_connect_timeout 3600;
root /var/www/nextcloud;
index index.php index.html /index.php$request_uri;
expires 1m;
location = / {
if ( $http_user_agent ~ ^DavClnt ) {
return 302 /remote.php/webdav/$is_args$args;
}
}
location = /robots.txt {
allow all;
log_not_found off;
access_log off;
}
location ^~ /apps/rainloop/app/data {
deny all;
}
location ^~ /.well-known {
location = /.well-known/carddav { return 301 /remote.php/dav/; }
location = /.well-known/caldav { return 301 /remote.php/dav/; }
location ^~ /.well-known { return 301 /index.php/$uri; }
try_files $uri $uri/ =404;
}
location ~ ^/(?!build|tests|config|lib|3rdparty|templates|data)(?:$|/) { return 404; }
location ~ ^/(?!\.|autotest|occ|issue|indie|db_|console) { return 404; }
location ~ \.php(?:$|/) {
rewrite ^/(?!index|remote|public|cron|core/ajax/update|status|ocs/v[12]|updater/.+|oc[ms]-
provider/.+|.+/richdocumentscode/proxy) /index.php$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 $path_info;
fastcgi_param HTTPS on;
fastcgi_param modHeadersAvailable true;
fastcgi_param front_controller_active true;
fastcgi_pass php-handler;
fastcgi_intercept_errors on;
fastcgi_request_buffering off;
}
location ~ \.(?:css|js|svg|gif)$ {
try_files $uri /index.php$request_uri;
expires 6M;
access_log off;
}
location ~ \.woff2?$ {
try_files $uri /index.php$request_uri;
expires 7d;
access_log off;
}
location / {
try_files $uri $uri /index.php$request_uri;
}
}

```

Speichern und schließen Sie diese Datei und erweitern dann die Server- und Systemsicherheit durch die Möglichkeit des sicheren Schlüsselaustauschs mittels eines [Diffie-Hellman Schlüssels](#) (dhparam.pem):

```
openssl dhparam -dsaparam -out /etc/ssl/certs/dhparam.pem 4096
```

Bitte haben Sie nun Geduld! Das Generieren kann – in Abhängigkeit von der Systemleistung – einige Minuten dauern. Erst wenn das Generieren abgeschlossen ist, starten wir den Webserver erneut durch.

```
service nginx restart
```

Wir beginnen nun die ‚eigentliche‘ Installation der Nextcloud Software und richten dafür die SSL Zertifikate von Let’s Encrypt mittels [acme](#) ein. Wechseln Sie dafür in das Arbeitsverzeichnis

```
cd /usr/local/src
```

und laden das aktuelle Nextcloud Release herunter:

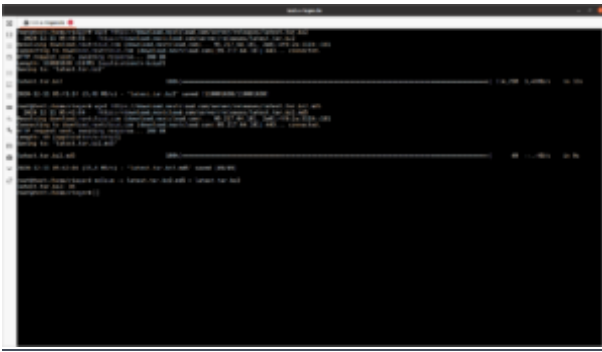
```
wget https://download.nextcloud.com/server/releases/latest.tar.bz2
```

```
wget https://download.nextcloud.com/server/releases/latest.tar.bz2.md5
```

Überprüfen Sie die Dateien:

```
md5sum -c latest.tar.bz2.md5 < latest.tar.bz2
```

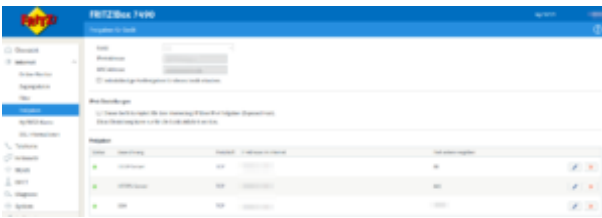
Nur wenn Ihnen der Test mit „OK“ bestätigt wird, fahren wir fort!



Entpacken Sie die Nextcloud Software in das Webverzeichnis (var/www), setzen dann die Berechtigung adäquat und Löschen die Download-Datei:

```
tar -xjf latest.tar.bz2 -C /var/www && chown -R www-data:www-data /var/www/ && rm -f latest.tar.bz2
```

Bitte stellen Sie sicher, dass Ihr Server sowohl über Port 80/TCP als auch über Port 443/TCP von außen erreichbar ist. Das Erstellen und Aktualisieren von Let's Encryptzertifikaten erfolgt zwingend über http und Port 80! Für das Zertifikatshandling erstellen wir nun einen dedizierten Benutzer und fügen diesen der www-data Gruppe hinzu:



```
adduser --disabled-login acmeuser
```

```
usermod -a -G www-data acmeuser
```

Diesem technischen Benutzer erteilen wir noch die notwendigen Berechtigungen, um bei einer Zertifikatserneuerung den notwendigen Webserverstart initiieren zu können.

```
visudo
```

In der Mitte der Datei, unterhalb von

```
[..]
```

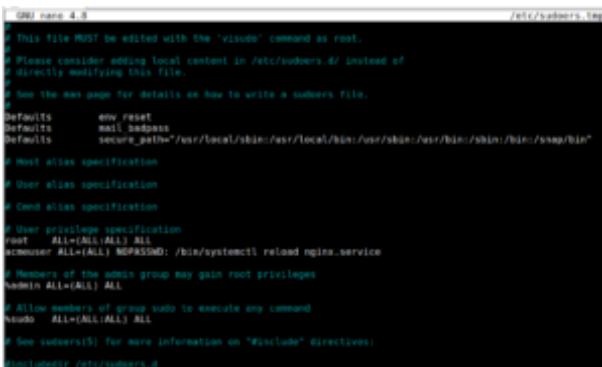
User privilege specification

```
root ALL=(ALL:ALL) ALL
```

```
[...]
```

tragen Sie die folgende Zeile ein:

```
acmeuser ALL=NOPASSWD: /bin/systemctl reload nginx.service
```



Mit **STRG+X** gefolgt von einem **y** speichern und verlassen Sie diese Datei.

Wechseln Sie in die Shell des neuen Benutzers (acmeuser) um die Zertifikatssoftware zu installieren und verlassen diese Shell danach wieder:

```
su - acmeuser  
curl https://get.acme.sh | sh  
exit
```

Passen Sie die entsprechenden Berechtigungen an, um die neuen Zertifikate darin speichern zu können:

```
chmod -R 775 /var/www/letsencrypt && chmod -R 770 /etc/letsencrypt && chown -R www-data:www-data /var/www/  
/etc/letsencrypt
```

Wechseln Sie erneut in die Shell des neuen Benutzers

```
su - acmeuser
```

und requestieren (beantragen) die SSL-Zertifikate. Ersetzen Sie dabei **ihre.domain.de** mit Ihrer Domain :

```
acme.sh --issue -d ihre.domain.de --keylength 4096 -w /var/www/letsencrypt --key-file /etc/letsencrypt/rsa-  
certs/privkey.pem --ca-file /etc/letsencrypt/rsa-certs/chain.pem --cert-file /etc/letsencrypt/rsa-certs/cert.pem --fullchain-file  
/etc/letsencrypt/rsa-certs/fullchain.pem --reloadcmd "sudo /bin/systemctl reload nginx.service"
```

```
acme.sh --issue -d ihre.domain.de --keylength ec-384 -w /var/www/letsencrypt --key-file /etc/letsencrypt/ecc-  
certs/privkey.pem --ca-file /etc/letsencrypt/ecc-certs/chain.pem --cert-file /etc/letsencrypt/ecc-certs/cert.pem --fullchain-  
file /etc/letsencrypt/ecc-certs/fullchain.pem --reloadcmd "sudo /bin/systemctl reload nginx.service"
```

Verlassen Sie die Shell des neuen Benutzers

```
exit
```

und legen sich dann ein Skript an, dass zukünftig die Berechtigungen überprüft und korrigiert (permissions.sh):

```
nano /root/permissions.sh
```

Kopieren Sie alle Zeilen in die Datei:

```
#!/bin/bash  
find /var/www/ -type f -print0 | xargs -0 chmod 0640  
find /var/www/ -type d -print0 | xargs -0 chmod 0750  
chmod -R 775 /var/www/letsencrypt  
chmod -R 770 /etc/letsencrypt  
chown -R www-data:www-data /var/www /etc/letsencrypt  
chown -R www-data:www-data /var/nc_data  
chmod 0644 /var/www/nextcloud/.htaccess  
chmod 0644 /var/www/nextcloud/.user.ini  
exit 0
```

Markieren Sie das Skript als ausführbar und führen es dann direkt aus:

```
chmod +x /root/permissions.sh  
/root/permissions.sh
```

Entfernen Sie Ihre bisher verwendeten Self-Signed-Zertifikate aus nginx und aktivieren Sie die neuen, vollwertigen und bereits gültigen SSL Zertifikate von Let's Encrypt. Starten Sie dann den Webserver neu:

```
sed -i 's/ssl-cert-snakeoil/d' /etc/nginx/conf.d/nextcloud.conf  
sed -i s/#ssl/g /etc/nginx/conf.d/nextcloud.conf  
service nginx restart
```

Um sowohl die SSL-Zertifikate automatisch zu erneuern, als auch den notwendigen Webserverneustart zu initiieren, wurde automatisch ein Cronjob angelegt.

