NEXTCLOUD INSTALLATIONSANLEITUNGEN

Nextcloud 20 Installationsanleitung v. 2.1

von Carsten Rieger | Aktualisiert 11. November 2020

Nextcloud auf Ubuntu 20.04 oder Debian 10.5

Dieser Artikel beschreibt die Installation, Konfiguration und Härtung, das Monitoring sowie einige Erweiterungsmöglichkeiten von Nextcloud auf einem Ubuntu Server 20.04 LTS Focal Fossa bzw. Debian Server 10.5 Buster. Die Installation basiert dabei auf den Komponenten nginx 1.19x mainline, Let's Encrypt TLS 1.3, MariaDB 10.5, PHP 7.4, 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: Mittwoch, 11. November 2020
Die ausführliche Aktualisierungshistorie finden Sie hier.

Inhaltsverzeichnis

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

. . .

Systemvorausetzungen seitens Nextcloud

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

1. Vorbereitungen und Installation des nginx Webserver

Die Ins tionsmedien für den zugrundeliegenden Linux-Server erhalten Sie hier:

Ubuntu 20.04.x LTS:Voraussetzungen
Download-Installationsmedium - SSH wird vorausgesetzt (s. Bild zuvor)

Debian 10.5.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:

```
n

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
```

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
```

Speichern Sie diese Datei und starten den Zeitserver neu:

```
systemctl restart systemd-timesyncd
```

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 hkps://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 hkps://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 Webservers 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 Webservers ist somit bereits abgeschlossen und wir fahren mit der Installation und den Anpassungen von PHP fort.

. . .

2. Installation und Konfiguration von PHP 7.4 (fpm)

Das PHP Repository wurde bereits im vorherigen Kapitel eingerichtet und aktiviert, so dass wir direkt mit der Installation beginnen können.:

```
apt update && apt install -y php7.4-fpm php7.4-gd php7.4-mysql php7.4-curl php7.4-xml
php7.4-zip php7.4-intl php7.4-mbstring php7.4-json php7.4-bz2 php7.4-ldap php7.4-apcu
php7.4-bcmath php7.4-gmp php7.4-imagick php7.4-smbclient imagemagick ldap-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/7.4/fpm/pool.d/www.conf /etc/php/7.4/fpm/pool.d/www.conf.bak
cp /etc/php/7.4/cli/php.ini /etc/php/7.4/cli/php.ini.bak
cp /etc/php/7.4/fpm/php.ini /etc/php/7.4/fpm/php.ini.bak
cp /etc/php/7.4/fpm/php-fpm.conf /etc/php/7.4/fpm/php-fpm.conf.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/7.4/fpm/pool.d/www.conf
sed -i "s/;env\[TMP\] = /env[TMP] = /" /etc/php/7.4/fpm/pool.d/www.conf
sed -i "s/;env\[TMPDIR\] = /env[TMPDIR] = /" /etc/php/7.4/fpm/pool.d/www.conf
sed -i "s/;env\[TEMP\] = /env[TEMP] = /" /etc/php/7.4/fpm/pool.d/www.conf
sed -i "s/;env\[PATH\] = /env[PATH] = /" /etc/php/7.4/fpm/pool.d/www.conf
sed -i "s/pm.max_children =.*/pm.max_children = 120/" /etc/php/7.4/fpm/pool.d/www.conf
sed -i "s/pm.start_servers =.*/pm.start_servers = 12/" /etc/php/7.4/fpm/pool.d
/www.conf
sed -i "s/pm.min_spare_servers =.*/pm.min_spare_servers = 6/" /etc/php/7.4/fpm/pool.d
/www.conf
sed -i "s/pm.max_spare_servers =.*/pm.max_spare_servers = 18/" /etc/php/7.4/fpm/pool.d
/www.conf
sed -i "s/;pm.max_requests =.*/pm.max_requests = 1000/" /etc/php/7.4/fpm/pool.d
/www.conf
```

```
sed -i "s/output_buffering =.*/output_buffering = 'Off'/" /etc/php/7.4/cli/php.ini
sed -i "s/max_execution_time =.*/max_execution_time = 3600/" /etc/php/7.4/cli/php.ini
sed -i "s/max_input_time =.*/max_input_time = 3600/" /etc/php/7.4/cli/php.ini
sed -i "s/post_max_size =.*/post_max_size = 10240M/" /etc/php/7.4/cli/php.ini
sed -i "s/upload_max_filesize =.*/upload_max_filesize = 10240M/" /etc/php/7.4/cli
/php.ini
sed -i "s/;date.timezone.*/date.timezone = Europe\/\Berlin/" /etc/php/7.4/cli/php.ini
```

