My Reverse Proxy with Raspberry 3B, Raspian Bullseye, NGINX

© Dr.-Ing. Rainer Knausenberger, May 2022

Last Change: 25.05.2022

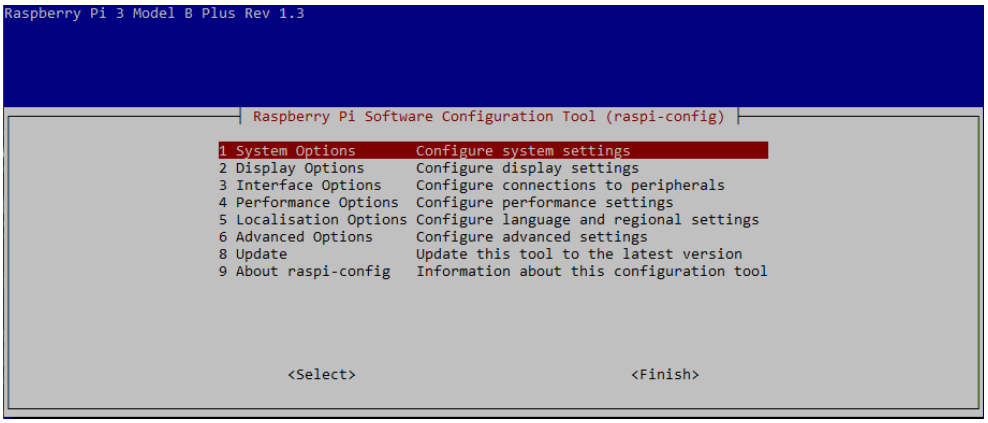
Following domains are available via the Reverse Proxy with http (port 80) and https (port 443):

- **knausenberger.info** (running IceWarp Mailserver, a php-system on Windows Server)
- **knausir.com** (File-Server & Media-Center on Windows Home Server 2011, based on Windows Server 2008 R2)

Every access using **http** is redirected to **https** automatically.

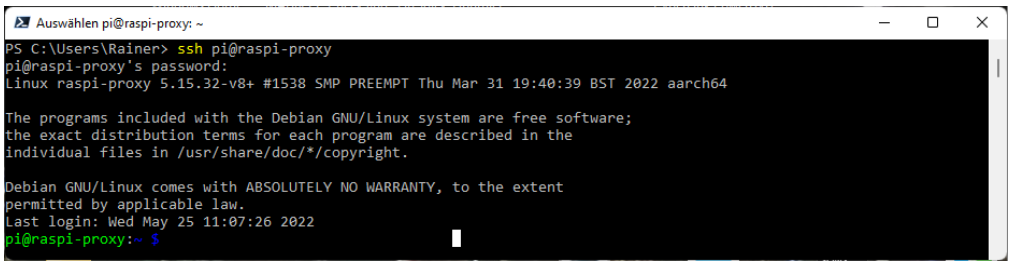
For setup and configuration of the Raspberry 3B you need an OS on SD card. Use Raspberry Pi Imager and select RASPBERRY PI OS LITE (64 BIT). Then write image to SD Card.

Insert the SD card into the Raspberry, connect display, keyboard and mouse and also a network cable. After first startup open raspi-config for



- change of the password for user ´pi´ (´52499Baesweiler´), -> menu item 1, then S3
- change of hostname (´raspi- proxy´), -> menu item 1, then S4
- selection of language, time zone, keyboard and WLAN Country, -> menu item 5
- activate SSH and if needed VNC, -> menu item 3, then I2 (and/or I3)

Now shutdown Raspberry (´shutdown´), then disconnect power supply, display, keyboard and mouse and finally start again with ethernet connected again. Further settings and installations can be done using **Power Shell** (´**SSH**´) from a Windows-Computer in LAN.

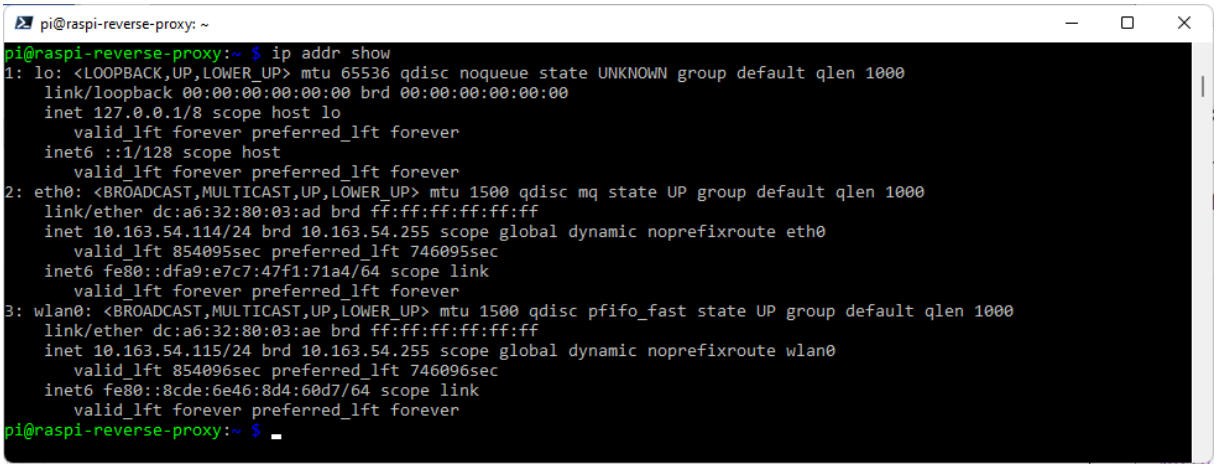


Setup static IP-Address under Raspian Bullseye (Debian 11):

Configure Static IP (optional). Alternatively you can also assign fixed IP address on your router.

First check the actual settings:

```
ip addr show
```



To change IP address from dhcp mode to static you need to do the following:

- Login to raspi-reverse-proxy using these credentials:
  user: pi
  password: 52499Baesweiler (originally: raspberry)

- Open file /etc/dhcpcd.conf (use nano or vim)
  Add at the end of the file:

```
# RASPI-PROXY IP
interface eth0
static ip_address=10.163.54.50/24   <- your static IP address here
static routers=10.163.54.1  <- your default gateway here
static domain_name_servers=10.163.54.1 8.8.8.8   <- your domain name servers (DNS) here
```

Save the file using Strg-x and Y/J.
Reboot raspi- proxy

The ethernet-interface name can be different from *„eth0“* (i.e. for example *„enxb827“*). The given name hast o be used!
The WiFi interface should be deactivated, because ist not needed in the Reverse Proxy.

```
sudo ifconfig wlan0 down
```

The next steps I have taken from URL https://linuxize.com/post/secure-nginx-with-let-s-encrypt-on-debian-10/ (valid also for Debian 11):

- Begin of take over from linuxize.com –

This tutorial shows how to install a free Let’s Encrypt SSL certificate on Debian 10, Buster (Debian 11, Bullseye) running Nginx as a web server and Reverse Proxy. We’ll also show how to configure Nginx to use the SSL certificate and enable HTTP/2.

Prerequisites

Ensure the following prerequisites are met before proceeding with the guide:

- **Logged in as root or user with** sudo privileges.
- The domain for which you want to obtain the SSL certificate **must** point to your public server IP. We'll use `example.com`.
- Nginx installed. (Installation directions you get by pointing with mouse over to beginning of this line and then pressing [Strg]+[Left mouse button].

Installing Certbot

We'll use the certbot tool to obtain and renew the certificates.

Certbot is a fully-featured and easy to use tool that automates the tasks for obtaining and renewing Let's Encrypt SSL certificates and configuring web servers to use the certificates.

The certbot package is included in the default Debian repositories. Run the following commands to install certbot:

```
sudo apt update
sudo apt install certbot
```

Generating DH (Diffie-Hellman) Group

Diffie–Hellman key exchange (DH) is a method of securely exchanging cryptographic keys over an unsecured communication channel.

We're going to generate a new set of 2048 bit DH parameters to strengthen the security:

```
sudo openssl dhparam -out /etc/ssl/certs/dhparam.pem 2048
```

You can also change the size up to 4096 bits, but the generation may take more than 30 minutes depending on the system entropy.

Obtaining a Let's Encrypt SSL certificate

To obtain an SSL certificate for the domain, we're going to use the Webroot plugin. It works by creating a temporary file for validating the requested domain in the `${webroot-path}/.well-known/acme-challenge` directory. The Let's Encrypt server makes HTTP requests to the temporary file to validate that the requested domain resolves to the server where certbot runs.

We're going to map all HTTP requests for `.well-known/acme-challenge` to a single directory, `/var/lib/letsencrypt`.

Run the following commands to create the directory and make it writeable for the Nginx server:

```
sudo mkdir -p /var/lib/letsencrypt/.well-known
sudo chgrp www-data /var/lib/letsencrypt
```

```
sudo chmod g+s /var/lib/letsencrypt
```

To avoid duplicating code, we'll create two snippets that will be included in all Nginx server block files.

