Apache2 Reverse Proxy mit dem Raspberry PI für einen Sicheren Zugriff aus dem Internet

Anleitung für was?

Eine Sichere Möglichkeit von Unterwegs per https Verschlüsselt auf Webportale innerhalb des Heimnetzwerks zuzugreifen ohne das diese Geräte https Unterstützen müssen und im Rounter / Firewall genau nur eine Port Weiterleitung (Freigabe) auf den Linux Rechner gemacht werden muss!

Warum das?

- Vermeidung von Cloud Diensten über welche Netzwerkverbindungen ins eigene Netzwerk geöffnet werden
- Damit Apps wie Homedroid einfach über eine URL von Intern und Extern Syncronisieren können.
- Sicher wäre eine VPN Verbindung nach Hause Sicherer aber auch im Täglichen Leben Unpraktischer zumal vor jedem Zugriff immer der VPN Tunnel aufgebaut werden muss.
- Dieser Proxy ist auf einem „Standard“ Betriebssystem Installiert und kann einfach auf dem Aktuellen Stand gehalten werden, alle anderen Geräte wie eine CCU oder Webcam haben meist sehr alte Module für den Webserver oder das Betriebsystem allgemein und damit auch viele Bekannte Sicherheitslücken über die ihr Netzwerk Angreifbar ist!

Voraussetzungen

- Linux Rechner mit Raspbian / Debian und 24/7 Online ist.
- Mit einer Statischen IP Adresse oder einer Reservierung in ihrem DHCP Server (meist der Router / Firewall) immer unter der gleichen IP Adresse erreichbar ist.
- Einen Router / Firewall mit einer fixen IP Adresse oder einem Dynamischen DNS Dienst (DynDNS der Bekannteste ist inzwischen kostenlos nicht mehr nutzbar aber es gibt je nach Router / Firewall auch noch andere immer noch Kostenlose Anbieter) und einer Port Weiterleitung vom Port 443 (https) auf die Interne Adresse des Linux Rechners.
- Bedienung des vi (vim) Editor –> eine Bedienungsanleitung finden Sie hier http://linuxwiki.de/Vim die Wichtigsten Befehle im Editor sind:
  - [i] Um in den Einfügemodus zu wechseln damit eine Datei verändert werden kann.
  - [ESC] um aus dem Einfügemodus in den Befehlsmodus umzuschalten.
  - [:wq] um eine Datei zu schreiben und den Editor zu verlassen.
  - [:q!] um eine Datei ohne Speicherung der Änderung zu schließen.

Sicherheit

Diese ist abhängig vom Kennwort das Sie wie hier in dieser Anleitung Beschrieben für den Zugriff

vergeben!

Zur Erhöhung der Sicherheit sei noch das fail2ban Paket unter Linux empfohlen welches auch in dieser Anleitung erklärt wird und eine „brute-force“ attacke auf die Dienste verhindert!

Vorbereitung Raspbian / Debian 9

Installation

System Aktualisieren

Zuerst bitte das System auf den neuesten stand bringen mit:

```
sudo aptitude update && aptitude dist-upgrade -y
```

Pakete Installieren

Dann müssen folgende Pakete Installiert werden:

```
sudo aptitude install apache2 apache2-utils libapache2-mod-rpaf vim apache2-
utils -y
```

Apache Module aktivieren

Dann müssen folgende Module im Apache aktiviert werden:

```
sudo a2enmod ssl headers rewrite proxy proxy_http
```

Einrichtung

Letsencrypt --> Kostenloses https Zertifikat

Apache Paket Installieren

```
sudo aptitude install python-letsencrypt-apache
```

Apache default SSL config

Eine Gültige ssl config mit dem DNS namen (ServerName in apache config) unter welchem der Reverse Proxy aus dem Internet erreichbar ist anlegen (mit dem Standard Snakeoil SSL Zertifikat

reicht!)

```
sudo vi /etc/apache2/sites-available/<dns name>.<endung>.conf
```

mit folgendem Inhalt anlegen:

```
<IfModule mod_ssl.c>
        <VirtualHost _default_:443>
                ServerAdmin webmaster@localhost
                ServerName <dns name>.<endung>

                DocumentRoot /var/www/html

                ErrorLog ${APACHE_LOG_DIR}/error.log
                CustomLog ${APACHE_LOG_DIR}/access.log combined

                SSLEngine on

                SSLCertificateFile      /etc/ssl/certs/ssl-cert-snakeoil.pem
                SSLCertificateKeyFile /etc/ssl/private/ssl-cert-snakeoil.key

        </VirtualHost>
</IfModule>
```

Diese Seite aktivieren mit:

```
a2enside <dns name>.<endung>.conf
```

Apache neu starten mit:

```
systemctl restart apache2.service
```

Port Forwarding im Router

auf die FIXE IP des Raspberry TCP Port 443 Einrichten (je nach Modell Unterschiedlich –> Google it!)