```
crontab -l -u acmeuser
```

Wir können nun mit der Einrichtung der Nextcloud fortfahren. Dazu verwenden Sie den nachfolgenden „silent“ Installationsbefehl:

```
sudo -u www-data php /var/www/nextcloud/occ maintenance:install --database "mysql" --database-name "nextcloud" --database-user "nextcloud" --database-pass "nextcloud" --admin-user "YourNextcloudAdmin" --admin-pass "YourNextcloudAdminPassword" --data-dir "/var/nc_data"
```

Erläuterungen:

database-name „nextcloud“ : Datenbankname aus [Kapitel 3](#)

database-user „nextcloud“ : Datenbankbenutzer aus [Kapitel 3](#)

database-pass „nextcloud“ : Datenbankbenutzerpasswort aus [Kapitel 3](#)

admin-user „YourNextcloudAdmin“ : frei wählbar von Ihnen

admin-pass „YourNextcloudAdminPassword“ : frei wählbar von Ihnen

Warten Sie bis die Installation der Nextcloud abgeschlossen wurde und passen dann die zentrale Konfigurationsdatei der Nextcloud [„config.php“](#) als Webuser www-data an:

1. Fügen Sie Ihre Domain als trusted domain hinzu, ergänzen Sie dabei **ihre.domain.de** mit Ihrer dedizierten Domain:

```
sudo -u www-data php /var/www/nextcloud/occ config:system:set trusted_domains 0 --value=ihre.domain.de
```

2. Setzen Sie Ihre Domain als overwrite.cli.url, ergänzen Sie dabei **ihre.domain.de** mit Ihrer dedizierten Domain:

```
sudo -u www-data php /var/www/nextcloud/occ config:system:set overwrite.cli.url --value=https://ihre.domain.de
```

Nun erweitern wir abschließend die Nextcloud Konfiguration. Sichern Sie dazu zuerst die bestehende config.php und führen dann die nachfolgenden Zeilen in einem Block aus:

```
sudo -u www-data cp /var/www/nextcloud/config/config.php /var/www/nextcloud/config/config.php.bak
```

```
sudo -u www-data sed -i 's/^[ ]*//' /var/www/nextcloud/config/config.php
```

```
sudo -u www-data sed -i ');d' /var/www/nextcloud/config/config.php
```

```
sudo -u www-data cat <<EOF >>/var/www/nextcloud/config/config.php
```

```
'activity_expire_days' => 14,
```

```
'auth.bruteforce.protection.enabled' => true,
```

```
'blacklisted_files' =>
```

```
array (
```

```
0 => '.htaccess',
```

```
1 => 'Thumbs.db',
```

```
2 => 'thumbs.db',
```

```
),
```

```
'cron_log' => true,
```

```
'default_phone_region' => 'DE',
```

```
'enable_previews' => true,
```

```
'enabledPreviewProviders' =>
array (
  0 => 'OC\Preview\PNG',
  1 => 'OC\Preview\JPEG',
  2 => 'OC\Preview\GIF',
  3 => 'OC\Preview\BMP',
  4 => 'OC\Preview\XBitmap',
  5 => 'OC\Preview\Movie',
  6 => 'OC\Preview\PDF',
  7 => 'OC\Preview\MP3',
  8 => 'OC\Preview\TXT',
  9 => 'OC\Preview\MarkDown',
),
'filesystem_check_changes' => 0,
'filelocking.enabled' => 'true',
'htaccess.RewriteBase' => '/',
'integrity.check.disabled' => false,
'knowledgebaseenabled' => false,
'logfile' => '/var/nc_data/nextcloud.log',
'loglevel' => 2,
'logtimezone' => 'Europe/Berlin',
'log_rotate_size' => 104857600,
'maintenance' => false,
'memcache.local' => '\OC\Memcache\APCu',
'memcache.locking' => '\OC\Memcache\Redis',
'overwriteprotocol' => 'https',
'preview_max_x' => 1024,
'preview_max_y' => 768,
'preview_max_scale_factor' => 1,
'redis' =>
array (
  'host' => '/var/run/redis/redis-server.sock',
  'port' => 0,
  'timeout' => 0.0,
),
'quota_include_external_storage' => false,
'share_folder' => '/Freigaben',
'skeletondirectory' => '',
'theme' => '',
```

```
'trashbin_retention_obligation' => 'auto, 7',  
'updater.release.channel' => 'stable',  
);  
EOF
```

Modifizieren Sie die „.user.ini“

```
sudo -u www-data sed -i "s/output_buffering=.*output_buffering=0/" /var/www/nextcloud/.user.ini
```

und passen die Nextcloud-Apps als user www-data an

```
sudo -u www-data php /var/www/nextcloud/occ app:disable survey_client  
sudo -u www-data php /var/www/nextcloud/occ app:disable firstrunwizard  
sudo -u www-data php /var/www/nextcloud/occ app:enable admin_audit  
sudo -u www-data php /var/www/nextcloud/occ app:enable files_pdfviewer
```

Nextcloud ist ab sofort voll einsatzfähig, optimiert und abgesichert. Starten Sie alle relevanten Services neu:

```
service nginx stop  
service php8.0-fpm stop  
service mysql restart  
service php8.0-fpm restart  
service redis-server restart  
service nginx restart
```

Richten Sie einen Cronjob für Nextcloud als „www-data“ – Benutzer ein:

```
crontab -u www-data -e
```

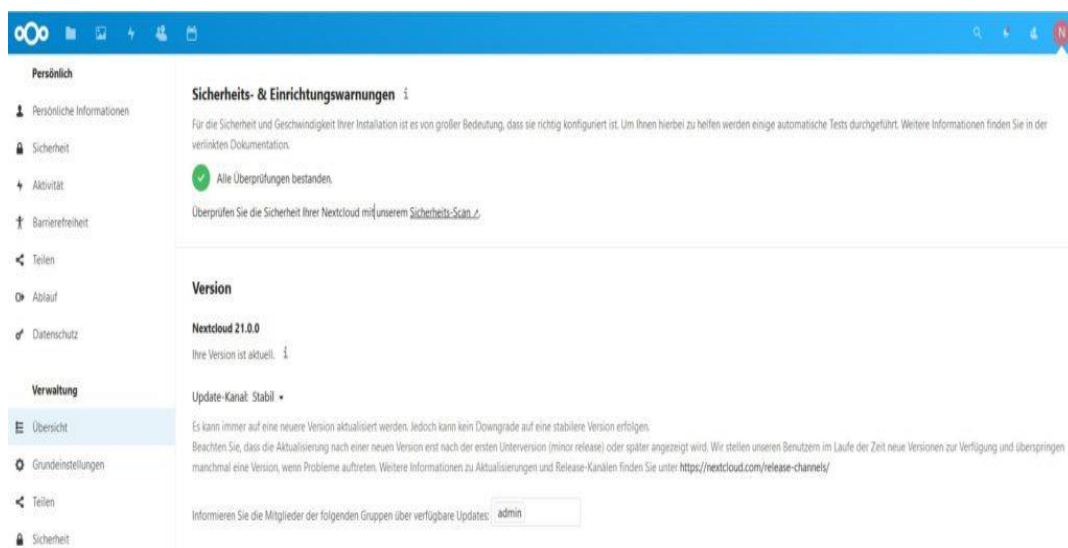
Fügen Sie diese Zeile ein

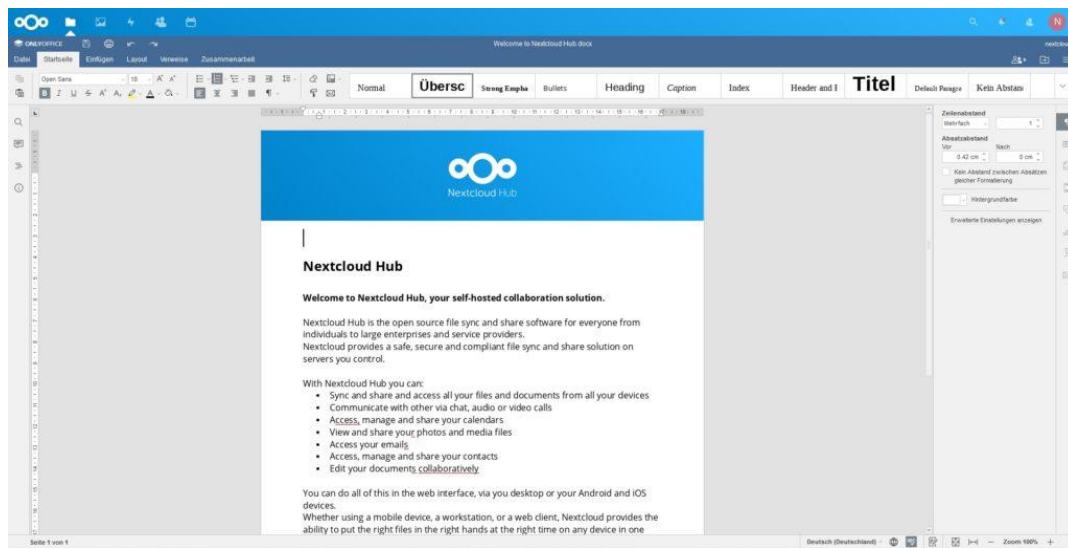
```
* /5 * * * * php -f /var/www/nextcloud/cron.php > /dev/null 2>&1
```

Speichern und schließen Sie dann die Datei und konfigurieren Sie den Nextcloud-Job von „Ajax“ zu „Cron“ mittels der Nextclouds CLI um:

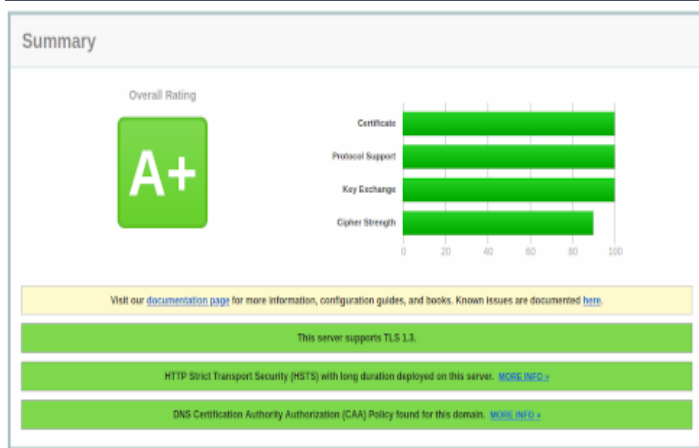
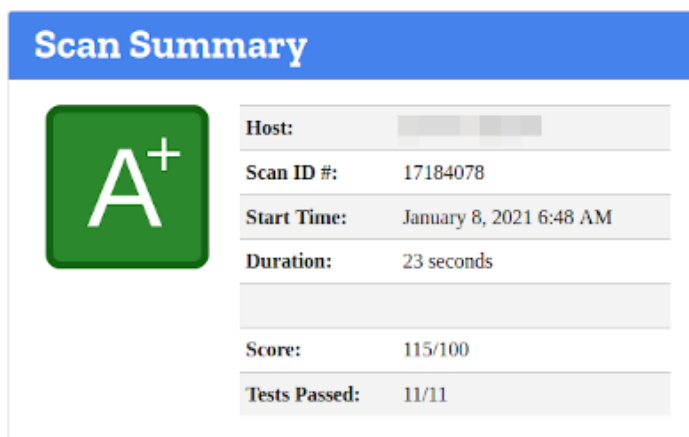
```
sudo -u www-data php /var/www/nextcloud/occ background:cron
```

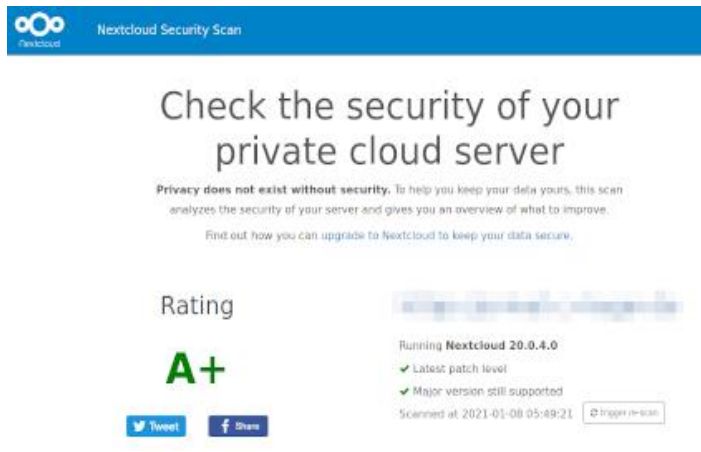
Bitte nehmen Sie sich etwas Zeit und überprüfen den Sicherheitsstatus Ihres Servers.





Ziel sollte zudem sein, mindestens das nachfolgend dargestellte „A+“-Ergebnis in den Tests zu erzielen:





6. Härtung (fail2ban and ufw)

Zuerst installieren wir fail2ban um den Server gegen Brute-force-Attacken und fehlerhafte Loginversuche zu schützen:

```
apt update && apt install fail2ban -y
```

Erstellen Sie eine neue Filterdatei und befüllen diese wie nachfolgend beschrieben (alternativ: [Download](#))

```
touch /etc/fail2ban/filter.d/nextcloud.conf
```

Kopieren Sie alles von „cat ...“ bis „... EOF“ in Ihre Zwischenablage und fügen es dann in die Shell ein:

```
cat <<EOF >/etc/fail2ban/filter.d/nextcloud.conf
```

[Definition]

```
_groupsre = (?:{?:,?\s*\w+":(?:["^"]+|\w+))*
```

```
failregex = ^\{%( _groupsre)s,?\s*"remoteAddr": "<HOST>"%( _groupsre)s,?\s*"message": "Login failed:
```

```
^\{%( _groupsre)s,?\s*"remoteAddr": "<HOST>"%( _groupsre)s,?\s*"message": "Trusted domain error.
```

```
datepattern = ,?\s*"time"\s*:\s*"%%Y-%%m-%%d[T ]%%H:%%M:%%S(%%z)?"
```

```
EOF
```

Bestätigen Sie mit <ENTER> um die Datei zu befüllen. Das Ergebnis sieht im Anschluß wie folgt aus:

```
cat /etc/fail2ban/filter.d/nextcloud.conf
```

```
[Definition]
_groupsre = (?:{?:,?\s*\w+":(?:["^"]+|\w+))*
failregex = ^\{%( _groupsre)s,?\s*"remoteAddr": "<HOST>"%( _groupsre)s,?\s*"message": "Login failed:
          ^\{%( _groupsre)s,?\s*"remoteAddr": "<HOST>"%( _groupsre)s,?\s*"message": "Trusted domain e
datepattern = ,?\s*"time"\s*:\s*"%%Y-%%m-%%d[T ]%%H:%%M:%%S(%%z)?"
```

Legen Sie nun eine neue Jail-Datei an ([Download hier](#)):

```
nano /etc/fail2ban/jail.d/nextcloud.local
```

Kopieren Sie alle nachfolgenden Zeilen hinein:

```
[nextcloud]
```

```
backend = auto
```

```
enabled = true
```

```
port = 80,443
```

```
protocol = tcp
```

```
filter = nextcloud
```

```
maxretry = 5
```

```
bantime = 3600
```

```
findtime = 36000
```

```
logpath = /var/nc_data/nextcloud.log
```

Mit den zuvor dargestellten Parametern wird nach 5 fehlerhaften Anmeldeversuchen (**maxretry**) innerhalb der letzten 36000 Sekunden (**findtime**, das entspricht 10h) die IP des potentiellen Angreifers für einen Zeitraum von 3600 Sekunden (**bantime**, entspricht 1h) gesperrt.

Starten Sie fail2ban neu und überprüfen den fail2ban-status:

