NextCloud over Tor (onion service)

This guide is about how to set up a nextcloud instance running on a Raspberry Pi and providing the cloud service over Tor (a hidden service on the onion-network).

The initial setup of a new Raspberry Pi is always the same and described in some detail here: https://www.spaetzle.info/raspberry-server/

Install Tor

Let’s start with installing the tor package:

```
sudo apt install tor -y
```

Save the default config file as reference and create a new one:

```
sudo mv /etc/tor/torrc /etc/tor/torrc.default
sudo nano /etc/tor/torrc
```

and past in the following:

```
Log notice file /var/log/tor/notices.log

ExitPolicy reject *:*

TransPort 127.0.0.1:9040
DNSPort   127.0.0.1:5300

AutomapHostsOnResolve 1
AutomapHostsSuffixes .onion,.exit
VirtualAddrNetworkIPv4 10.42.0.0/16

HiddenServiceDir /var/lib/tor/services/nextcloud
HiddenServicePort  80 127.0.0.1:80
HiddenServicePort 443 127.0.0.1:443
```

If your running on an SD-Card (not recommended anyhow; if possible rather use a SSD-drive instead) you should add the following line to the config above:

AvoidDiskWrites 1

A crucial step is to manually create the directory for the hidden service:

sudo -u debian-tor mkdir /var/lib/tor/services/

After changing the config one should check the config, then restart the tor service and check the log file for warnings and errors:

```
sudo -u debian-tor tor --verify-config
sudo systemctl restart tor
cat /var/log/tor/notices.log
```

Firewall (nftables)

First install the firewall frontend and enable the firewall:

```
sudo apt install nftables -y
sudo systemctl enable nftables.service
```

Enable the following firewall rules, starting with a config file in your home directory

nano ~/nftables.conf

and paste in

```
#!/usr/sbin/nft -f

flush ruleset

table ip filter {
    chain input {
        type filter hook input priority 0; policy drop;

        iifname lo accept
```

```
        ct state established,related accept
        ct state invalid drop

        tcp dport ssh ct state new limit rate 10/minute accept
        tcp dport { http, https } ct state new accept

        icmp type echo-request limit rate 1/second accept
    }

    chain forward {
        type filter hook forward priority 0; policy drop;
    }

    chain output {
        type filter hook output priority 0; policy drop;
        oifname lo accept

        ct state established,related accept
        ct state invalid drop

        skuid "debian-tor" accept

        oifname eth0 udp dport ntp accept
        ip daddr 127.0.0.1 counter accept   # not needed ???
          ip daddr { 192.168.178.0/24, 192.168.200.0/24,
255.255.255.255 } accept
    }
}

table ip nat {
    chain input {
        type nat hook input priority 100; policy accept;
    }

    chain output {
        type nat hook output priority -100; policy accept;

        skuid "debian-tor" accept

        udp dport domain redirect to :5300
        ip daddr { 192.168.178.0/24, 192.168.200.0/24 } accept
```

```
        tcp flags & (fin | syn | rst | ack) == syn redirect to
:9040
    }
}
```

and activate these firewall rules with

```
sudo nft -f nftables.conf
```

In case something goes horribly wrong (e.g. you lock ssh sessions) you can hard reboot the server and will start without the firewall rules.

Note that nft uses its own matching of service names to port numbers – to see the list simply type in:

```
nft describe tcp dport
```

Once you're happy with them working make them permanent with copying them to the standard place (enabled on reboot):

```
sudo cp /etc/nftables.conf /etc/nftables.conf.default
sudo cp nftables.conf /etc/nftables.conf
```

Install Nextcloud

Install php

Start by installing php with:

```
sudo apt install -y apache2 mariadb-server libapache2-mod-php php-gd php-json php-mysql php-curl php-mbstring php-intl php-imagick php-xml php-zip php-apcu
```

Prepare MySQL (MariaDB)

To initialize the MariaDB database start with:

```
sudo mariadb_secure_installation
```

and answer the questions accordingly (e.g. remove anonymous user). Now the database is ready and we create a nextcloud-

user in mysql: Log into MariaDB database server with the following command:

```
sudo mariadb -u root
```

Then create a database for NextCcoud using the MariaDB command below. This name of the database could be nextcloud (but one can use whatever name is prefered). Note: Don’t leave out the semicolon at the end.

```
> create database nextcloud;
```

Then create a new user.