Open your text editor and create the first snippet, letsencrypt.conf:

```
sudo nano /etc/nginx/snippets/letsencrypt.conf
```

/etc/nginx/snippets/letsencrypt.conf

```
location ~ /.well-known/acme-challenge/ {
  allow all;
  root /var/lib/letsencrypt/;
  default_type "text/plain";
  try_files $uri =404;
}
```

The second snippet ssl.conf includes the ciphers recommended by Mozilla, enables OCSP Stapling, HTTP Strict Transport Security (HSTS), and enforces few security-focused HTTP headers.

```
sudo nano /etc/nginx/snippets/ssl.conf
```

/etc/nginx/snippets/ssl.conf

```
ssl_dhparam /etc/ssl/certs/dhparam.pem;

ssl_session_timeout 1d;
ssl_session_cache shared:SSL:10m;
ssl_session_tickets off;

ssl_protocols TLSv1.2 TLSv1.3;
ssl_ciphers ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-RSA-AES128-GCM-SHA256:ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-GCM-SHA384:ECDHE-ECDSA-CHACHA20-POLY1305:ECDHE-RSA-CHACHA20-POLY1305:DHE-RSA-AES128-GCM-SHA256:DHE-RSA-AES256-GCM-SHA384;
ssl_prefer_server_ciphers off;

ssl_stapling on;
ssl_stapling_verify on;
resolver 8.8.8.8 8.8.4.4 valid=300s;
resolver_timeout 30s;

add_header Strict-Transport-Security "max-age=63072000" always;
add_header X-Frame-Options SAMEORIGIN;
add_header X-Content-Type-Options nosniff;
```

Once done, open the domain server block file and include the letsencrypt.conf snippet as shown below:

```
sudo nano /etc/nginx/sites-available/example.com.conf
```

/etc/nginx/sites-available/example.com.conf

```
server {
  listen 80;
  listen [::]:80;
  server_name knausir.com www.knausir.com;

  include snippets/letsencrypt.conf;
}
```

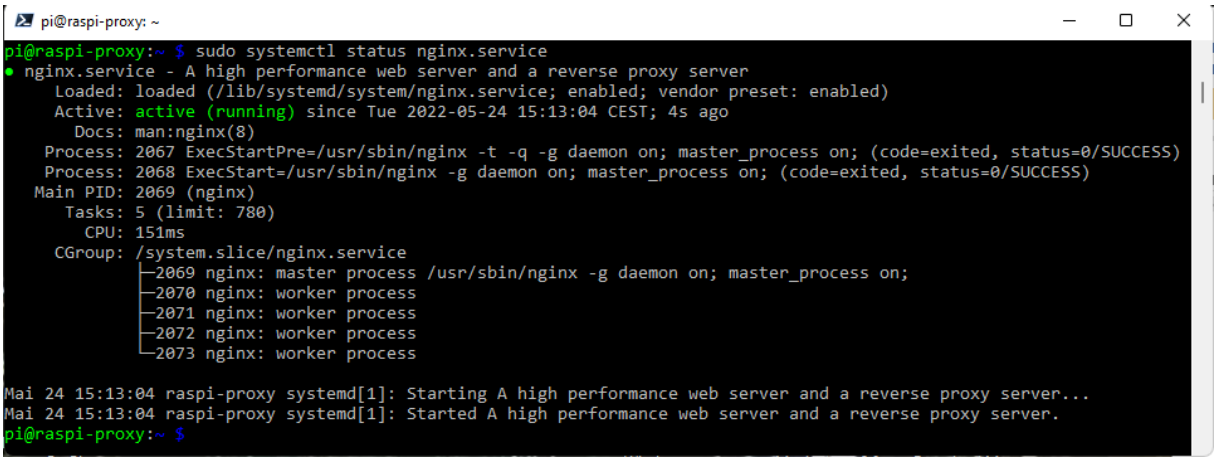
Create a symbolic link to the sites-enabled directory to enable the domain server block:

```
sudo ln -s /etc/nginx/sites-available/example.com.conf /etc/nginx/sites-enabled/
```

Restart the Nginx service for the changes to take effect:

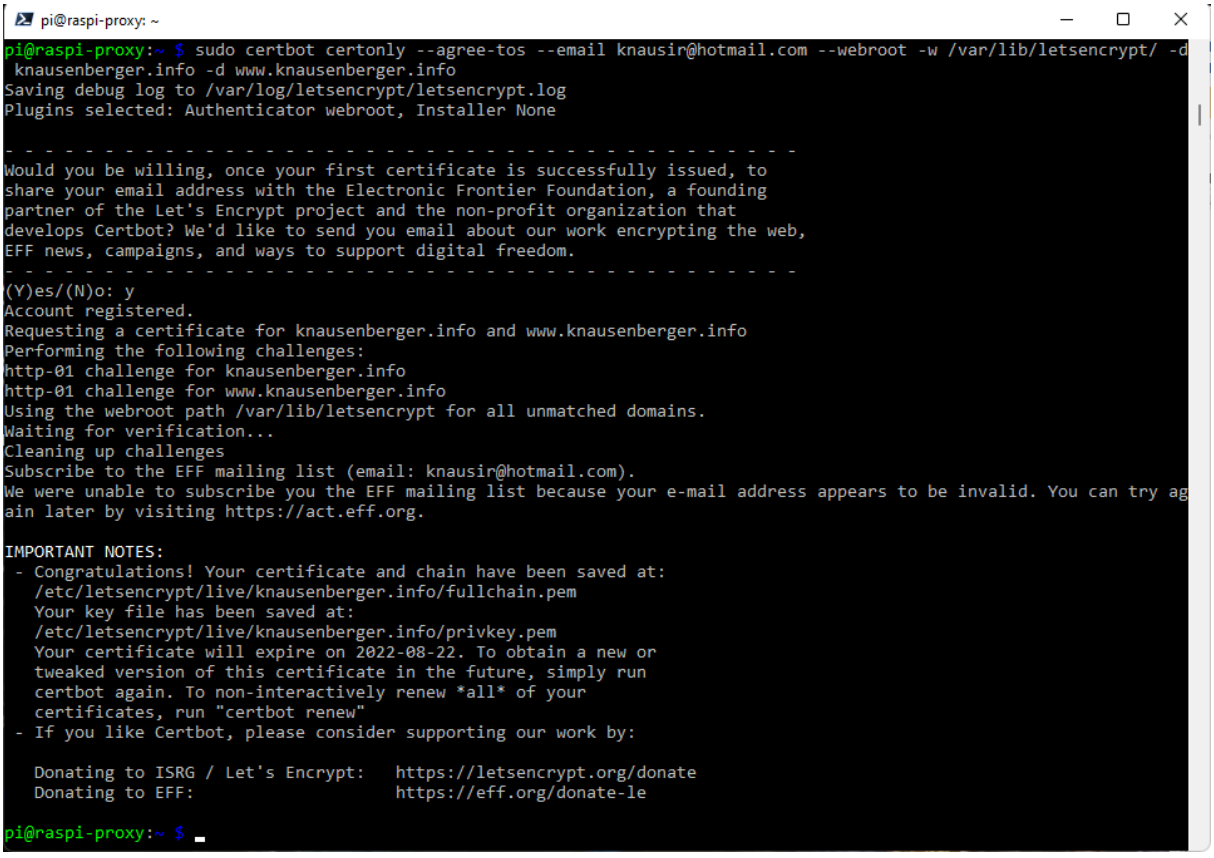
The file example.com.conf **must** contain a valid domain name (here knausir.com) for testing. Otherwise you will get an error.

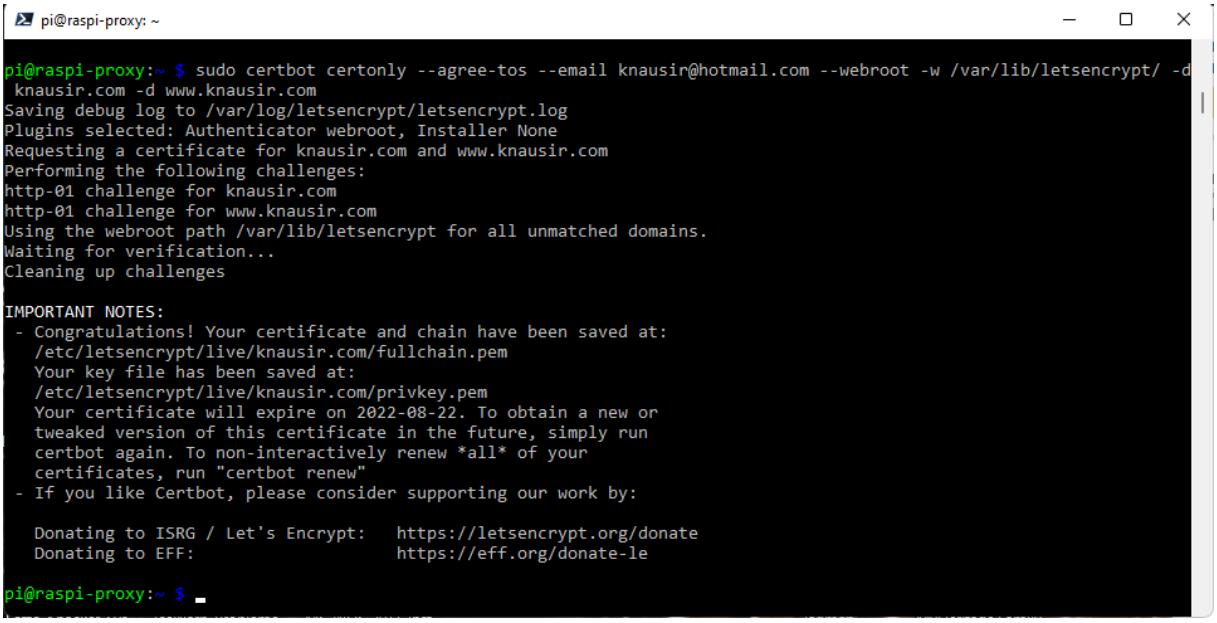
```
sudo systemctl restart nginx
```



You're now ready to obtain the SSL certificate files by running the following command:

```
sudo certbot certonly --agree-tos --email admin@example.com --webroot -w
/var/lib/letsencrypt/ -d example.com -d www.example.com
```