```
service fail2ban restart
```

```
fail2ban-client status nextcloud
```

Ab sofort werden IP-Adressen, von denen 5 oder mehr fehlerhafte Anmeldeversuche innerhalb der letzten 10h ausgegangen sind für 1h gesperrt und Ihr Server somit vor weiteren Attacken geschützt. Wenn Sie die Sperre manuell testen wollen und die resultierende Sperre Ihrer IP respektive bereits gesperrte IPs entsperren wollen, so führen Sie zuerst diesen Befehl aus,

```
fail2ban-client status nextcloud
```

um sich die gesperrten IP-Adressen anzeigen zu lassen. Die dargestellte(n) IP(s) können Sie mittels des nachfolgenden Befehls entsperren:

```
fail2ban-client set nextcloud unbanip <ip-adresse>
```

Abschließend installieren wir noch eine Firewall, die sogenannte **un**complicated **f**irewall (ufw):

Sofern Sie zuvor den SSH-Port von 22 auf einen anderen Port geändert haben, so müssen Sie die 22 entsprechend ersetzen!

```
apt install ufw -y
```

```
ufw allow 80/tcp
```

```
ufw allow 443/tcp
```

```
ufw allow 22/tcp
```

Möchten Sie SSH nicht nach außen freigeben (**empfohlen!**) und nur aus dem internen Netz nutzen, so ersetzen Sie den letzten ufw-Befehl (**ufw allow 22/tcp**) durch diesen:

```
ufw allow proto tcp from 192.168.2.0/24 to any port 22
```

Ersetzen Sie das exemplarische Netz (192.168.2.0/24) durch das bei Ihnen genutzte Netz!

Setzen Sie das Firewall-Logging auf „medium“ und verhindern nicht definierte eingehende Verbindungen.

```
ufw logging medium
```

```
ufw default deny incoming
```

Aktivieren Sie die Firewall und starten diese neu:

```
ufw enable
```

```
service ufw restart
```

Nextcloud kommuniziert mit verschiedenen Remote-Servern, um gewisse Information verarbeiten, austauschen und bereitstellen zu können:

- www.nextcloud.com, www.startpage.com, www.eff.org, www.edri.org

Überprüfung der Internetverbindung

- apps.nextcloud.com
Verfügbare Apps / AppStore
- updates.nextcloud.com
Nextcloud Updates
- lookup.nextcloud.com
Aktualisierung von Federated Clouds/Services
- push-notifications.nextcloud.com
Push Nachrichten für mobile Clients
- surveyserver.nextcloud.com
Umfragen – nur sofern der Administrator dem Versenden ANONYMISierter DATEN zugestimmt hat
- Beliebige „remote“ Nextcloud Server die federiert sind

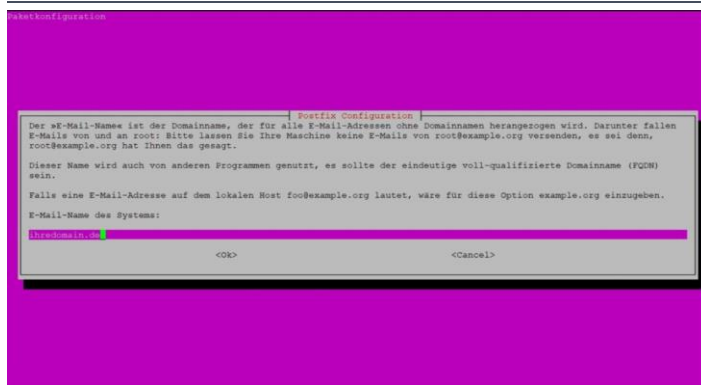
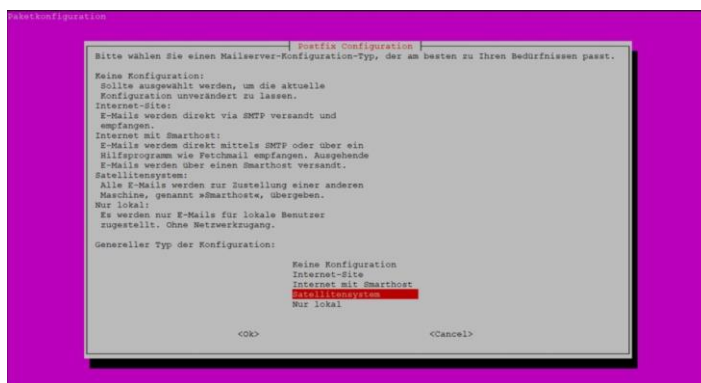
Quelle: [Nextcloud](#), 24. Februar 2021

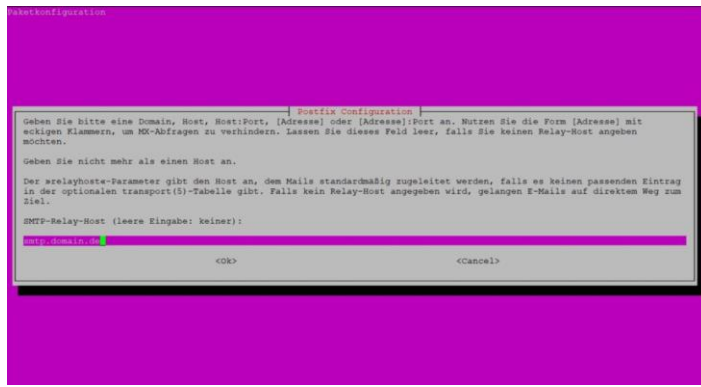
7. Systemmails per postfix versenden

Aktualisieren Sie ihren Server und installieren Sie postfix. Sie haben damit die Möglichkeit, sich von fail2ban, apticron und bei SSH-Anmeldungen per Mail informieren zu lassen:

```
apt update && apt upgrade -y && apt install -y postfix mailutils
```

Erstellen Sie Ihre Mailkonfiguration anhand meines nachfolgenden Beispiels: Wählen Sie zuerst Satellitensystem aus, geben dann ihre Domäne ein (bspw.domain.de) und zuletzt noch den smtp-Server.





Passen Sie nun die Postfixkonfiguration an:

```
vi /etc/postfix/main.cf
```

Ersetzen Sie die **rot** markierten Stellen:

```
smtpd_banner = $myhostname ESMTP $mail_name (Ubuntu)
biff = no
append_dot_mydomain = no
readme_directory = no
compatibility_level = 2
smtpd_tls_cert_file=/etc/ssl/certs/ssl-cert-snakeoil.pem
smtpd_tls_key_file=/etc/ssl/private/ssl-cert-snakeoil.key
smtpd_tls_security_level=may
smtp_tls_CApath=/etc/ssl/certs
smtp_tls_security_level=may
smtp_tls_session_cache_database = btree:${data_directory}/smtp_scache
smtpd_relay_restrictions = permit_mynetworks permit_sasl_authenticated defer_unauth_destination
myhostname = ihre.domain.de
alias_maps = hash:/etc/aliases
alias_database = hash:/etc/aliases
mydestination = $myhostname, ihre.domain.de, localhost.domain.de, localhost
relayhost = smtp.domain.de:587
mynetworks = 127.0.0.0/8 [::ffff:127.0.0.0]/104 [::1]/128
mailbox_size_limit = 0
recipient_delimiter = +
inet_interfaces = loopback-only
inet_protocols = all
#Bei Problemen mit IPv6 stellen Sie die Zeile wie nachfolgend um
#inet_protocols = ipv4
smtp_sasl_auth_enable = yes
smtp_sasl_password_maps = hash:/etc/postfix/sasl_passwd
smtp_sasl_security_options = noanonymous
sender_canonical_maps = hash:/etc/postfix/sender_canonical
```

```
smtp_use_tls = yes
```

```
smtp_enforce_tls = yes
```

Hinterlegen Sie nun die Zugangsdaten für das Versenden von Mails:

```
vi /etc/postfix/sasl_passwd
```

Tragen Sie die Daten wie folgt ein und ersetzen die roten Werte durch Ihre:

```
smtp.domain.de ihremail.domain.de:passwort
```

Da in dieser Datei das Passwort im Klartext steht setzen wir die Dateiberechtigungen auf 600