```
sed -i "s/memory_limit = 128M/memory_limit = 1024M/" /etc/php/7.4/fpm/php.ini
sed -i "s/output_buffering =.*/output_buffering = 'Off'/" /etc/php/7.4/fpm/php.ini
sed -i "s/max_execution_time =.*/max_execution_time = 3600/" /etc/php/7.4/fpm/php.ini
sed -i "s/max_input_time =.*/max_input_time = 3600/" /etc/php/7.4/fpm/php.ini
sed -i "s/post_max_size =.*/post_max_size = 10240M/" /etc/php/7.4/fpm/php.ini
sed -i "s/upload_max_filesize =.*/upload_max_filesize = 10240M/" /etc/php/7.4/fpm
/php.ini
sed -i "s/;date.timezone.*/date.timezone = Europe\/\Berlin/" /etc/php/7.4/fpm/php.ini
sed -i "s/;session.cookie_secure.*/session.cookie_secure = True/" /etc/php/7.4/fpm
/php.ini
sed -i "s/;opcache.enable=.*/opcache.enable=1/" /etc/php/7.4/fpm/php.ini
sed -i "s/;opcache.enable_cli=.*/opcache.enable_cli=1/" /etc/php/7.4/fpm/php.ini
sed -i "s/;opcache.memory_consumption=.*/opcache.memory_consumption=128/" /etc/php
/7.4/fpm/php.ini
sed -i "s/;opcache.interned_strings_buffer=.*/opcache.interned_strings_buffer=8/"
/etc/php/7.4/fpm/php.ini
sed -i "s/;opcache.max_accelerated_files=.*/opcache.max_accelerated_files=10000/"
/etc/php/7.4/fpm/php.ini
sed -i "s/;opcache.revalidate_freq=.*/opcache.revalidate_freq=1/" /etc/php/7.4/fpm
/php.ini
sed -i "s/;opcache.save_comments=.*/opcache.save_comments=1/" /etc/php/7.4/fpm/php.ini
```

```
sed -i '$aapc.enable_cli=1' /etc/php/7.4/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 php7.4-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 Passworts 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 php7.4-redis -y
```

Passen Sie die Rediskonfiguration 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/php7.4-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 default_server;
listen [::]:443 ssl http2 default_server;
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;
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;
location = /robots.txt {
allow all;
log_not_found off;
access_log off;
}
location = /.well-known/carddav {
return 301 $scheme://$host:$server_port/remote.php/dav;
}
location = /.well-known/caldav {
return 301 $scheme://$host:$server_port/remote.php/dav;
}
```

```
client_max_body_size 10240M;
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;
location / {
rewrite ^ /index.php;
}
location ~ ^\/(?:build|tests|config|lib|3rdparty|templates|data)\/ {
deny all;
}
location ~ ^\/(?:\.|autotest|occ|issue|indie|db_|console) {
deny all;
}
location ^~ /apps/rainloop/app/data {
deny all;
}
location ~ ^\/(?:index|remote|public|cron|core\/ajax\/update|status|ocs\/v[12]|updater
\/.+|oc[ms]-provider\/.+|.+\/richdocumentscode\/proxy)\.php(?:$|\/) {
fastcgi_split_path_info ^(.+?\.php)(\/.*|)$;
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 $path_info;
fastcgi_param HTTPS on;
fastcgi_param modHeadersAvailable true;
fastcgi_param front_controller_active true;
fastcgi_pass php-handler;
fastcgi_intercept_errors on;
fastcgi_request_buffering off;
}
location ~ ^\/(?:updater|oc[ms]-provider)(?:$|\/) {
try_files $uri/ =404;
index index.php;
}
location ~ \.(?:css|js|woff2?|svg|gif|map)$ {
try_files $uri /index.php$request_uri;
add_header Cache-Control "public, max-age=15778463";
add_header Strict-Transport-Security "max-age=15768000; includeSubDomains; preload;"
always;
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;
access_log off;
}
location ~ \.(?:png|html|ttf|ico|jpg|jpeg|bcmap|mp4|webm)$ {
try_files $uri /index.php$request_uri;
access_log off;
}
}
```

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

Bitte haben Sie nun Geduld! Das Generieren kann – in Abhängigkeit von der Systemleistung – wenige Minuten bis zu Stunden 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
```

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
```

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 /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
```