Zertifikat installieren

Folgenden Befehl Interaktiv ausführen

```
letsencrypt
```

und die Domäne auswählen für die das Zertifikat erstellt werden soll.

Nach der eingabe einer Mail Adresse unter der sie Benachrichtigt werden können falls es Fragen wengen dem Zertifikat gibt wird dies erstellt und die Abfrage ob sie auch http auf https umleiten wollen können sie beantworten wie sie möchten.

Cronjob zum Automatischen Aktualisieren

Jeden Sonntat um 1 Uhr Morgens wird Versucht das Zertifikat zu erneuern was aber erst nach 4 Wochen gemacht wird.

```
  * 1 * * 1 /usr/bin/letsencrypt -q renew
```

Nach dem neustarten vom Apache sollte wenn sie den DNS Namen aufrufen das gültige Zertifikat vom Browser angezeigt werden.

HTTPS Basic Authentication einrichten

Ordner für Zertifikate erstellen

```
sudo mkdir /etc/apache2/ssl
```

Benutzer erstellen

Jetzt erstellen wir eine Kennwort Datei

```
sudo htpasswd -c /etc/apache2/ssl/httpsusers admin
```

Wobei der Benutzername angepasst werden kann und dann zweimal das Kennwort für den Benutzer bestätigt werden muss. Wenn ein weiterer Benutzer erstellt werden soll muss der Befehl so aussehen

```
sudo htpasswd /etc/apache2/ssl/httpsusers Benutzername
```

Authentifizierung im Apache einbinden

Nun kann die /etc/apache2/sites-available/<dns name>.<endung>.conf Erweitert werden um:

```
 <Location />
      AuthType basic
      AuthName "home"
      AuthUserFile  /etc/apache2/ssl/httpsusers
      Require valid-user
  </Location>
  <Directory /var/www/html/>
      Options Indexes FollowSymLinks MultiViews
      AllowOverride All
      Require all granted
  </Directory>
```

Als nächstes empfehle ich fail2ban zu Installieren und aktivieren (siehe unten)

Nun können die Blöcke von unten mit den Konfigurationen für die Dienste an die Datei /etc/apache2/sites-available/<dns name>.<endung>.conf angefügt werden

ALT Vorbereitung Raspbian / Debian 7 und 8

System Aktualisieren und Pakete Installieren

System Aktualisieren

Zuerst bitte das System auf den neuesten stand bringen mit:

```
sudo aptitude update && aptitude dist-upgrade -y
```

Pakete Installieren

Dann müssen folgende Pakete Installiert werden:

```
sudo aptitude install apache2 apache2-utils libapache2-mod-proxy-html vim -y
```

Apache Module aktivieren

Dann müssen folgende Module im Apache aktiviert werden:

```
sudo a2enmod ssl rewrite proxy proxy_http
```

SSL (https) Zufriff für den Apache Einrichten

Damit der Browser nicht beim Besuch der Seite darauf hinweist eine nicht Sichere Seite zu Besuchen, sie aber bei jedem Browser auch das selbst erstellte Zertifikat dauerhaft Importieren können, wie das mit z.B. dem Firefox mit nur drei Klicks gemacht werden kann sehen sie am Ende dieses Abschnittes. Da ihr aber das Zertifikat selbst auf eurem System mit dem Nachfolgendem Befehl generiert und die Seite auch nur innerhalb eurer Familie benutzt wird sehe ich das als keinen gravierenden Nachteil.

Wie diese beiden Varianten genutzt werden können erklären die beiden nächsten Punkte.

entweder Offizielles Zertifikat

Es gibt neben den vielen Komerziellen Anbietern auch welche die ein Kostenloses SSL Zertifikat anbieten, z.B.: Start SSL, aber im Fall von StartSSL gibt es nur ein Zertifikat welches nur ein Jahr gültig

ist und dann erneuert werden muss.