```
chmod 600 /etc/postfix/sasl_passwd
```

Schreiben Sie nun die Domäne in die Datei mailname:

```
vi /etc/mailname
```

Ersetzen Sie den roten Wert durch Ihre Domäne:

```
domain.de
```

Abschließend definieren wir noch den Bezug von Benutzern zu Mailadressen. Öffnen Sie die Datei

```
vi /etc/postfix/sender_canonical
```

und legen dort Benutzer und Mailadressen fest:

```
root mail@domain.de
```

```
<Ihr Benutzer> mail@domain.de
```

```
www-data mail@domain.de
```

```
default mail@domain.de
```

Jetzt werden die Konfiguration kompiliert und Postfix neu gestartet:

```
postmap /etc/postfix/sasl_passwd
```

```
postmap /etc/postfix/sender_canonical
```

```
service postfix restart
```

Testen Sie nun den Versand einer Mail über Ihr Postfix

```
echo „Dies ist eine Testmail“ | mailx -s „Test“ ihremail@domain.de
```

Sollte die Mail nicht ankommen, so sehen Sie bitte im Log (mail.log) nach:

```
tail -f /var/log/mail.log
```

Passen Sie die PHP-Konfiguration an um auch PHP-Mails über postfix zu versenden:

```
nano /etc/php/8.0/fpm/php.ini
```

Setzen Sie den sendmail_path wie folgt:

```
sendmail_path = "/usr/sbin/sendmail -t -i"
```

und starten dann PHP neu:

```
service php8.0-fpm restart
```

Sie können nun auch Nextcloud entsprechend konfigurieren



Ihr Mailserver ist nun einsatzbereit und Sie können nun weitere Systemmails (bspw. von fail2ban und apticron) konfigurieren:

(optional) 7.1 Konfiguration von fail2ban-Mailbenachrichtigungen

Ersetzen Sie in der fail2ban-Konfiguration die nachfolgenden Parameter durch Ihre, um Benachrichtigungen bei fehlerhaften Loginversuchen und Bans zu erhalten. Sichern Sie dazu die Originalkonfiguration von fail2ban und bearbeiten diese dann:

```
cp /etc/fail2ban/jail.conf /etc/fail2ban/jail.conf.bak
```

```
nano /etc/fail2ban/jail.conf
```

Ersetzen Sie die roten Werte:

```
...
destemail = ihre@mailadresse.de
...
sender = ihre@mailadresse.de
...
mta = mail
...
# action = %(action_)s
action = %(action_mwl)s
...
```

Um bei einem Serverneustart fail2ban-Mails zu unterdrücken passen Sie die folgende Datei an:

```
nano /etc/fail2ban/action.d/mail-buffered.local
```

Kopieren Sie den Inhalt hinein:

[Definition]

```
actionstart =
```

```
actionstop =
```

Legen Sie dann Dummy-Dateien durch das Ausführen der folgenden Befehle an:

```
cp /etc/fail2ban/action.d/mail-buffered.local /etc/fail2ban/action.d/mail.local
```

```
cp /etc/fail2ban/action.d/mail-buffered.local /etc/fail2ban/action.d/mail-whois-lines.local
```

```
cp /etc/fail2ban/action.d/mail-buffered.local /etc/fail2ban/action.d/mail-whois.local
```

```
cp /etc/fail2ban/action.d/mail-buffered.local /etc/fail2ban/action.d/sendmail-buffered.local
```

```
cp /etc/fail2ban/action.d/mail-buffered.local /etc/fail2ban/action.d/sendmail-common.local
```

Starten Sie den fail2ban-Service neu

```
service fail2ban restart
```

und Sie werden ab sofort (nur noch) bei Bans, also neu blockierten IP-Adressen die durch fehlerhafte Loginversuche aufgefallen sind, informiert.

(optional) 7.2 Installation von Apticron inkl. Mailbenachrichtigungen

Apticron informiert Sie über verfügbare Systemaktualisierungen bzw. auch dann, wenn ihr System „up2date“ ist. Installieren Sie apticron aus den Standardsoftwarequellen Ubuntu:

```
apt update && apt upgrade -y && apt install apticron -y
```

Nun passen wir apticron an und ändern wenigstens die folgenden Parameter:

Nur für **Ubuntu 18.04.+**:

```
cp /etc/apticron/apticron.conf /etc/apticron/apticron.conf.bak
```

```
nano /etc/apticron/apticron.conf
```

Nur für **Ubuntu 20.04+ und Debian 10.5+**:

```
cp /usr/lib/apticron/apticron.conf /etc/apticron/apticron.conf
```

```
nano /etc/apticron/apticron.conf
```

Ab hier geht es wieder für alle Betriebssysteme weiter:

```
...
EMAIL="ihre@mailadresse.de"
...
SYSTEM="ihre.domain.de"
...
NOTIFY_HOLDS="1"
...
NOTIFY_NO_UPDATES="1"
...
CUSTOM_SUBJECT='$SYSTEM: $NUM_PACKAGES package update(s)'
...
CUSTOM_NO_UPDATES_SUBJECT='$SYSTEM: no updates available'
...
CUSTOM_FROM="ihre@mailadresse.de"
...
```

Überprüfen Sie apticron und den soeben konfigurierten Mailversand indem sie apticron aufrufen:

```
apticron
```

Sie erhalten nun umgehend eine Mailbenachrichtigung über Ihren aktuellen Systemzustand. Passen Sie zuletzt noch den Cronjob an, um sich regelmäßig und automatisch benachrichtigen zu lassen:

```
cp /etc/cron.d/apticron /etc/cron.d/apticron.bak
```

```
nano /etc/cron.d/apticron
```

```
30 7 * * * root if test -x /usr/sbin/apticron; then /usr/sbin/apticron --cron; else true; fi
```

Apticron würde Sie am obigen Beispiel jeden Morgen um **07.30 Uhr** per Mail über Ihren Systemaktualitätsgrad informieren.

(optional) 7.3 Mailbenachrichtigungen bei SSH-Einwahl

Passen Sie die Profildatei an und erweitern diese am Ende um die folgenden Zeilen:

```
nano /etc/profile

if [ -n "$SSH_CLIENT" ]; then

    echo 'Login on' `hostname` `date` `who -m` | mail -s "Login on `hostname` from `echo $SSH_CLIENT |
    awk '{print $1}'`" ihre@mailadresse.de

fi
```

Bei jeder erfolgreichen SSH-Einwahl werden Sie ab sofort aktiv benachrichtigt.

8. Optimieren & aktualisieren der Nextcloud per Skript

Erstellen Sie ein Skript um die Nextcloud sowie die aktivierten Apps zu aktualisieren und zu optimieren:

```
cd /root

nano upgrade.sh

#!/bin/bash

/usr/sbin/service nginx stop

sudo -u www-data php /var/www/nextcloud/updater/updater.phar

sudo -u www-data php /var/www/nextcloud/occ status

sudo -u www-data php /var/www/nextcloud/occ -V

sudo -u www-data php /var/www/nextcloud/occ db:add-missing-primary-keys

sudo -u www-data php /var/www/nextcloud/occ db:add-missing-indices

sudo -u www-data php /var/www/nextcloud/occ db:add-missing-columns

sudo -u www-data php /var/www/nextcloud/occ db:convert-filecache-bigint

sed -i "s/output_buffering=.*output_buffering=0/" /var/www/nextcloud/.user.ini

chown -R www-data:www-data /var/www/nextcloud

redis-cli -s /var/run/redis/redis-server.sock <<EOF

FLUSHALL

quit

EOF

sudo -u www-data php /var/www/nextcloud/occ files:scan --all

sudo -u www-data php /var/www/nextcloud/occ files:scan-app-data

sudo -u www-data php /var/www/nextcloud/occ app:update --all

/usr/sbin/service php8.0-fpm restart

/usr/sbin/service nginx restart

exit 0
```

Markieren Sie das Skript als ausführbar und führen Sie es als privilegierter Benutzer regelmäßig aus.