```
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
```

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 /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 '/ssl-cert-snakeoil/d' /etc/nginx/conf.d/nextcloud.conf
sed -i s/#\ssl/\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 legen wir folgendes ‚renewal'-Skript an:

```
nano /root/renewal.sh
```

```
#!/bin/bash
sudo -u acmeuser "/home/acmeuser/.acme.sh"/acme.sh --cron --home "/home/acmeuser
/.acme.sh"
/usr/sbin/service nginx stop
/usr/sbin/service mysql restart
/usr/sbin/service redis-server restart
/usr/sbin/service php7.4-fpm restart
/usr/sbin/service nginx restart
exit 0
```

Speicherns Sie dieses Skript und markieren es als „ausführbar".

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

Nun deaktivieren wir den Standard ‚acme'-Cronjob

```
crontab -e -u acmeuser
```

durch das Voranstellen eines ‚#, Zeichens

```
# 44 0 * * * "/home/acmeuser/.acme.sh"/acme.sh --cron --home "/home/acmeuser/.acme.sh"
> /dev/null
```

und legen dann einen neuen Cronjob an

```
crontab -e
```

```
@weekly /root/renewal.sh 2>&1
```

Ab sofort wird wöchentlich nach erneuerbaren SSL-Zertifikaten gesucht und bei einem möglichen Zertifikatsrenewal die SSL-Zertifikate aktualisiert und der Webserver neu gestartet – voll automatisch!

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

```
sudo -u www-data php7.4 /var/www/nextcloud/occ maintenance:install --database "mysql"
--database-name "nextcloud" --database-user "nextcloud" --database-pass "nextcloud"
--admin-user "YourNextcloudAdmin" --admin-pass "YourNextcloudAdminPasssword" --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 “YourNextcloudAdminPasssword” : 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 php7.4 /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 php7.4 /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,
'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' => '/Shares',
'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 php7.4 /var/www/nextcloud/occ app:disable survey_client
```

```
sudo -u www-data php7.4 /var/www/nextcloud/occ app:disable firstrunwizard
```

```
sudo -u www-data php7.4 /var/www/nextcloud/occ app:enable admin_audit
```

```
sudo -u www-data php7.4 /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 php7.4-fpm stop
service mysql restart
service php7.4-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 * * * * php7.4 -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 php7.4 /var/www/nextcloud/occ background:cron
```

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
```

zur Kontrolle: cat /etc/fail2ban/filter.d/nextcloud.conf

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 Parameteren 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*, enspricht 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 entsperrren

```
fail2ban-client set nextcloud unbanip <ip-adresse1> <ip-adresse2> ...<ip-adresseN>
```

Abschließend installieren wir noch eine Firewall, die sogenannte *u*ncomplicated *f*ire*w*all (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
  Netxcloud 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, 30.August 2020

7. Systemmails per msmtp versenden

Aktualisieren Sie ihren Server und installieren Sie msmtp. 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 msmtp msmtp-mta mailutils -y
```

Erstellen Sie Ihre Mailkonfiguration anhand meines nachfolgenden Beispiels – erzeugen Sie beide Konfigurationsdateien (sowohl unter */etc/msmtprc*, als auch unter *~/.msmtprc*) und kopieren in beide folgenden Zeilen hinein:

```
touch /etc/msmtprc && touch /home/<IhrBenutzer>/.msmtprc
```

```
nano /etc/msmtprc
```

und

```
nano /home/<IhrBenutzer>/.msmtprc
```

Passen die roten Werte in beiden Dateien identisch an:

```
defaults
port 587
tls on
tls_starttls on
tls_trust_file /etc/ssl/certs/ca-certificates.crt
#Your Mail:
account ihre@mailadresse.de
#Your SMTP-Server:
host smtp.domain.com
#Mails will be sent from:
from ihre@mailadresse.de
auth on
#Your Mailaccount:
user ihre@mailadresse.de
#Your Password:
password IhR-mAILPasswort
#Default Mailaccount:
account default: ihre@mailadresse.de
aliases /etc/aliases
# find out more about the configuration here: https://marlam.de/msmtp/msmtprc.txt
```

Setzen Sie die Berechtigungen wie folgt:

```
chown <IhrBenutzer>:<IhrBenutzer> /home/<IhrBenutzer>/.msmtprc
chmod 600 /etc/msmtprc && chmod 600 /home/<IhrBenutzer>/.msmtprc
```

Erstellen Sie die mail.rc Datei und fügen Sie die folgende Zeile hinzu:

```
touch /etc/mail.rc
```

```
nano /etc/mail.rc
```

```
set sendmail="/usr/bin/msmtp -t"
```

Nutzen Sie „Logrotate“, um die Logdateien vom System verwalten zu lassen – erstellen Sie dafür die Datei mit folgendem Inhalt:

```
touch /etc/logrotate.d/msmtp
```

```
nano /etc/logrotate.d/msmtp
```

Fügen Sie alle Zeilen hinzu:

```
/var/log/msmtp/*.log {
rotate 12
monthly
compress
missingok
notifempty
}
```

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

```
nano /etc/php/7.4/fpm/php.ini
```

Setzen Sie den *sendmail_path* wie folgt:

```
sendmail_path = "/usr/bin/msmtp -t"
```

und starten dann PHP neu:

```
service php7.4-fpm restart
```

Nun legen Sie die Mail-Aliase zum Versenden von Systemmails fest:

```
nano /etc/aliases
```

Passen Sie die Mailadresse des root- und default-Benutzers an – erweitern Sie die Datei bei Bedarf nach Ihren Wünschen: Passen Sie die Datei nach Ihren Bedürfnissen und Benutzern an:

```
root: ihre@mailadresse.de
<IhrBenutzer>: ihre@mailadresse.de
default: ihre@mailadresse.de
```

Testen Sie nun Ihre Mailserverkonfiguration durch folgenden Einzeiler:

```
echo "Eine Testmail..." | mail -s "Testmail" ihre@mailadresse.de
```

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 Banns 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 Banns, 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 Ubuntus:

```
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 Mailbenachrichtigungen 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 Sktipt 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 php7.4 /var/www/nextcloud/updater/updater.phar
sudo -u www-data php7.4 /var/www/nextcloud/occ status
sudo -u www-data php7.4 /var/www/nextcloud/occ -V
sudo -u www-data php7.4 /var/www/nextcloud/occ db:add-missing-indices
sudo -u www-data php7.4 /var/www/nextcloud/occ db:add-missing-columns
sudo -u www-data php7.4 /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 php7.4 /var/www/nextcloud/occ files:scan --all
sudo -u www-data php7.4 /var/www/nextcloud/occ files:scan-app-data
sudo -u www-data php7.4 /var/www/nextcloud/occ app:update --all
/usr/sbin/service php7.4-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 apache2-utils autoconf automake cmake git gcc libssl-dev libuv1-dev make
pkg-config uuid-dev zlib1g-dev -y
```

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

Um das Monitorring zu schützen nutzen wir die Apapche2-utils und setzen einen Passwortschutz vor Netdata:

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

Nun wird die Installation gestartet

```
./netdata-installer.sh
```

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 / {
rewrite ^ /index.php;
}
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
```

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)), HHD/SD und die Nextcloud external storage app. Wie das im einzelnen funktioniert beschreiben die nachfolgenden Beispiele:

10.1 Nextcloud Speicher mittels NAS (nfs) vergößern:

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 php7.4-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
```

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

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 php7.4-fpm stop
redis-cli -s /var/run/redis/redis-server.sock
FLUSHALL
quit
sudo -u www-data php7.4 /var/www/nextcloud/occ files:scan --all -v
sudo -u www-data php7.4 /var/www/nextcloud/occ files:scan-app-data -v
service php7.4-fpm start && service nginx start
```

Nach dem erfolgreichen Neuaufbau des Nextcloud Index

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 php7.4-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/sda1' 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/sda1' ein:


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

Das persitente 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
```

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
```

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
```

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',
...
```

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

```
service php7.4-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 php7.4 occ files:scan --all -v
sudo -u www-data php7.4 occ files:scan-app-data -v
service nginx restart
```

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 verschieden 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,
- ...

Weiterführende Dokumentationen zu dieser App finden Sie hier.

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!

AUTOR

Carsten Rieger

Carsten Rieger ist ein angestellter Senior IT-Systemengineer und zudem auch als Kleinunternehmer (Freelancer) aktiv. Er arbeitet seit 2005 im Linux- und Microsoftumfeld, ist ein Open Source Enthusiast und hoch motiviert, Linux Installationen und Troubleshooting durchzuführen. Dabei arbeitet er vorrangig mit Debian und Ubuntu Linux, Nginx und Apache Webservern, MariaDB, PHP, Cloud Infrastrukturen (bspw. Nextcloud) und auch vielen anderen Open Source Projekten (bspw. HAProxy, Jitsi, BigBlueButton, CheckMK etc.). Zudem engagiert er sich ehrenamtlich für die Dr. Michael & Angela Jacobi Stiftung - und das schon seit 2012.

32 BEITRÄGE