If the SSL certificate is successfully obtained, the following message will be printed on your terminal:

```
IMPORTANT NOTES:
 - Congratulations! Your certificate and chain have been saved at:
   /etc/letsencrypt/live/example.com/fullchain.pem
   Your key file has been saved at:
   /etc/letsencrypt/live/example.com/privkey.pem
   Your cert will expire on 2022-08-22. To obtain a new or tweaked
   version of this certificate in the future, simply run certbot
   again. To non-interactively renew *all* of your certificates, run
   "certbot renew"
 - If you like Certbot, please consider supporting our work by:

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

The SSL certificate generation process ends with 4 certificate files:

cert.pem
chain.pem
fullchain.pem
privkey.pem

in the /etc/letsencrypt/live/example.*com*/ folder.

Edit the domain server block and include the SSL certificate files as follows:

```
sudo nano /etc/nginx/sites-available/example.com.conf
```

/etc/nginx/sites-available/example.com.conf

```
server {
    listen 80;
    listen [::]:80;
    server_name www.example.com example.com;

    include snippets/letsencrypt.conf;
    return 301 https://$host$request_uri;
}

server {
    listen 443 ssl http2;
    listen [::]:443 ssl http2;
    server_name www.example.com;
```

```
    ssl_certificate /etc/letsencrypt/live/example.com/fullchain.pem;
    ssl_certificate_key /etc/letsencrypt/live/example.com/privkey.pem;
    ssl_trusted_certificate /etc/letsencrypt/live/example.com/chain.pem;
    include snippets/ssl.conf;
    include snippets/letsencrypt.conf;

    return 301 https://example.com$request_uri;
}

server {
    listen 443 ssl http2;
    listen [::]:443 ssl http2;
    server_name example.com;

    ssl_certificate /etc/letsencrypt/live/example.com/fullchain.pem;
    ssl_certificate_key /etc/letsencrypt/live/example.com/privkey.pem;
    ssl_trusted_certificate /etc/letsencrypt/live/example.com/chain.pem;
    include snippets/ssl.conf;
    include snippets/letsencrypt.conf;

    # . . . other code
    location / {
      proxy_pass http://127.0.0.1:3000;
      proxy_http_version  1.1;
      proxy_cache_bypass  $http_upgrade;
      proxy_headers_hash_max_size 512;

      proxy_set_header Upgrade           $http_upgrade;
      proxy_set_header Connection        "upgrade";
      proxy_set_header Host              $host;
      proxy_set_header X-Real-IP         $remote_addr;
      proxy_set_header X-Forwarded-For   $proxy_add_x_forwarded_for;
      proxy_set_header X-Forwarded-Proto $scheme;
      proxy_set_header X-Forwarded-Host  $host;
      proxy_set_header X-Forwarded-Port  $server_port;
    }

}
```

The configuration above tells Nginx to redirect from HTTP to HTTPS and from www to non-www version.

Restart or reload the Nginx service for the changes to take effect:

```
sudo systemctl restart nginx
```

Open your website using `https://`, and you'll notice a green lock icon.

If you test your domain using the SSL Labs Server Test, you'll get an A+ grade.

The next 2 lines are added by me.

Now repeat this for every domain (example2.com, example3.com …) you wish to run behind Reverse Proxy.

Auto-renewing Let's Encrypt SSL certificate

Let's Encrypt's certificates are valid for 90 days. To automatically renew the certificates before they expire, the certbot package creates a cronjob that runs twice a day and automatically renews any certificate 30 days before its expiration.

On renewal the nginx service must be reloaded for the server to load the certificate. Append `--renew-hook "systemctl reload nginx"` to the `/etc/cron.d/certbot` file so as it looks like this:

```
sudo nano /etc/cron.d/certbot
```

/etc/cron.d/certbot

```
0 */12 * * * root test -x /usr/bin/certbot -a \! -d /run/systemd/system && perl -e 'sleep int(rand(3600))' && certbot -q renew --renew-hook "systemctl reload nginx"
```

Test the automatic renewal process, by running this command:

```
sudo certbot renew --dry-run
```

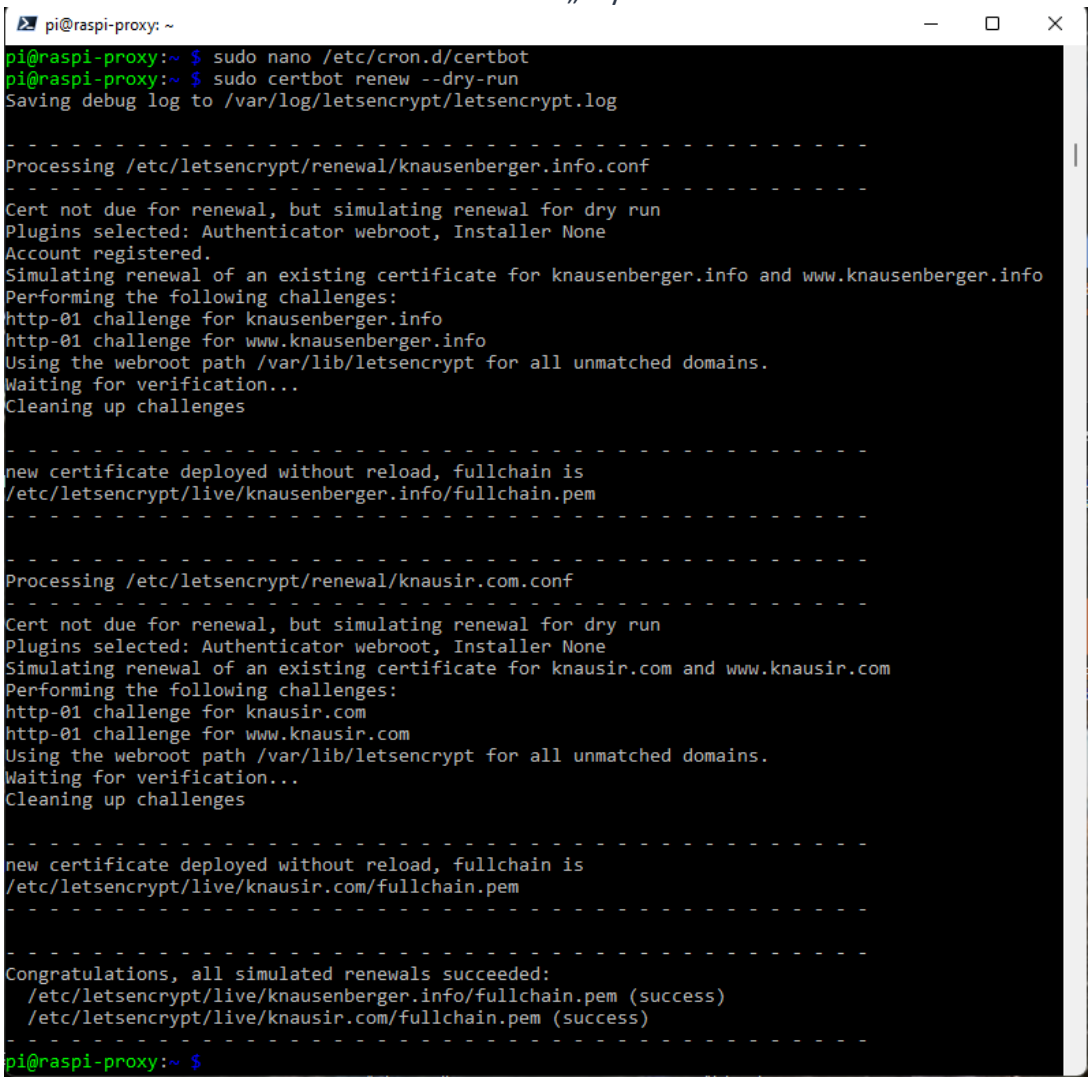
If there are no errors, it means that the renewal process was successful.

Conclusion

Having an SSL certificate is a must nowadays. It secures your website, increases SERP ranking position, and allows you to enable HTTP/2 on your web server.

- End of take over from linuxize.com –

Result of the „dry-run“:



You can see, every domain´s certificates defined in /etc/nginx/sites-enabled/ are renewed.

Finally we check whether nginx is running without any errors:
sudo systemctl restart nginx
sudo systemctl status nginx.service

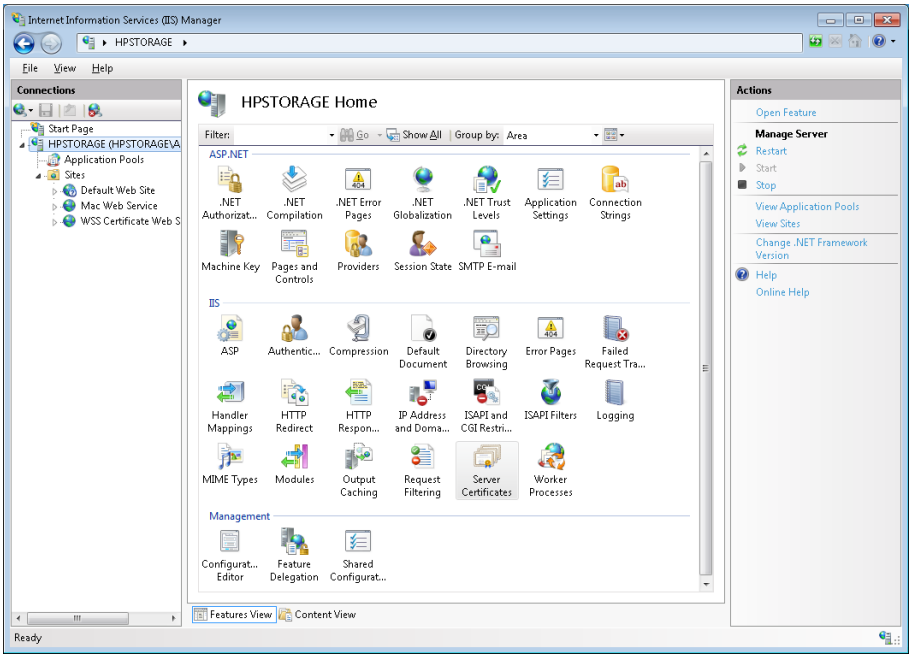
```
pi@raspi-proxy: ~
pi@raspi-proxy:~ $ sudo systemctl restart nginx
pi@raspi-proxy:~ $ sudo systemctl status nginx.service
● nginx.service - A high performance web server and a reverse proxy server
     Loaded: loaded (/lib/systemd/system/nginx.service; enabled; vendor preset: enabled)
     Active: active (running) since Tue 2022-05-24 15:56:00 CEST; 46s ago
       Docs: man:nginx(8)
    Process: 2139 ExecStartPre=/usr/sbin/nginx -t -q -g daemon on; master_process on; (code=exited, status=0/SUCCESS)
    Process: 2140 ExecStart=/usr/sbin/nginx -g daemon on; master_process on; (code=exited, status=0/SUCCESS)
   Main PID: 2141 (nginx)
      Tasks: 5 (limit: 780)
        CPU: 192ms
     CGroup: /system.slice/nginx.service
             ├─2141 nginx: master process /usr/sbin/nginx -g daemon on; master_process on;
             ├─2142 nginx: worker process
             ├─2143 nginx: worker process
             ├─2144 nginx: worker process
             └─2145 nginx: worker process