Wie dieses Zertifikat erstellt werden kann ist von Anbieter zu Anbieter etwas unterschiedlich aber nach der bekanntgabe der Daten (common Name, Namen usw.) erhaltet ihr die beiden Zertifikat Dateien den Öffentlichen und Privaten Schlüssel. Gerade bei den Kostenlosen Anbietern wird das Zertifikat aber meistens nicht auf eurem System erstellt und somit könnt euer Privater Schlüssel nicht mehr Garantiert Privat sein, was aber in diesem Fall nicht so schlimm ist da dieser nicht zum Zugriff auf die durch die Authetifizierung geschützten Inhalte genutzt werden kann sondern nur eine Aktive SSL Verbindung könnte mitgelesen werden.

oder Selbst erstelltes Zertifikat

Dieses wird mit einer Gültigkeit von 3650 Tagen (10 Jahre) erstellt und muss bei jedem Browser anders Importiert werden damit die Meldung wegen einer angeblich unsicheren Verbindung nicht jedesmal angezeigt wird.

Zertifikat selbst erstellen

```
sudo openssl req -x509 -nodes -days 3650 -newkey rsa:2048 \
-out /etc/apache2/ssl/server.crt -keyout /etc/apache2/ssl/server.key
```

Beim erstellen des Zertifikats bitte mindestens folgendes anpassen, bei den anderen Abfragen können Sie mit [Enter] weitergehen:

```
Country Name (2 letter code) [AU]:DE
Common Name (e.g. server FQDN or YOUR name) []:haus.<your-dns-name>.com
```

Der [Common Name] sollte dem DNS Namen (z.B. dem DynDNS Domain Namen <user>.<dnsprovider-domain>.<endung>) entsprechen unter welchem der Reverse Proxy später aus dem Internet erreicht werden kann! Nur dann ist gewährleistet das der Browser den Zugriff nach der Bestätigung das ihr diesem selbst erstellten Zertifikat vertraut auch in Zukunft erlaubt!

Die ssl Seite in Apache Aktivieren

```
sudo a2ensite default-ssl
```

Dann in der Datei **/etc/apache2/sites/enabled/default-ssl.conf** folgende Anpassungen machen:

```
sudo vi /etc/apache2/sites-enabled/default-ssl.conf
```

und dort im Tag **<VirtualHost _default_:443>** folgende Zeilen hinzufügen:

```
SSLCertificateFile    /etc/apache2/ssl/server.crt
SSLCertificateKeyFile /etc/apache2/ssl/server.key
```

und die zwei des mit installierten Standard Zertifikats deaktivieren / auskomentieren indem sie ein **#** voranstellen!

```
#       SSLCertificateFile    /etc/ssl/certs/ssl-cert-snakeoil.pem
#       SSLCertificateKeyFile /etc/ssl/private/ssl-cert-snakeoil.key
```