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

Again, you can use your preferred name for this user. Replace ‚your-password‘ with your preferred password (leave the single quotes in place):

```
> GRANT ALL PRIVILEGES ON nextcloud.* TO nextcloud@localhost IDENTIFIED BY 'your-password';
```

The above command will create the user and grant all privileges. Now flush MariaDB privileges and exit:

```
> FLUSH PRIVILEGES;
> exit;
```

Install Nextcloud package

To download the files, first get the download link in a browser (on nextcloud.com, download section, server packages), copy the link and then use the wget command (note that the actual filename will change once new versions of nextcloud will be released):

```
wget https://download.nextcloud.com/server/releases/nextcloud-x.y.z.zip
```

and download the checksum (just add „.sha256“ to the above download command):

```
wget
https://download.nextcloud.com/server/releases/nextcloud-x.y.z
.zip.sha256
```

and check it with:

```
sha256sum -c nextcloud-x.y.z.zip.sha256
```

and then unzip the downloaded nextcloud package, copy it to the webserver directory and change the ownership:

```
unzip nextcloud-x.y.z.zip
cp -r nextcloud /var/www
sudo chown -R www-data:www-data /var/www/nextcloud/
```

Enable the apache webserver

First, lets tell apache to list on which IP addresses and which ports:

```
sudo nano /etc/apache2/ports.conf
```

and fill it with something along (but change to your local IP addresses):

```
Listen 127.0.0.1:80 http
Listen 192.168.200.42:80 http


<IfModule ssl_module>
        Listen 127.0.0.1:443 https
        Listen 192.168.200.42:443 https
</IfModule>

<IfModule mod_gnutls.c>
        Listen 127.0.0.1:443 https
        Listen 192.168.200.42:443 https
</IfModule>
```

Next we create a config file for our actual nextcloud instance

```
sudo nano /etc/apache2/sites-available/nextcloud.conf
```

and paste in:

```
ServerName abc.mynet

<VirtualHost 127.0.0.1 192.168.200.22>
        ServerName abc.mynet
                                                ServerAlias
h72qy8dg3rhd55rn7u3zkaibw4598dupq544wrlqsmx4d3oxjxvuurad.onion
        DocumentRoot /var/www/nextcloud/
</VirtualHost>


<Directory /var/www/nextcloud/>
  Options +FollowSymlinks
  AllowOverride All

 <IfModule mod_dav.c>
  Dav off
 </IfModule>

 SetEnv HOME /var/www/nextcloud
 SetEnv HTTP_HOME /var/www/nextcloud

</Directory>
```

To let apache check the config for errors use:

```
sudo apache2ctl configtest
```

Finally, enable this new config together with two required apache modules:

```
sudo a2ensite nextcloud.conf
sudo a2dissite 000-default.conf
sudo a2enmod rewrite
sudo a2enmod headers
sudo a2dismod status
```

Before actually activating the new config we apply a few more things. First some additional measures to improve anonymity:

sudo nano /etc/apache2/conf-enabled/security.conf

and change it so it shows these two configs:

```
ServerTokens Prod
ServerSignature Off
```

Finally activate all changes by restarting apache:

sudo systemctl reload apache2

Fire up nextcloud

Configuration

To connect to the database just point your webbrowser to your new nextcloud server and complete the installation wizard. This also creates the basic config file for nextcloud which we also need to change manually a bit:

sudo nano /var/www/nextcloud/config/config.php

One should add additional so-called trusted domains; here we want to add out onion web-address. To get your new onion address look it up here:

sudo cat /var/lib/tor/services/nextcloud/hostname

so with a few other addional tweaks, part of your config file (not a complete example!) might look like:

```
  'trusted_domains' =>
  array (
    0 => 'localhost',
    1 => '127.0.0.1',
    2 => '192.168.202.44',
    3 => 'xxx.bet',
                                                4            =>
'h9dfype6yrhd55rn7u3dk7ebwhhkgospq544wrlqsmx4d3oxjxvuur99.onio
n',
  ),
  'overwrite.cli.url' => 'http://xxx.bet',
```

```
  'memcache.local' => '\OC\Memcache\APCu',
'htaccess.RewriteBase' => '/',
  'trashbin_retention_obligation' => 'auto,90',
```

Php configuration

The php config should be changed to e.g. accept uploads of larger files (note that the php version number might be different):

```
sudo nano /etc/php/7.3/apache2/php.ini
```

and change (search for the options in this very lengthy config file):

```
memory_limit = 512M
post_max_size = 256M
upload_max_filesize = 256M
```

crontab

You might improve a bit on the nextcloud performance by using cron:

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

and add at the very bottom:

```
*/15  *  *  *  * /usr/bin/php -f /var/www/nextcloud/cron.php
```

Finally, log into nextcloud and on the admin panel enable cron.

Update Nextcloud

Although there is a possibility to update your Nextcloud instance via the web frontend this might be failing in same cases due to time-outs. The safer approach is to simply run:

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

on the command line interface of your machine.