Mai 24 15:55:59 raspi-proxy systemd[1]: Starting A high performance web server and a reverse proxy server...
Mai 24 15:56:00 raspi-proxy systemd[1]: Started A high performance web server and a reverse proxy server.
pi@raspi-proxy:~ $
```

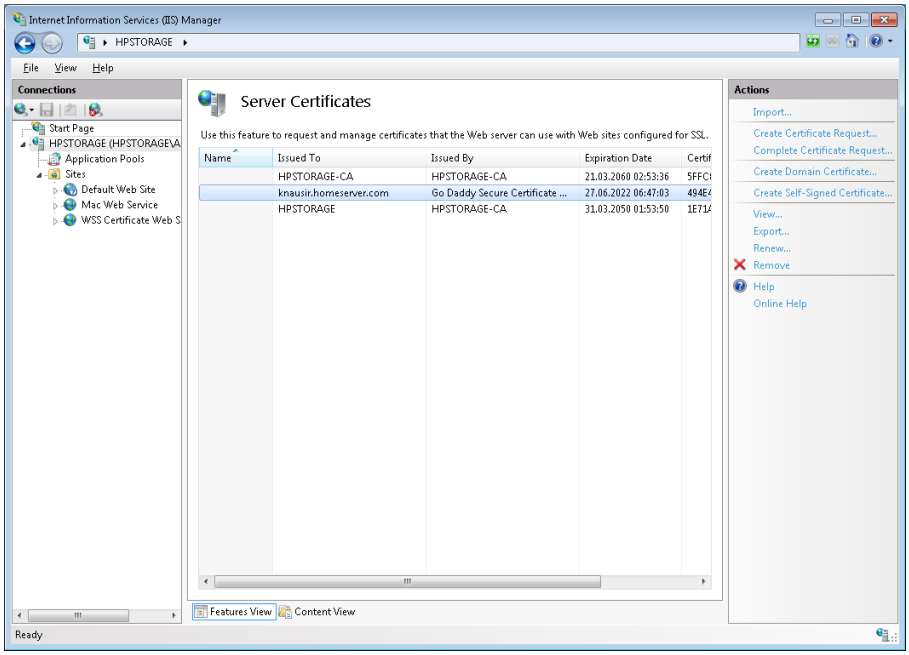
And now we are there, where we want to be!

Dealing with Windows Server certificates on Reverse Proxy

Microsoft Windows Server uses built-in certificates for SSL and inside domain controller in pfx-format. This certificate has to be copied to Reverse Proxy and then to be made useful for NGINX. First you have to open the IIS Manager:



Open „Server Certificates“.



Select the domain certificate (knausir.homeserver.com).
Click on „Export…“. Define the path and password. I put it in C:\Users\Rainer\Desktop\Homeserver_CA\.
Open https://www.sslshopper.com/ssl-converter.html

SSL Converter

Use this **SSL Converter to convert SSL certificates** to and from different formats such as **pem, der, p7b, and pfx.** Different platforms and devices require SSL certificates to be converted to different formats. For example, a Windows server exports and imports .pfx files while an Apache server uses individual PEM (.crt, .cer) files. To use the SSL Converter, just select your certificate file and its current type (it will try to detect the type from the file extension) and then select what type you want to convert the certificate to and click **Convert Certificate.** For more information about the different SSL certificate types and how you can convert certificates on your computer using OpenSSL, see below.

Certificate Conversion Options

Certificate File to Convert

Durchsuchen

Type of Current Certificate

PFX/PKCS#12

Type To Convert To

Standard PEM

PFX Password

⚠ Your private key is intended to remain on the server. While we try to make this process as secure as possible by using SSL to encrypt the key when it is sent to the server, for complete security, we recommend that you manually convert the certificate on your server using the OpenSSL commands below.

Convert Certificate

PEM Format

The PEM format is the most common format that Certificate Authorities issue certificates in. PEM certificates usually have extentions such as **.pem, .crt, .cer, and .key.** They are Base64 encoded ASCII files and contain "-----BEGIN CERTIFICATE-----" and "-----END CERTIFICATE-----" statements. Server certificates, intermediate certificates, and private keys can all be put into the PEM format.

Click on „Durchsuchen“ and point to the homeserver’s .pfx-file. Select the format of the homeserver certificate (PFX) and the format to convert to (PEM). Also enter the password defined during export on IIS and then click on „Convert Certificate“. Store the file in the same folder as the .pfx. file.

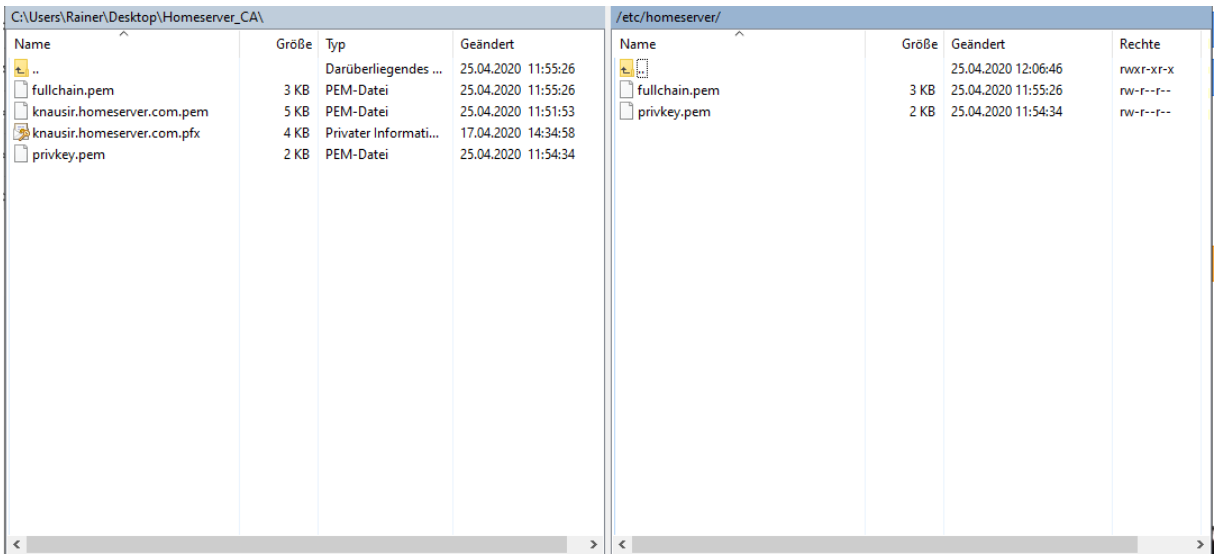
Using notepad or editor you now devide this .PEM file into 2 separate files „fullchain.pem“ and „privkey.pem“.

The content of **fullchain.pem** looks like:

and the content of **privkey.pem** looks like