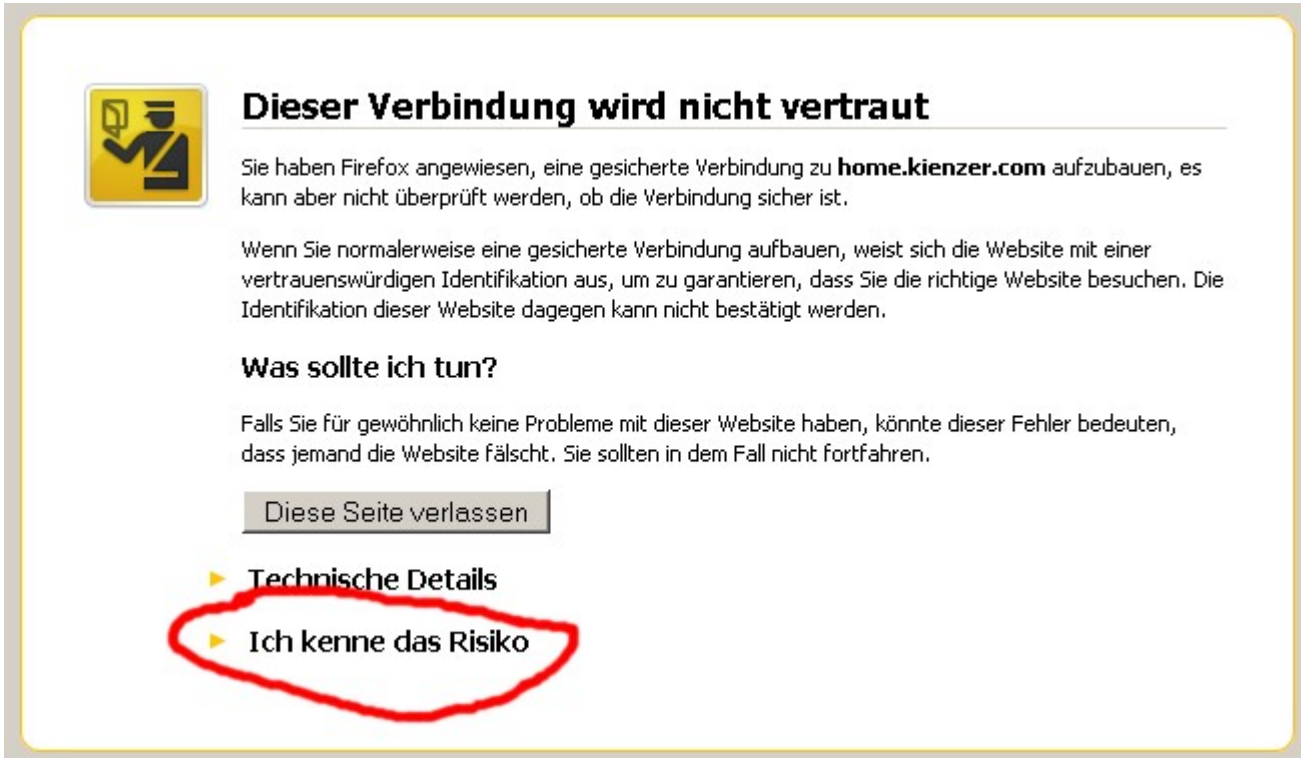
Dann den Webserver neu Starten

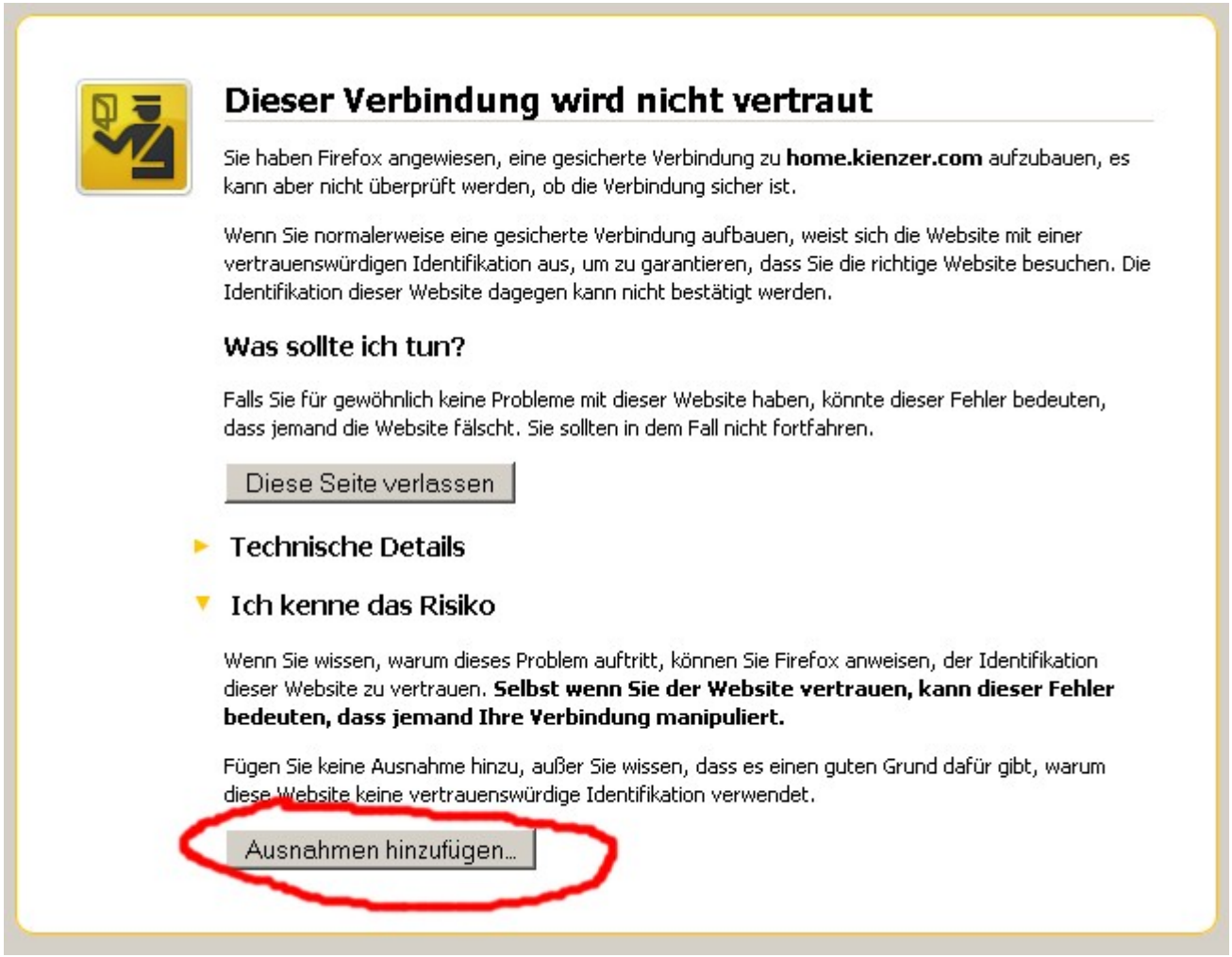
```
sudo service apache2 restart
```

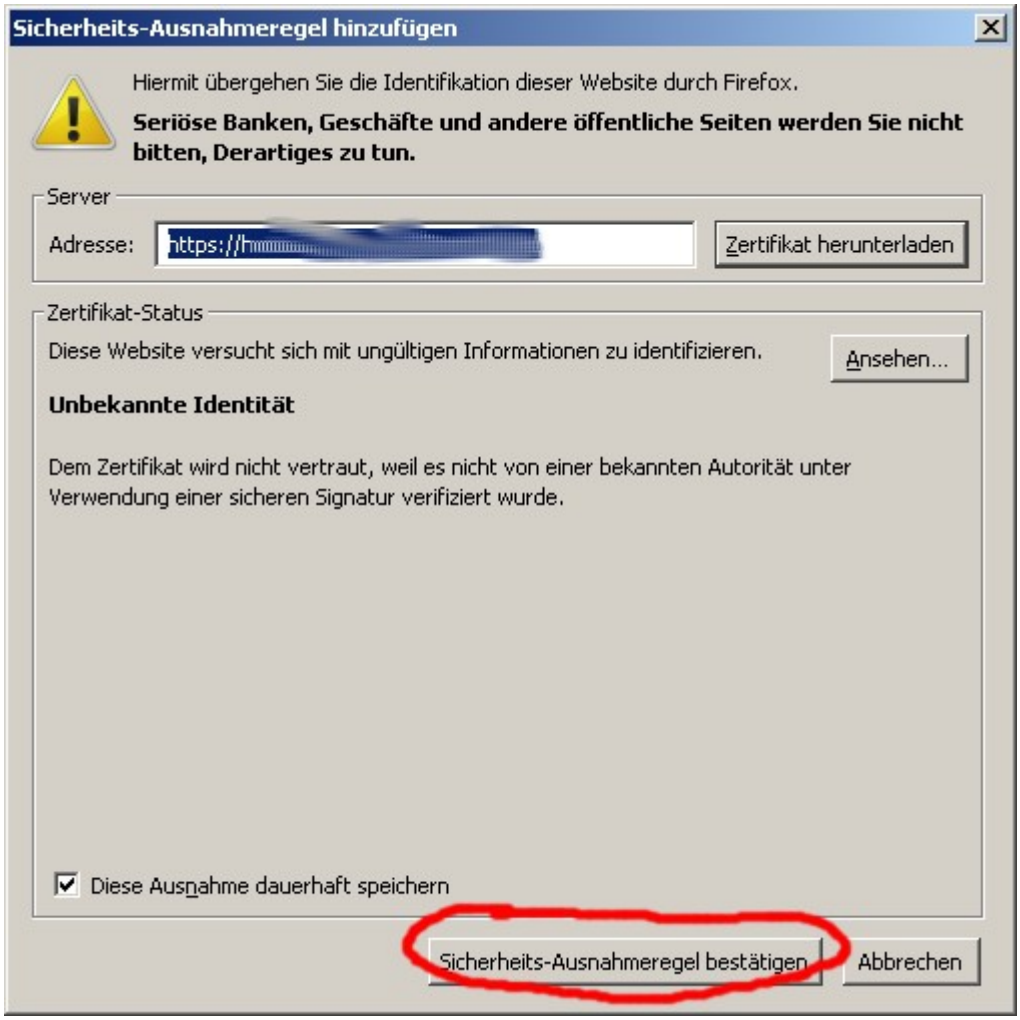
Nun könnt ihr Testen ob der Raspi über https erreichbar ist

```
https://<ip oder rechnername>
```

und auch das Zertifikat ansehen / importieren dazu müsst ihr folgendes im Firefox Browser auf jedem Rechner einmalig machen:




Dieser Verbindung wird nicht vertraut
Sie haben Firefox angewiesen, eine gesicherte Verbindung zu home.kienzer.com aufzubauen, es kann aber nicht überprüft werden, ob die Verbindung sicher ist.
Wenn Sie normalerweise eine gesicherte Verbindung aufbauen, weist sich die Website mit einer vertrauenswürdigen Identifikation aus, um zu garantieren, dass Sie die richtige Website besuchen. Die Identifikation dieser Website dagegen kann nicht bestätigt werden.
Was sollte ich tun?
Falls Sie für gewöhnlich keine Probleme mit dieser Website haben, könnte dieser Fehler bedeuten, dass jemand die Website fälscht. Sie sollten in dem Fall nicht fortfahren.
Diese Seite verlassen
Technische Details
Ich kenne das Risiko
Wenn Sie wissen, warum dieses Problem auftritt, können Sie Firefox anweisen, der Identifikation dieser Website zu vertrauen. Selbst wenn Sie der Website vertrauen, kann dieser Fehler bedeuten, dass jemand Ihre Verbindung manipuliert.
Fügen Sie keine Ausnahme hinzu, außer Sie wissen, dass es einen guten Grund dafür gibt, warum diese Website keine vertrauenswürdige Identifikation verwendet.
Ausnahmen hinzufügen...



HTTPS Basic Authentication einrichten

Ordner für Zertifikate erstellen

```
sudo mkdir /etc/apache2/ssl
```

Benutzer erstellen

Jetzt erstellen wir eine Kennwort Datei

```
sudo htpasswd -c /etc/apache2/ssl/httpsusers admin
```

Wobei der Benutzername angepasst werden kann und dann zweimal das Kennwort für den Benutzer bestätigt werden muss. Wenn ein weiterer Benutzer erstellt werden soll muss der Befehl so aussehen

```
sudo htpasswd /etc/apache2/ssl/httpsusers Benutzername
```

Authentifizierung im Apache einbinden

Nun muss diese Datei noch in den Apache für den Aufruf des Webroot eingebunden werden dazu editieren wir wieder die Datei

```
sudo vi /etc/apache2/sites-enabled/default-ssl.conf
```

und fügen im **<VirtualHost _default_:443>** Tag den Block **<Location />** wie folgt ein:

```
<Location />
    Deny from all
    AuthType basic
    AuthName "home"
    AuthUserFile  /etc/apache2/ssl/httpsusers
    Satisfy Any
    Require valid-user
</Location>
```

bis Apache 2.2

```
<Directory "/">
    AllowOverride None
    Order allow,deny
    Allow from all
</Directory>
```

ab Apache 2.4

```
<Directory "/">
    AllowOverride None
    Require all granted
</Directory>
```

Damit sind alle Seiten die über den Proxy in Zukunft Freigegeben werden Kennwortgeschützt!

Dann laden wir die Apache Konfiguration erneut um die Änderungen zu übernehmen

```
sudo service apache2 reload
```

Wenn wir nun die Webseite erneut per https aufrufen werden wir nach Benutzernamen und Kennwort gefragt und sehen die Seite erst nach eingabe von unseren oben erstellten Benutzernamen / Kennwort.

Proxy Konfigurations Blöcke

Proxy für Homematic CCU

Dazu editieren wir wieder die datei

```
sudo vi /etc/apache2/sites-enabled/default-ssl.conf
```

und fügen am ende des **<VirtualHost _default_:443>** Tag den folgenden angepassten Block ein:

```
<IfModule mod_proxy.c>
    ProxyRequests off
    RewriteEngine On

    redirectmatch ^/ccu$ /ccu/

    rewritecond %{REQUEST_URI} ^/ccu/
    rewriterule (.*) $1 [PT]

    rewritecond %{HTTP_REFERER} https://<dns name>.<endung>/ccu/ [OR]
    rewritecond %{HTTP_REFERER} https://<dns name>.<endung>/webui/ [OR]
    rewritecond %{HTTP_REFERER} https://<dns name>.<endung>/pda/ [OR]
    rewritecond %{HTTP_REFERER} https://<dns name>.<endung>/api/ [OR]
    rewritecond %{HTTP_REFERER} https://<dns name>.<endung>/addons/cuxd/
[OR]
    rewritecond %{HTTP_REFERER} https://<dns name>.<endung>/addons/cuxchart/
[OR]
    rewritecond %{HTTP_REFERER} https://<dns name>.<endung>/ise/ [OR]
    rewritecond %{HTTP_REFERER} \?sid\=\@.+\@ [OR]
    rewritecond %{THE_REQUEST} \?sid\=\@.+\@

    rewriterule ^/(.*) /ccu/$1 [PT]
    rewriterule ^/pda/(.*) /ccu/pda/$1
    rewriterule ^/webui/(.*) /ccu/webui/$1
    rewriterule ^/addons/db/(.*) /ccu/addons/db/$1
    rewriterule ^/addons/cuxd/(.*) /ccu/addons/cuxd/$1
    rewriterule ^/addons/cuxchart/(.*) /ccu/addons/cuxchart/$1
    rewriterule ^/ise/(.*) /ccu/ise/$1

    ProxyPass /ccu/ http://<ip-der-ccu>/ timeout=1200
    ProxyPassReverse /ccu/ http://<ip-der-ccu>/ timeout=1200
</IfModule>
```