```
chmod +x /root/upgrade.sh

/root/upgrade.sh
```

Die Installation und Absicherung Ihres Nextcloudservers wurde erfolgreich abgeschlossen und so wünsche ich Ihnen viel Spaß mit Ihren Daten in Ihrer privaten Cloud. Über eine Spende würden sich meine Frau, meine Zwillinge und ich sehr freuen!



9. Optional: Systemüberwachung mit netdata

Laden Sie zuerst weitere Softwarekomponenten herunter und installieren dann Netdata von git:

```
cd /usr/local/src
```

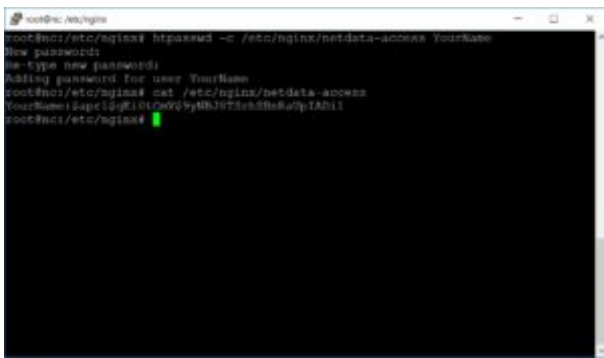
```
apt install -y apache2-utils autoconf automake cmake git gcc libssl-dev libuv1-dev make libtool libjson-c-dev libelf-dev libjson-c-dev pkg-config uuid-dev zlib1g-dev
```

```
git clone https://github.com/firehol/netdata.git --depth=1
```

```
cd netdata
```

Um das Monitoring zu schützen nutzen wir die Apache2-utils und setzen einen Passwortschutz vor Netdata:

```
htpasswd -c /etc/nginx/netdata-access IhrName
```



Nun wird die Installation gestartet

```
./netdata-installer.sh --disable-telemetry -u
```

Nach wenigen Momenten ist Netdata bereits vollständig installiert und lauffähig – es sind dennoch wenige Konfigurationen notwendig:

```
nano /etc/netdata/netdata.conf
```

Ändern Sie den Wert für "history" auf bspw. **14400** (Daten der letzten 4 Stunden werden vorgehalten, benötigt ca. 60 MB RAM) im Bereich [global]:

```
history = 14400
```

Passen Sie zudem den [web] Bereich dahingehend an, dass Netdata nur auf localhost hört:

```
bind to = 127.0.0.1
```

Um nun das Webinterface verwenden zu können wird die bestehende vHost-Datei (nextcloud.conf) erweitert:

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

Fügen Sie die **roten** Zeilen hinzu:

```
[...]
```

```
location ^~ /apps/rainloop/app/data {
```

```

deny all;
}

location /netdata {
return 301 /netdata/;
}

location ~ /netdata/(?<ndpath>.*) {
auth_basic "Bitte Zugangsdaten eingeben";
auth_basic_user_file /etc/nginx/netdata-access;
proxy_http_version 1.1;
proxy_pass_request_headers on;
proxy_set_header Connection "keep-alive";
proxy_store off;
proxy_pass http://netdata/$ndpath$is_args$args;
gzip on;
gzip_proxied any;
gzip_types *;
}

[...]
```

Erweitern Sie den Webserver nginx um seine integrierte Statusfunktionen – legen Sie dazu einen neuen vHost an (/etc/nginx/conf.d/stub_status.conf)

```
touch /etc/nginx/conf.d/stub_status.conf && nano /etc/nginx/conf.d/stub_status.conf
```

und kopieren alle Zeilen hinein:

```

server {
listen 127.0.0.1:80 default_server;
server_name 127.0.0.1;
location /stub_status {
stub_status on;
allow 127.0.0.1;
deny all;
}
}
```

Abschließend wird die Webserverkonfiguration (/etc/nginx/nginx.conf) um die **roten** Zeilen erweitert, so dass Netdata im bestehenden Webserver aufgerufen werden kann:

```
nano /etc/nginx/nginx.conf
```

```

[...]
```

```

http {
server_names_hash_bucket_size 64;
upstream netdata {
server 127.0.0.1:19999;
```

```
keepalive 64;
```

```
}
```

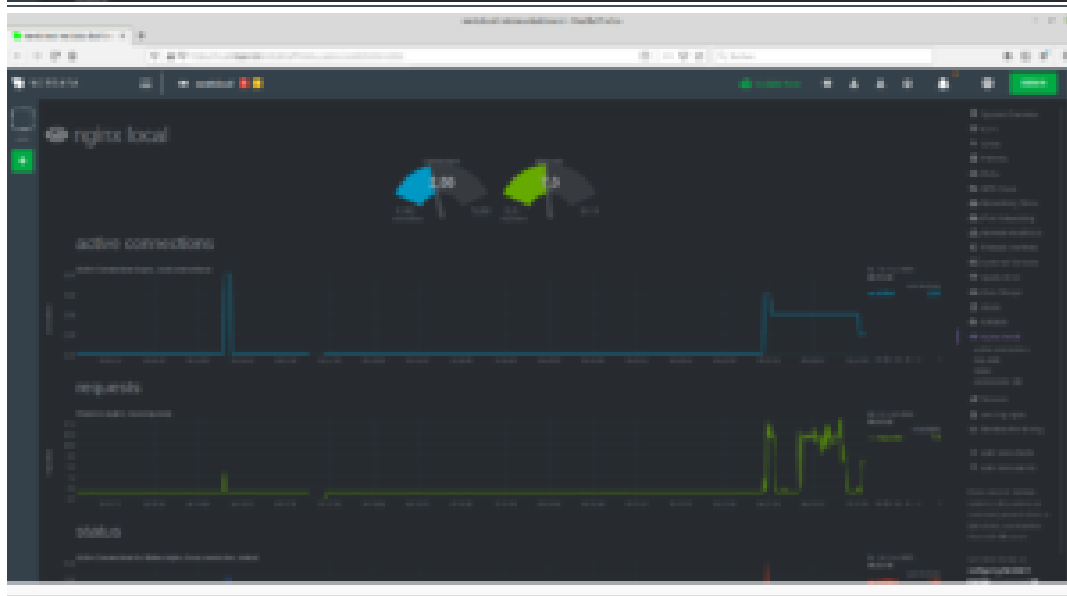
```
[...]
```

Nach einem abschließenden Neustart der Netdata- und Webserver-Dienste

```
service netdata restart && service nginx restart
```

können Sie Netdata bereits nutzen und Ihr System analysieren:

<https://ihre.domain.de/netdata>





Netdata: Netdata is an all-in-one monitoring solution, expertly crafted with a blazing-fast C core, flanked by hundreds of collectors. Featuring a comprehensive dashboard with thousands of metrics, extreme performance and configurability, it is the ultimate single-node monitoring tool [...]

Um Netdata zu aktualisieren genügt es, folgendes Skript auszuführen

```
/usr/libexec/netdata/netdata-updater.sh
```

```
root@nextcloud:/usr/local/src/netdata# /usr/libexec/netdata/netdata-updater.sh
Thu Jun  4 06:43:31 CEST 2020 : INFO: Running on a terminal - (this script also supports
Thu Jun  4 06:43:32 CEST 2020 : INFO: Current Version: 001022001198
Thu Jun  4 06:43:32 CEST 2020 : INFO: Latest Version: 001022001198
Thu Jun  4 06:43:32 CEST 2020 : INFO: Newest version (current=001022001198 >= latest=001022001198)
root@nextcloud:/usr/local/src/netdata#
```

Die aktuelle Version wird geprüft und ggf. auf die aktuelle aktualisiert

10. Optional: Nextcloud Speicher erweitern/verschieben

Der Nextcloud Datenspeicher lässt sich relativ einfach erweitern. Möglichkeiten sind NFS (oder Samba (cifs)), HDD/SD und die Nextcloud external storage app. Wie das im einzelnen funktioniert beschreiben die nachfolgenden Beispiele:

10.1 Nextcloud Speicher mittels NAS (nfs) vergrößern:

Zuerst installieren wir die notwendigen Module

```
apt install nfs-common
```

und erweitern dann die fstab

```
cp /etc/fstab /etc/fstab.bak
```

```
nano /etc/fstab
```

um das Laufwerk persistent im System einzubinden:

```
<IP-NFS-SERVER>:/<Freigabename> /<ihr>/<mountpoint> nfs auto,nofail,noatime,nolock,intr,tcp,actimeo=1800 0 0
```

Nach einem erfolgreichen Einhängen mittels

```
chown -R www-data:www-data /<ihr>/<mountpoint>
```

```
mount /<ihr>/<mountpoint>
```

und dem grundlegenden Kopiervorgang muss sowohl die config.php noch angepasst, als auch der Nextcloud Index neu aufgebaut werden. Stoppen Sie dazu zuerst die Nextcloud

```
service php8.0-fpm stop && service nginx stop
```

und editieren dann die config.php hinsichtlich des neuen Datenverzeichnisses:

```
sudo -u www-data nano /var/www/nextcloud/config/config.php
```

```
'cron_log' => true,  
'datadirectory' => '/<ihr>/<mountpoint>',  
'dbtype' => 'mysql',  
'dbname' => 'nextcloud'
```

[...]