```
-----BEGIN PRIVATE KEY-----
MIIEvAIBADANBgkqhkiG9w0BAQEFAASCBKYwggSiAgEAAoIBAQDmLE+qPcLijKQl
EgBBYPnaZEV8kk0YfnalsqRygwHA/6ZWetSIHeJjWQjzbSVflqW+T3tdbkJkJcVn
8DKpg95HDbJpYHwCWVGSYH1x0vLmVCBoKFWgKQfPXb2toXvaLZvoBMX0pY3qsL2d
Kegg6bYUvsVYJPM97I56+VnWzR8vuvvRyWdATgVC02IXS2zgdcyf1QiS46L9DdUP
ivSy7ExRxaZtS6OiUL9gQ3gZMecRgYftvQkSKOJmAcZozEe56osba75gEeLuM1yy
U/SAOe4Z+oZvhlwfceu2RjAd8P3812Q4Fi37gvbLTOMLuXPPXtGQIwTyeT2gzBe2
bKq1jPjTAgMBAAECggEAXju/9dPJSNUJkr+26Q1Z3WZ16FVBSQj1bFsh5lvqSOih
YgBxqLSLbd8Swg5qYaMcgjiqKgstFTE9i5sUMsBxLSfXZjsBKjkFI2drlEEn/tJs
Xhg9Tg/eSbB4KnDVcvJ/RNNHdCyfWHk5d/GjPm5QhFfCFZ8zEqQZhRE3DbD21qPx
CM10MW6K6f0r9e8o7dM/8MoHYS1giD1QAy1Uzk1cFNqm64fGiGgZS9df77apzYk1
6pz7dL7G8YkyyQP71U0NU20FIH9p4AAtq9+YhL1RRnI7LCIFExTvqMvxCmp55W6U
q0Lv+AKpG5kaZKnhyY1x8/y4+5ORBFmQOC8u+pNeQQKBgQD4sihwsMHp/W9Wywu8
DmBD1z3vHTd77ESTTZmWAkHTjTK67zMNkKKgG5ORw80Eb+ziGw0zLrIzqXrZ+t3C
yV3vTvAv8TpbyuAzg5E1mr7N7N5BoDncaySKTlu+9pb/geVcJ7Dgfjr8Xx4sbwur
34iFvKhQn7mKOtFLh3xpq1yJ4QKBgQDs7uM8giT47VzjgpDSI39b5z54OI3JJUg6
kYIoZQWMZF9chBJdX53AlP0vg4Ujn7rpxxhjKH3yRo7qgAO20X8BoxIN1OsGy5o+
9viUI3PfzdFOpK0rF4pLQ2o61/1zc10EGx+wGSPm3FPnr0WibSiYC/ANSsbU+YJ/
JmyEUkehMwKBgD3z88sO3V+y+LeAYW35jtWHqbAVcSr30qV3YV1TOp6+v/J9gBRT
AZAtDnfzdPE0CKjS4FjtuwrPn4EJ05CgpTvmSENJLqaLRYrPrarWf83AwiVLCt7z
+ehSWNybhUY1T8PDR3bFO2B/OX8MMrYgO1xFgRCEma3rtRPBrqaMP8NBAoGAOpxN
siYvpzRwigc+c/08ACwQFAVJbby/RV3BaOS0nqolvOke8eI5T+6LQzmxscJNpo8N
xVm67jfuGqxYPt1MIceJK8rD6QZsva6yh1LN00tCv8WU4NgN6dhfauc2V1yvDdj3
2cQeUXmnd+SvwSJnrFUnqOIalYyzb2i9523DMmkCgYBS/V1k1XnPOtOY28PhW84I
T/uxk+7Ljlp2aweVbhGNV7YvrKdCWhA6G6doD8trjl22e+vciHhi4qpXIGqbUoxQ
66qUylIwxd/JDJTH8USkEO5neG79uZUIDF75XKk/0XOOI/2Os59w0emGYaXllVjK
E0hUeN6K65mQatXeFnJMKQ==
-----END PRIVATE KEY-----
```

These 2 files you now copy to the folder /etc/homeserver/ on the Reverse Proxy using WinSCP.



Because of the long validity (15 months) actualization of the certificate copies on Reverse Proxy is seldom needed. Therefore you should make a printed copy of this document and carefully safe it.

Common Nginx Reverse Proxy Options to enable Websockets

copied from: Setting up an Nginx Reverse Proxy | Linuxize

Serving content over HTTPS has become a standard nowadays. In this section, we will give you an example of HTTPS Nginx reverse proxy configuration including the recommended Nginx proxy parameters and headers.

```
  location / {
    proxy_pass http://127.0.0.1:3000;
    proxy_http_version  1.1;
    proxy_cache_bypass  $http_upgrade;

    proxy_set_header Upgrade           $http_upgrade;
    proxy_set_header Connection        "upgrade";
    proxy_set_header Host              $host;
    proxy_set_header X-Real-IP         $remote_addr;
    proxy_set_header X-Forwarded-For   $proxy_add_x_forwarded_for;
    proxy_set_header X-Forwarded-Proto $scheme;
    proxy_set_header X-Forwarded-Host  $host;
    proxy_set_header X-Forwarded-Port  $server_port;
  }
```

- `proxy_http_version 1.1` - Defines the HTTP protocol version for proxying, by default it is set to 1.0. For Websockets and keepalive connections you need to use the version 1.1.
- `proxy_cache_bypass $http_upgrade` - Sets conditions under which the response will not be taken from a cache.
- `Upgrade $http_upgrade` and `Connection "upgrade"` - These header fields are required if your application is using Websockets.
- `Host $host` - The `$host` variable in the following order of precedence contains: hostname from the request line, or hostname from the `Host` request header field, or the server name matching a request.
- `X-Real-IP $remote_addr` - Forwards the real visitor remote IP address to the proxied server.
- `X-Forwarded-For $proxy_add_x_forwarded_for` - A list containing the IP addresses of every server the client has been proxied through.
- `X-Forwarded-Proto $scheme` - When used inside an HTTPS server block, each HTTP response from the proxied server is rewritten to HTTPS.
- `X-Forwarded-Host $host` - Defines the original host requested by the client.
- `X-Forwarded-Port $server_port` - Defines the original port requested by the client.

The Scripts on my Reverse Proxy:

/etc/nginx/sites-available/default

```
            (not linked to /sites-enabled/ !)

# Default server configuration

server {
        listen 80 default_server;
        listen [::]:80 default_server;

        root /var/www/html;
        index index.html index.htm index.nginx-debian.html;

        server_name _;

        location / {
                # First attempt to serve request as file, then
                # as directory, then fall back to displaying a 404.
                try_files $uri $uri/ =404;
        }
}
```

Content of file: index.nginx-debian.html

```
<!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx!</title>
<style>
    body {
        width: 35em;
        margin: 0 auto;
        font-family: Tahoma, Verdana, Arial, sans-serif;
    }
</style>
</head>
<body>
<h1>Welcome to nginx!</h1>
<p>If you see this page, the nginx web server is successfully installed and
working. Further configuration is required.</p>

<p>For online documentation and support please refer to
<a href="http://nginx.org/">nginx.org</a>.<br/>
Commercial support is available at
<a href="http://nginx.com/">nginx.com</a>.</p>

<p><em>Thank you for using nginx.</em></p>
</body>
</html>
```

/etc/nginx/sites-available/knausenberger.info.conf

```
# Configuration for knausenberger.info and www.knausenberger.info

server {
  listen 80;
  listen [::]:80;
  server_name knausenberger.info www.knausenberger.info;

  include snippets/letsencrypt.conf;
  return 301 https://$host$request_uri;
}

server {
  listen 443 ssl http2;
  listen [::]:443 ssl http2;
  server_name www.knausenberger.info;

  ssl_certificate /etc/letsencrypt/live/knausenberger.info/fullchain.pem;
  ssl_certificate_key /etc/letsencrypt/live/knausenberger.info/privkey.pem;
  ssl_trusted_certificate /etc/letsencrypt/live/knausenberger.info/chain.pem;
  include snippets/ssl.conf;
  include snippets/letsencrypt.conf;

  return 301 https://knausenberger.info$request_uri;
}

server {
  listen 443 ssl http2;
  listen [::]:443 ssl http2;
  server_name knausenberger.info;

  ssl_certificate /etc/letsencrypt/live/knausenberger.info/fullchain.pem;
  ssl_certificate_key /etc/letsencrypt/live/knausenberger.info/privkey.pem;
  ssl_trusted_certificate /etc/letsencrypt/live/knausenberger.info/chain.pem;
  include snippets/ssl.conf;
  include snippets/letsencrypt.conf;

  # . . . other code
  location / {
    include proxy_params;
    proxy_pass https://10.163.54.30;
    proxy_http_version  1.1;
    proxy_cache_bypass $http_upgrade;
#    proxy_headers_hash_max_size 512;


    proxy_set_header Upgrade           $http_upgrade;
    proxy_set_header Connection        „upgrade“;
    proxy_set_header Host              $host;
    proxy_set_header X-Real-IP         $remote_addr;
    proxy_set_header X-Forwarded-For   $proxy_add_x_forwarded_for;
    proxy_set_header X-Forwarded-Proto $scheme;
    proxy_set_header X-Forwarded-Host  $host;
    proxy_set_header X-Forwarded-Port  $server_port;
  }
}
```

/etc/nginx/sites-available/knausir.com.conf