Alle Daten in < bla bla > sind durch eigene Daten zu ersetzen!

Nach jeder Änderung der Apache config ist diese neu einzulesen mit

```
sudo service apache2 reload
```

Das war es auf der Linux box, wenn sie schon den Dynamischen DNS Dienst und die Port Weiterleitung für das Port 443 (https) in dem Router / Firewall aktiviert haben ist ihre CCU aus dem Internet nun über folgende Adresse nach eingabe ihrer Zugangsdaten erreichbar:

```
https://name.dynamicdns.com/ccu
```

Proxy für CCU.IO, DASHUI, YAHUI und charts

Danke an MrLee aus dem homematic Forum für diesen Block!

```
<IfModule mod_proxy.c>
    ProxyRequests off
    RewriteEngine On
    redirectmatch ^/ccuio$ /ccuio/
    rewritecond %{REQUEST_URI} ^/ccuio/
    rewriterule (.*) $1 [PT]
    rewritecond %{HTTP_REFERER} https://mein.host.ie/ccuio/ [OR]
    rewritecond %{HTTP_REFERER} https://mein.host.ie/ccu.io/
    rewritecond %{HTTP_REFERER} https://mein.host.ie/css/ [OR]
    rewritecond %{HTTP_REFERER} https://mein.host.ie/js/ [OR]
    rewritecond %{HTTP_REFERER} https://mein.host.ie/fn/ [OR]
    rewritecond %{HTTP_REFERER} https://mein.host.ie/img/ [OR]
    rewritecond %{HTTP_REFERER} https://mein.host.ie/lib/ [OR]
    rewritecond %{HTTP_REFERER} https://mein.host.ie/lib/js/ [OR]
    rewritecond %{HTTP_REFERER} https://mein.host.ie/socket.io/ [OR]
    rewritecond %{HTTP_REFERER} https://mein.host.ie/lang/ [OR]
    rewritecond %{HTTP_REFERER} https://mein.host.ie/auth/ [OR]
    rewritecond %{HTTP_REFERER} https://mein.host.ie/dashui/ [OR]
    rewritecond %{HTTP_REFERER} https://mein.host.ie/charts/ [OR]
    rewritecond %{HTTP_REFERER} https://mein.host.ie/charts/css [OR]
    rewritecond %{HTTP_REFERER} https://mein.host.ie/charts/img [OR]
    rewritecond %{HTTP_REFERER} https://mein.host.ie/charts/js [OR]
    rewritecond %{HTTP_REFERER} https://mein.host.ie/log [OR]

    rewritecond %{HTTP_REFERER} https://mein.host.ie/yahui/ [OR]
    rewritecond %{HTTP_REFERER} https://mein.host.ie/yahui/css [OR]
    rewritecond %{HTTP_REFERER} https://mein.host.ie/yahui/img [OR]
    rewritecond %{HTTP_REFERER} https://mein.host.ie/yahui/js [OR]

    rewritecond %{HTTP_REFERER} https://mein.host.ie/eventlist/ [OR]
    rewritecond %{HTTP_REFERER} https://mein.host.ie/widgets/

    rewriterule ^/(.*) /ccuio/dashui/$1 [PT]
    rewriterule ^/ccu.io/(.*) /ccuio/ccu.io/$1 [PT]
```

```
    rewriterule ^/css/(.*) /ccuio/dashui/css/$1 [PT]
    rewriterule ^/js/(.*) /ccuio/dashui/js/$1 [PT]
    rewriterule ^/fn/(.*) /ccuio/dashui/fn/wifi/$1 [PT]
    rewriterule ^/img/(.*) /ccuio/dashui/img/$1 [PT]
    rewriterule ^/lib/(.*) /ccuio/lib/$1 [PT]
    rewriterule ^/lib/js/(.*) /ccuio/lib/js/$1 [PT]
    rewriterule ^/socket.io/(.*) /ccuio/socket.io/$1 [PT]
    rewriterule ^/lang/(.*) /ccuio/lang/$1 [PT]
    rewriterule ^/auth/(.*) /ccuio/auth/$1 [PT]
    rewriterule ^/dashui/(.*) /ccuio/dashui/$1 [PT]
    rewriterule ^/charts/(.*) /ccuio/charts/$1 [PT]
    rewriterule ^/charts/css(.*) /ccuio/charts/css$1 [PT]
    rewriterule ^/charts/img(.*) /ccuio/charts/img$1 [PT]
    rewriterule ^/charts/js(.*) /ccuio/charts/js$1 [PT]

    rewriterule ^/yahui/(.*) /ccuio/yahui/$1 [PT]
    rewriterule ^/charts/css(.*) /ccuio/yahui/css$1 [PT]
    rewriterule ^/charts/img(.*) /ccuio/yahui/img$1 [PT]
    rewriterule ^/charts/js(.*) /ccuio/yahui/js$1 [PT]

    rewriterule ^/eventlist/(.*) /ccuio/eventlist/$1 [PT]
    rewriterule ^/widgets/(.*) /ccuio/dashui/widgets/$1 [PT]
    rewriterule ^/log/(.*) /ccuio/log/$1 [PT]

    ProxyPass /ccuio http://<interne IP:Port>timeout=1200
    ProxyPassReverse /ccuio http://<interne IP:Port>timeout=1200
</IfModule>
```