```
'datadirectory' => '/<ihr>/<mountpoint>',
```

```
'logfile' => '/<ihr>/<mountpoint>/nextcloud.log',
```

[...]

Kopieren Sie nun das vorherige Datenverzeichnis in das neue Verzeichnis:

```
rsync -av --progress --stats /<altes Datenverzeichnis> /<ihr>/<mountpoint>
```

Sobald dieser Kopiervorgang abgeschlossen ist wird der Nextcloud Index neu aufgebaut:

```
service nginx stop && service php8.0-fpm stop
```

```
redis-cli -s /var/run/redis/redis-server.sock
```

```
FLUSHALL
```

```
quit
```

```
sudo -u www-data php /var/www/nextcloud/occ files:scan --all -v
```

```
sudo -u www-data php /var/www/nextcloud/occ files:scan-app-data -v
```

```
service php8.0-fpm start && service nginx start
```

Nach dem erfolgreichen Neuaufbau des Nextcloud Index

Folders	Files	Elapsed time
2736	70162	00:01:49

Scanning AppData for files

Folders	Files	Elapsed time
4837	9104	00:00:28

stehen Ihnen die Daten unter Nutzung des NFS Shares bereits zur Verfügung. Planen Sie die Daten sowohl über Nextcloud als auch über das Share direkt bearbeiten zu können, so sollten Sie den Parameter

'filesystem_check_changes' => 1,

in der config.php setzen. Dieser sorgt dafür, dass unabhängig wo die Daten zuletzt bearbeitet wurden, die Nextcloud file app stets synchron zum NFS (also aktuell) ist.

10.2 Nextcloud Speicher mittels weiterer HDD/SSD erweitern

Nehmen wir an, die neue Festplatte kann unter '/dev/sda' für Nextcloud eingebunden werden. Wir formatieren diese HDD/SSD mit dem Dateisystem 'ext4' und binden sie persistent am System (/etc/fstab) ein. Fangen wir an und stoppen zuerst den Nextcloud Server:

```
service nginx stop
service php8.0-fpm stop
service redis-server stop
service mysql stop
```

Nun überprüfen wir die Verfügbarkeit des neuen Laufwerks am Server

```
fdisk -l
```

und partitionieren es wie folgt

(**Annahme:** Die neue Festplatte ist unter /dev/sda verfügbar):

```
fdisk /dev/sda
```

1. Wählen Sie '**o**' um eine neue Partitionstabelle zu erzeugen
2. Wählen Sie '**n**' um eine neue Partition zu erstellen
3. Wählen Sie '**p**' (primary partition type), also eine primäre Partition
4. Wählen Sie die Partitions-Nummer: **1**
5. Weitere Eingaben können mit der ENTER-Taste ohne weitere Angaben, also mit den Standardwerten übernommen werden <**Enter**>
6. Schreiben Sie die Konfiguration fest: '**w**' und drücken <**ENTER**>

Die neue Partition '/dev/sda**1**' wurde bereits erzeugt und muss nur noch formatiert werden:

```
mkfs.ext4 /dev/sda1
fdisk -s /dev/sda1
```

Nun erstellen wir ein neues Verzeichnis '/nc_diskdata' und hängen die neue Partition '/dev/sda**1**' ein:

```
mkdir -p /nc_data
chown -R www-data:www-data /nc_data
```

[illegible]

Das persistente Einhängen erfolgt in der fstab:

```
cp /etc/fstab /etc/fstab.hd.bak
```

```
nano nano /etc/fstab
```

Fügen Sie am Ende folgende Zeile hinzu:

```
/dev/sda1 /nc data ext4 defaults 0 1
```

```
root@ntfsdroid:~#
# UNCONFIGURED PTAB FOR BASE SYSTEM
LABEL=boot /media/boots vfat defaults 0
UUID=133ce78-9841-42fe-8823-94a304a09559 / ext4 defaults 0 0
UUID=133ce78-9841-42fe-8823-94a304a09559 / ext4 errors=remount-ro,nosime 0 1
tmpfs /tmp tmpfs defaults,nosime,nosuid,nodv,noswac,mode=1777 0 0
tmpfs /var/tmp tmpfs defaults,nosime,nosuid,nodv,noswac,mode=1777 0 0
/dev/sdal /no_data ext4 defaults 0 1
```

Nun führen wir den folgenden Befehl aus, um das Laufwerk einzubinden:

mount -a

Ein Blick in das Dateisystem zeigt uns bereits die neue Platte im System:

df -h

```
root@ecscloudi-4 ~# df -h
Filesystem      Size  Used Avail Use% Mounted on
udev            739M    0  739M   0% /dev
tmpfs           172M    0   172M   0% /run
/dev/mmcblk0p2  7.1G  1.9G   5.1G  28% /
tmpfs           550M    0   550M   0% /dev/shm
tmpfs           5.0M    0   5.0M   0% /run/lock
tmpfs           5.0M    0   5.0M   0% /sys/fs/cgroup
tmpfs           838M    0   838M   0% /tmp
tmpfs           650M    0   650M   0% /tmp
/dev/mmcblk0p1 128M    18M  110M  14% /media/iso
/dev/sda1       459G    59G   400G  13% /_co_data
root@ecscloudi-4 ~#
```

Nun überführen wir die Bestandsdaten noch in das neue Verzeichnis

(**Annahme:** Ihre Nextcloud Daten lagen bisher unter /var/nc_data):

```
rsync -av /var/nc data/ /nc data
```

```
root@kali:~# curl -s https://www.vulnhub.com/ -o /var/www/html/vuln_data/
sending incremental file list

  appstore
  appstore
  index.html
nextcloud.jpg
appdata_appdata_qfrvcg/
appdata_appdata_qfrvcg/appstore/
appdata_appdata_qfrvcg/appstore/apps.json
appdata_appdata_qfrvcg/appstore/categories.json
appdata_appdata_qfrvcg/entire/
appdata_appdata_qfrvcg/entire/cinematic/
appdata_appdata_qfrvcg/entire/
appdata_appdata_qfrvcg/sharing/
appdata_appdata_qfrvcg/sharing/0/
appdata_appdata_qfrvcg/sharing/0/conn-ore-filetypes_applications-gif.svg
appdata_appdata_qfrvcg/sharing/0/conn-ore-filetypes_files.svg
appdata_appdata_qfrvcg/sharing/0/conn-ore-filetypes_videos.svg
cinematic/
cinematic/actors/
cinematic/films/
cinematic/films/Seasonal/Monster.pdf
cinematic/films/Seasonal/My4
cloudcraft/films/Documenta/
cloudcraft/films/Documenta/About.sdm
cloudcraft/films/Documenta/About.tan
cloudcraft/films/Photos/
cloudcraft/films/Photos/Chaos.jpg
cloudcraft/films/Photos/Hummingbird.jpg
cloudcraft/films/Photos/War.jpg
files_external/
files_external/nextclouda.jpg

root 7,784,997 bytes received 818 bytes 13,524,850.00 bytes/sec
total time is 7.764,970 seconds on 1 line
```

und passen die Nextcloud config.php hinsichtlich des neuen Datenverzeichnisses an:

```
sudo -u www-data nano /var/www/nextcloud/config/config.php
```

Ändern Sie es wie folgt:

```
...
'datadirectory' => '/nc_data',
...
[...]
'datadirectory' => '/nc_data',
'logfile' => '/nc_data/nextcloud.log',
[...]
```

Zum Abschluß starten wir die zuvor beendeten Dienste neu und führen einen Indizierungslauf durch:

```
service php8.0-fpm start
service redis-server start
service mysql start
cd /var/www/nextcloud
redis-cli -s /var/run/redis/redis-server.sock
FLUSHALL
quit
sudo -u www-data php occ files:scan --all -v
sudo -u www-data php occ files:scan-app-data -v
service nginx restart
```