```
# Configuration for knausir.com and www.knausir.com

server {
  listen 80;
  listen [::]:80;
  server_name knausir.com www.knausir.com;

  include snippets/letsencrypt.conf;
  return 301 https://$host$request_uri;
}

server {
  listen 443 ssl http2;
  listen [::]:443 ssl http2;
  server_name www.knausir.com;

  ssl_certificate /etc/letsencrypt/live/knausir.com/fullchain.pem;
  ssl_certificate_key /etc/letsencrypt/live/knausir.com/privkey.pem;
  ssl_trusted_certificate /etc/letsencrypt/live/knausir.com/chain.pem;
  include snippets/ssl.conf;
  include snippets/letsencrypt.conf;

  return 301 https://knausir.com$request_uri;
}

server {
  listen 443 ssl http2;
  listen [::]:443 ssl http2;
  server_name knausir.com;

  ssl_certificate /etc/letsencrypt/live/knausir.com/fullchain.pem;
  ssl_certificate_key /etc/letsencrypt/live/knausir.com/privkey.pem;
  ssl_trusted_certificate /etc/letsencrypt/live/knausir.com/chain.pem;
  include snippets/ssl.conf;
  include snippets/letsencrypt.conf;

  # . . . other code
  location / {
    proxy_buffering off;
    include proxy_params;
    proxy_pass https://10.163.54.2;
    proxy_http_version  1.1;
    proxy_cache_bypass $http_upgrade;
#     proxy_headers_hash_max_size 512;

    proxy_set_header Upgrade           $http_upgrade;
    proxy_set_header Connection        „upgrade“;
    proxy_set_header Host              $host;
    proxy_set_header X-Real-IP         $remote_addr;
    proxy_set_header X-Forwarded-For   $proxy_add_x_forwarded_for;
    proxy_set_header X-Forwarded-Proto $scheme;
    proxy_set_header X-Forwarded-Host  $host;
    proxy_set_header X-Forwarded-Port  $server_port;
  }
}
```

To make sure the scripts are started in nginx we create links for these scripts in /etc/nginx/sites-enabled/

sudo ln -s /etc/nginx/sites-available/knausenberger.info.conf /etc/nginx/sites-enabled/
sudo ln -s /etc/nginx/sites-available/knausir.com.conf /etc/nginx/sites-enabled/
sudo systemctl restart nginx.service

Together with these scripts, the converted Windows Server certificate and the Let's Encrypt-certificates the system shows error free functionality.

The web sites are opening and are usable.

Zwischenzeitlich hat sich die Situation verändert:

Hinter dem Reverse Proxy betrieben erneuert der IceWarp Mailserver sein Zertifikat nicht mehr.

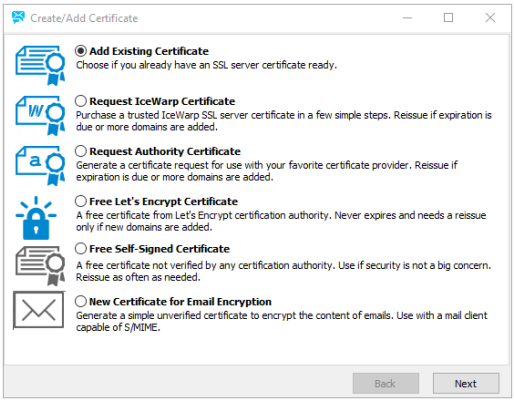
Deshalb greifen wir stattdessen auf die Zertifikatsdateien ***fullchain.pem*** sowie ***privkey.pem*** im Reverse Proxy zu und portieren sie zum IceWarp Server.

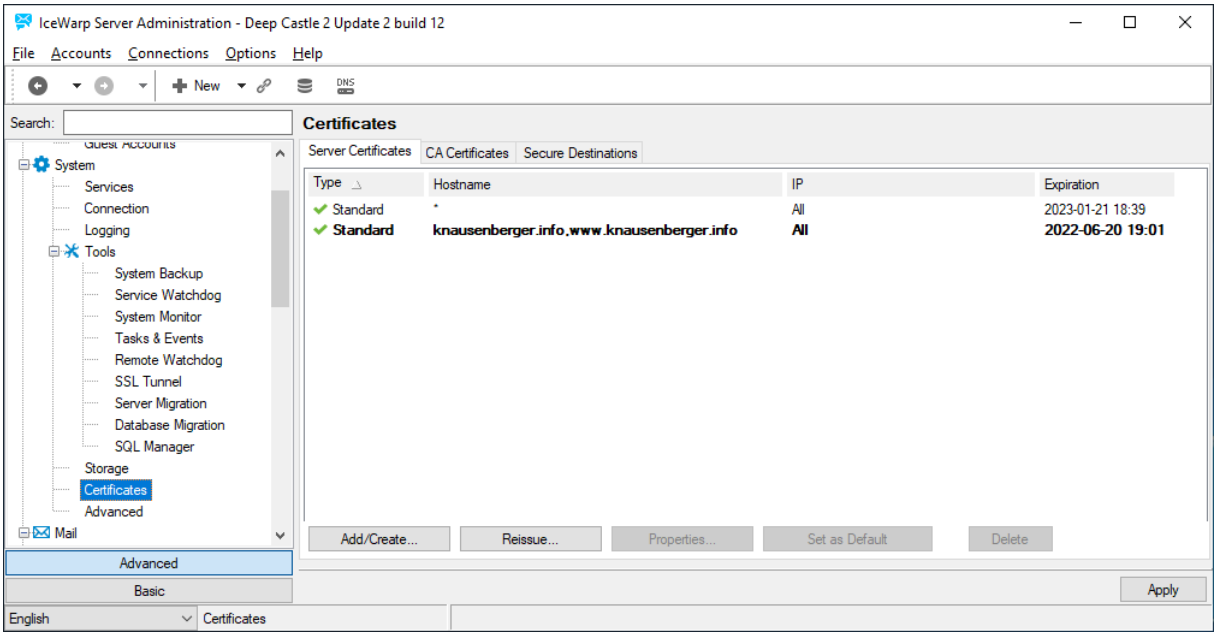
```
root@raspi-proxy:/etc/letsencrypt/live/knausenberger.info# sudo su
root@raspi-proxy:/etc/letsencrypt/live/knausenberger.info# cp fullchain.pem /etc/icewarp-server/fullchain.pem
```

Wenn das hakt, stimmen die Zugriffsrechte nicht. Dann muss man diese entsprechend ändern:

```
root@raspi-proxy:/etc/icewarp-server# chmod -v 777 privkey.pem
der Modus von 'privkey.pem' wurde von 0600 (rw-------) in 0777 (rwxrwxrwx) geändert
root@raspi-proxy:/etc/icewarp-server#
```

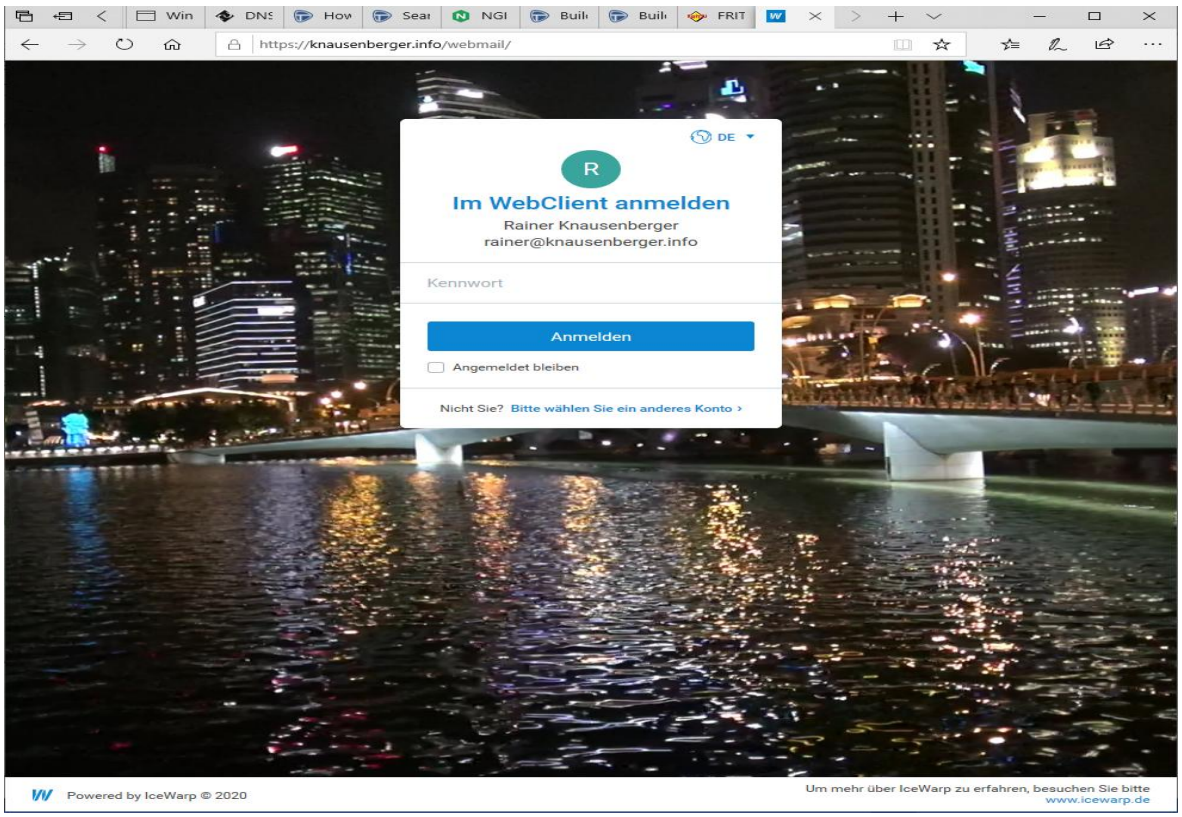
Nun lassen sich die beiden Dateien ***fullchain.pem*** und ***privkey.pem*** vom Reverse Proxy mittels WinSCP auf den IceWarp-Rechner kopieren, der mit der Administrationskonsole des IceWarp Servers umgehen kann (Remote Desktop).
Den Inhalt von ***privkey.pem*** muss man noch an den Inhalt von ***fullchain.pem*** mittels Notepad anhängen, dann lässt es sich den IceWarp-Zertifikaten hinzufügen.



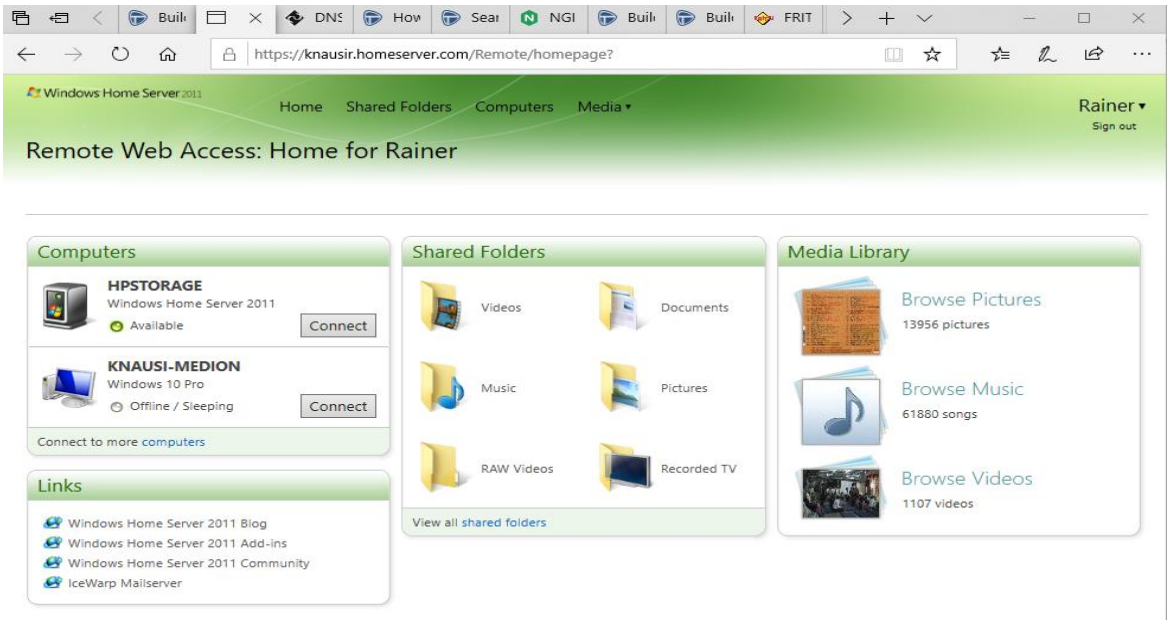


IceWarp Server requires directing port 10000 to the Icewarp-Server. This ist o be done in the router.

knausenberger.info

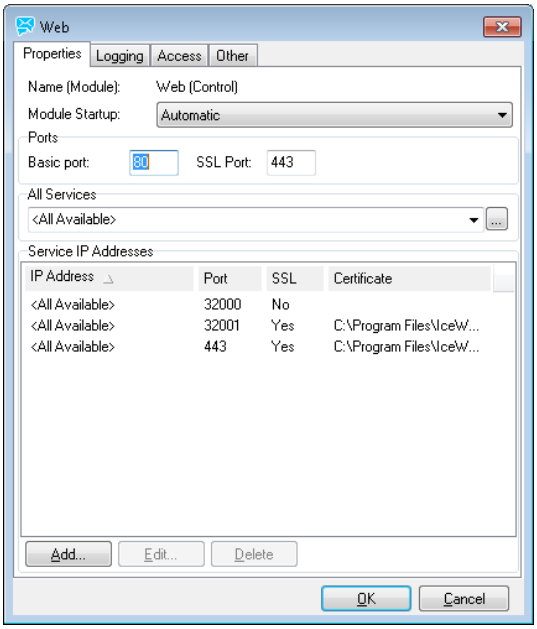


knausir.com

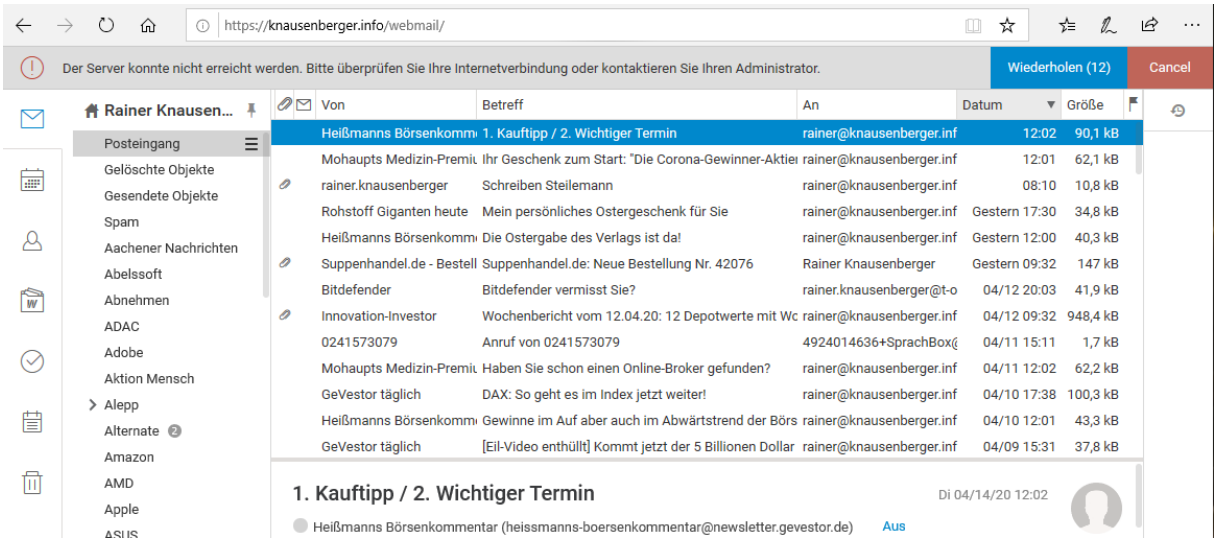


IceWarp Server knausenberger.info

In der Server-Administration ist die Web-Konfiguration wieder auf http (Port 80) und https (Port 443) gesetzt, dank dem Einsatz des Reverse Proxy. Ich hatte bisher die Ports 79 und 442 benutzt mit den entsprechenden Freigaben im Router. Auch in Services | Smart Discover sind die Ports wieder „zurückgebogen“. Da aber alle http-Anfragen im Reverse-Proxy per redirect auf https geändert werden, müssen auch im **Smart Discover** alle URL´s auf https geändert werden. Das entspricht auch den Vorgaben in den aktuellen IceWarp Manuals.



Nach einem erfolgreichen Start des WebClients lässt dieser sich wie gewohnt bedienen, aber nach einigen wenigen Sekunden blendet sich folgende Fehlermeldung in die Menueleiste ein.



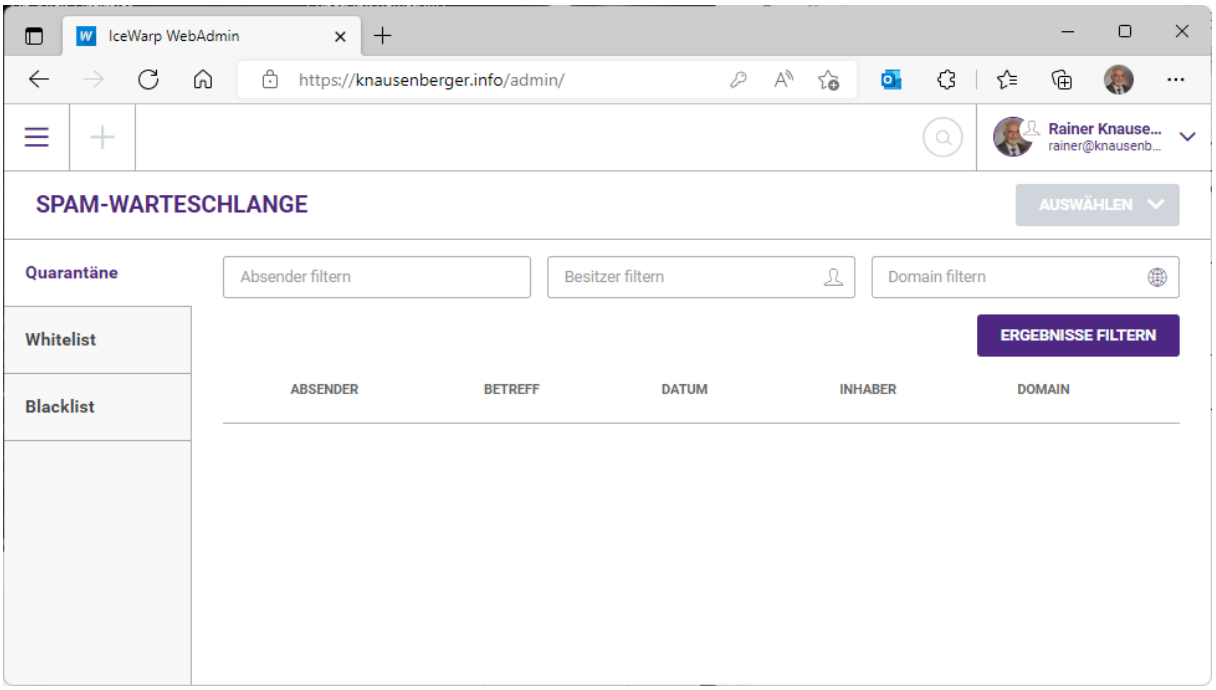
bzw.

Man konnte diese Meldungen bestätigen (Wiederholen/Retry bzw. Cancel und auch OK), sie kamen aber ständig wieder, was sehr lästig war.

Die Ursache liegt darin, dass IceWarp Server zur Kommunikation mit den Clients **Websockets** benutzt, was der Reverse Proxy in der bisherigen Auslegung jetzt berücksichtigt. Leider wird in den Publikationen und Manuals zum IceWarp Server **nirgends auf diesen Sachverhalt hingewiesen**.

Dieser Fehler tritt nach Konfiguration des Reverse Proxy entsprechend der hier vorliegenden Anleitung nicht mehr auf!

Es bleibt aber ein weiterer Fehler, den ich bisher nicht beseitigen konnte. Dieser betrifft den WebAmin.



Wertung, offene Punkte und Fragen

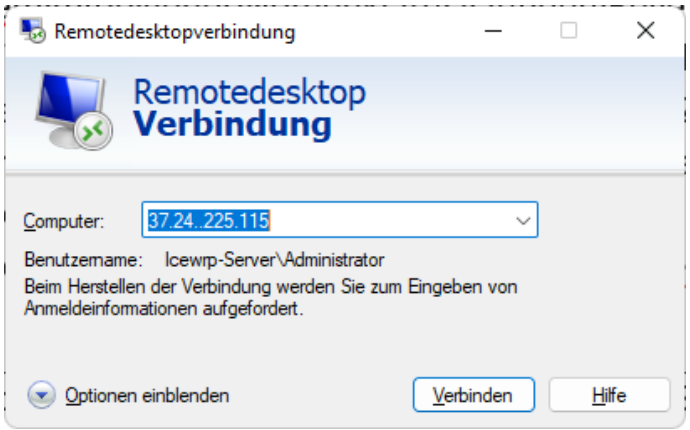
Frage 1: Liegt es an den unterschiedlichen Zertifikaten im Reverse-Proxy und Icewarp-Server? Wie kann ich das abschalten? Könnte man das Icewarp-Server-Zertifikat auch im Reverse-Proxy benutzen oder umgekehrt?