```
root@kali:~# cd /var/www/html/
root@kali:~/var/www/html# sudo -u www-data php onc_files.php --all -v
Scanning www for size 1 out of 1 (cloudbeef)
Folder /cloudbeef/
Folder /cloudbeef/oxide
Folder /cloudbeef/files
File /cloudbeef/files/Meecloud.apt
File /cloudbeef/files/Meecloud Personal.pdf
Folder /cloudbeef/files/Pixins
File /cloudbeef/files/Pixins/Custom.jpg
File /cloudbeef/files/Pixins/Customize.jpg
File /cloudbeef/files/Pixins/Wat.jpg
Folder /cloudbeef/files/Comments
File /cloudbeef/files/Comments/About.txt
File /cloudbeef/files/Comments/About.com
=====
| Folders | Files | Elapsed time |
=====
| 5 | 7 | 00:00:00 |
=====
root@kali:~/var/www/html#
```

Ab sofort steht Ihnen die gesamte Kapazität der neuen Festplatte für Ihre Nextcloud zur Verfügung.

10.3 Nextcloud Speicher mittels „External Storage“-App erweitern

In Ergänzung zu den Kapiteln [10.1](#) und [10.2](#) lässt sich der Nextcloud Speicher auch mittels der Nextcloud eigenen App „external storage“ erweitern.

So ist es möglich, ohne Komplikationen auf verschiedenen Speichermedien zuzugreifen:

- Dateien können „Out-of-the-Box“ neu erstellt, bearbeitet und gelöscht werden – sowohl innerhalb, als auch außerhalb der Nextcloud und werden dabei stets synchron gehalten,
- Sie können weitere Laufwerke und Shares als zusätzlichen Nextcloud Speicher bereitstellen,
- Sie können Benutzer erlauben, Ihre eigenen Devices als externen Speicher zu nutzen,
- ...

External storages

External storage enables you to mount external storage services and devices as secondary Nextcloud storage devices. You may also allow users to mount their own external storage services.

Folder name	External storage	Authentication	Configuration	Available for
☒ /	Local	None	/external_files	☐

Add storage

Amazon S3

FTP

Local

Nextcloud

OpenStack Object Storage

SFTP

SMB / CIFS

WebDAV

☐ Allow users to mount external storage

Global credentials
Global credentials can be used to authenticate with multiple external storages that have the same credentials.

External storages

External storage enables you to mount external storage services and devices as secondary Nextcloud storage devices. You may also allow users to mount their own external storage services.

Folder name	External storage	Authentication	Configuration	Available for
☒ /	Local	None	/external_files	☐

Add storage

☐ Allow users to mount external storage

Global credentials
Global credentials can be used to authenticate with multiple external storages that have the same credentials.

☒ Enable previews
☐ Enable sharing
☒ Check for changes
☐ Compatibility with Mac NFD encoding (slow)
☐ Read only

Weiterführende Dokumentationen zu dieser App finden Sie [hier](#).

11. Nextcloud High Performance Backend für Dateien

Wir beginnen mit der Einrichtung des High Performance Backends für Dateien und wechseln dafür in den Nextcloud App Store. In der Kategorie **Werkzeuge** findet man die App **Client Push**.



Nach der Installation und Aktivierung über den App Store sind die Tätigkeiten in der Nextcloud-Oberfläche bereits abgeschlossen. Weiter geht es hier auf der Kommandozeile des Servers. Als erstes braucht der virtuelle Host für Nextcloud eine kleine Erweiterung

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

Am Ende der Datei fügen wir die roten Zeilen hinzu:

[...]

```
location / {
```

```

    try_files $uri $uri /index.php$request_uri;
}

location /push/ {

    proxy_pass http://localhost:7867/;

    proxy_http_version 1.1;

    proxy_set_header Upgrade $http_upgrade;
    proxy_set_header Connection "Upgrade";

    proxy_set_header Host $host;

    proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
}
}

```

Testen Sie die Konfiguration und starten dann den Webserver neu

```
nginx -t && service nginx restart
```

Das Setup des High Performance Backends für Dateien wird dann über OCC aufgerufen:

```
cd /var/www/nextcloud
```

```
sudo -u www-data php occ notify_push:setup
```

Sofern die die Konfiguration valide ist, werden weitere Anweisungen angezeigt, um eine systemd Unit für das Nextcloud High Performance Backend für Dateien anzulegen.

```

You're installing the Nextcloud High Performance Backend for Files.
This setup wizard is intended for use on single server instances.
When the Nextcloud server, web server/external proxy and push daemon all run on the same machine.
If your setup is more complex or involves any kind of load balancing
you should follow the manual setup instructions on the README instead.
https://github.com/nextcloud/notify_push
Please refer to README or FAQ to learn more...

Allowing self-signed certificates in the push config.
Place the following systemd unit at /etc/systemd/system/notify_push.service

[Unit]
Description = Push daemon for Nextcloud clients

[Service]
Environment=PORT=7867
Environment=NEXTCLOUD_URL=https://www.yourdomain.de
Environment=ALLOW_SELF_SIGNED=yes
ExecStart=/var/www/nextcloud/apps/notify_push/bin/x86_64/notify_push /var/www/nextcloud/config/config.php
User=www-data

[Install]
WantedBy = multi-user.target

And run sudo systemctl enable --now notify_push
Please refer to README or FAQ to learn more...

```

Dazu öffnen wir eine weitere, zusätzliche SSH-Session und erstellen den Service:

```
nano /etc/systemd/system/notify_push.service
```

[Unit]

Description = Push daemon for Nextcloud clients

[Service]

Environment=PORT=7867

Environment=NEXTCLOUD_URL=https://ihre.domain.de

ExecStart=/var/www/nextcloud/apps/notify_push/bin/x86_64/notify_push /var/www/nextcloud/config/config.php

User=www-data

[Install]

WantedBy = multi-user.target

```

[Unit]
Description = Push daemon for Nextcloud clients

[Service]
Environment=PORT=7867
Environment=NEXTCLOUD_URL=https://www.yourdomain.de
Environment=ALLOW_SELF_SIGNED=yes
ExecStart=/var/www/nextcloud/apps/notify_push/bin/x86_64/notify_push /var/www/nextcloud/config/config.php
User=www-data

[Install]
WantedBy = multi-user.target

```

Der Service wird anschließend aktiviert und gestartet:

```
systemctl enable --now notify_push
```

Sofern der Dienst korrekt gestartet wurde

```
root@nextcloud:~# systemctl status notify_push.service
● notify_push.service - Push daemon for Nextcloud clients
   Loaded: loaded (/usr/systemd/system/notify_push.service; enabled; vendor preset: enabled)
   Active: active (running) since Wed 2021-02-24 09:10:10 CET; 1h ago
     Main PID: 10481 (notify_push)
       CGroup: /system.slice/notify_push.service
              └─476111 /usr/www/nextcloud/apps/notify_push/nextcloud_notify_push /usr/www/nextcloud/config/config.php

Feb 24 09:10:10(10481) [systemd()] Started Push daemon for Nextcloud clients.
```

wechseln wir in die erste SSH Session zurück und bestätigen den Dialog mit ENTER:

```
root@nextcloud:~# cd /usr/www/nextcloud/bin && ./data.php --url notify_push:active
This setup wizard is intended for use on single server (instances)
where the Nextcloud server, web server/reverse proxy and push daemon all run on the same machine.
If your setup is more complex or involves any kind of load balancing
this wizard follows the manual setup instructions on the wizard instead
https://github.com/nextcloud/notify_push
Press enter to continue or ESC to cancel...

- Allowing self-signed certificates in the push config.
Place the following systemd config at /usr/systemd/system/notify_push.service

[Unit]
Description = Push daemon for Nextcloud clients

[Service]
ExecStart=/usr/bin/php
Environment=HTTPS_PROXY=https://
Environment=HTTPS_SELF_SIGNED=true
ExecStartPre=/usr/www/nextcloud/apps/notify_push/bin/mk_data_dir/notify_push /usr/www/nextcloud/config/config.php
ExecStart=/usr/bin/php

[Install]
WantedBy = multi-user.target

root@nextcloud:~# systemctl enable --now notify_push
Press enter to continue or ESC to cancel...

1. push server seems to be functioning correctly.
2. reverse proxy seems to be configured already.
3. notify_push seems to be setup correctly.
configuration saved
root@nextcloud:~#
```

Sollten app-Updates kommen, so wird ein Serviceneustart benötigt.

```
service notify_push restart
```

Die Einrichtung des High Performance Backends für Dateien ist somit erfolgreich abgeschlossen.

Die Installation und Absicherung Ihres Nextcloudservers wurde erfolgreich abgeschlossen und so wünsche ich Ihnen viel Spaß mit Ihren Daten in Ihrer privaten Cloud.