Eine andere Möglichkeit könnte es auch sein, für den IceWarp-Server im Reverse-Proxy nicht die „Proxy_Pass“ Direktive sondern eine an die php-Syntax angepasste Konfiguration zu benutzen.

Frage 2: Wie müsste diese dann aussehen? Man findet im Internet Konfigurationen für WordPress, welches ebenfalls eine php-Anwendung ist.

Wertung 1: Die Verwendung eines Reverse-Proxy (basierend auf einem Raspberry 3b mit Raspian Buster und NGINX) ist eine sehr benutzerfreundliche, mit geringem finanziellen und zeitlichen Aufwand zu realisierende Lösung, da man sich als Anwender nicht um irgendwelche Ports kümmern muss. Ein weiterer Vorteil ist, dass auch der Windows-Server hinter dem Reverse-Proxy betrieben werden kann, dessen IIS-Zugangsports 80 und 443 nicht geändert werden dürfen, da sonst Funktionen „sterben“. Man muss allerdings auf dem Reverse-Proxy für den Windows-Server dasselbe Zertifikat verwenden wie das auf dem Windows-Server installierte, da der Windows-Server – unabhängig vom Einsatz eines Reverse Proxy - jeden Webaufruf über HTTP auf HTTPS umstellt. Da das Windows-Server-Zertifikat im .pfx-Format vorliegt, muss es ins pem-Format mittels OpenSSL konvertiert und dann in die Teile fullchain.pem und privkey.pem zerlegt werden, die dann im Reverse-Proxy an die richtigen Stellen kopiert werden müssen. Diese Prozedur ist recht aufwendig, muss aber für den Windows-Server nur gelegentlich durchgeführt werden, da die Gültigkeitsdauer des Zertifikats ca. 2 Jahre beträgt.

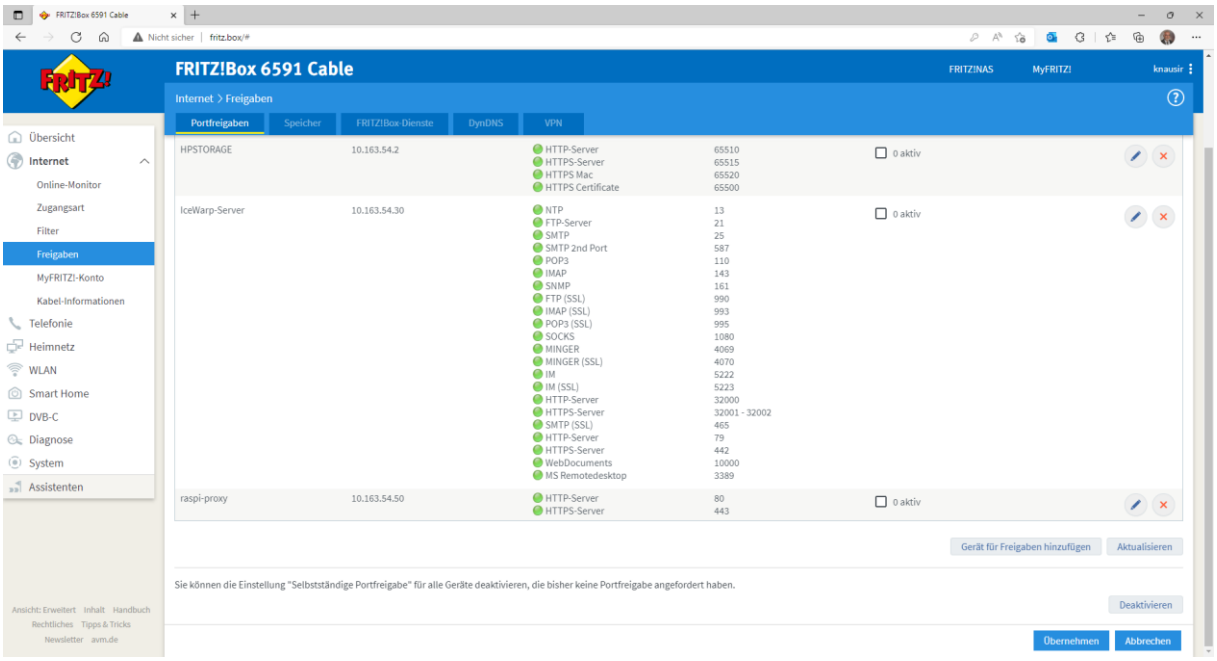
Frage 3: Gibt es eine analoge Vorgehensweise, die im IceWarp-Server verwendeten Let´sEncrypt-Zertifikate (**********.pem mit eigener Nomenklatur) auf den Reverse-Proxy zu übertragen oder umgekehrt, denn diese Zertifikate müssten in beiden Maschinen eigentlich identisch sein? Könnte man dann noch einen Update-Mechanismus einrichten, um die beiden Zertifikate nach dem alle 90 Tage fälligen Renew synchron zu halten? Das auto-renew ist auf dem Reverse-Proxy für die Zertifikate zu den Domänen **knausenberger.info** und **knausir.com** bereits eingerichtet (s.o.) und läuft automatisch. Es wäre sehr komfortabel, wenn das auch für die IceWarp Zertifikate genutzt werden könnte.

Hinweis: Mittlerweile habe ich einen RDP-Zugang zu meinem IceWarp-Server angelegt, was den Umweg über den Homeserver erspart. Die zwingend zugehörige Portfreigabe (ohne IPv6) im Router sehen Sie im unteren Bild.



Der Benutzer muss ggf. angepasst werden auf Icewarp-Server\Administrator, dessen Passwort Ihnen aus früheren Aktionen bekannt ist. **Achtung**: Die IP-Adresse ist jetzt: **37.24.225.115**

Hier meine Router-Freigaben:



Alle Freigaben (bis auf Remotedesktop) gelten auch für IPv6.