Alle Daten in < bla bla > sind durch eigene Daten zu ersetzen!

Nach jeder Änderung der Apache config ist diese neu einzulesen mit

```
sudo service apache2 reload
```

Proxy für Enigma2 Receiver

Das hier gezeigt Beispiel geht davon aus das ihr auch das autotimer und den web bouquet editor installiert habt, ansonsten Kommentiert die Zeilen mit einem # am beginn aus wo autotimereditor und oder bouqueteditor vorkommen.

```
<IfModule mod_proxy.c>
      ProxyRequests off
      RewriteEngine On

      redirectmatch ^/dreambox$ /dreambox/
```

```
      rewritecond %{REQUEST_URI} ^/dreambox/
      rewriterule (.*) $1 [PT]

      rewritecond %{HTTP_REFERER} https://<domain.endung>/dreambox/ [OR]
      rewritecond %{HTTP_REFERER} https://<domain.endung>/web/ [OR]
      rewritecond %{HTTP_REFERER} https://<domain.endung>/webadmin/ [OR]
      rewritecond %{HTTP_REFERER}
https://<domain.endung>/dreambox/autotimereditor/ [OR]
      rewritecond %{HTTP_REFERER}
https://<domain.endung>/dreambox/bouqueteditor/
      rewriterule ^/(.*) /dreambox/$1 [PT]
      rewriterule ^/webadmin/(.*) /dreambox/webadmin/$1 [PT]
      rewriterule ^/web/(.*) /dreambox/web/$1 [PT]
      rewriterule ^/autotimereditor/(.*) /dreambox/autotimereditor/$1 [PT]
      rewriterule ^/bouqueteditor/(.*) /dreambox/bouqueteditor/$1 [PT]

      ProxyPass /dreambox/ http://<ip der Enigma2 box>/ timeout=1200
      ProxyPassReverse /dreambox/ http://<ip der Enigma2 box>/ timeout=1200
</IfModule>
```

Alle Daten in < bla bla > sind durch eigene Daten zu ersetzen!

Nach jeder Änderung der Apache config ist diese neu einzulesen mit

```
sudo service apache2 reload
```

Proxy für Logitech Media Server

Das hier gezeigt Beispiel ist für einen squeezebox Media Server:

```
<IfModule mod_proxy.c>
      ProxyRequests off
      RewriteEngine On

      redirectmatch ^/squeeze$ /squeeze/

      rewritecond %{REQUEST_URI} ^/squeeze/
      rewriterule (.*) $1 [PT]

      rewritecond %{HTTP_REFERER} https://<dns name>.<endung>/squeeze/ [OR]
      rewritecond %{HTTP_REFERER} https://<dns name>.<endung>/clixmlbrowser/
[OR]
      rewritecond %{HTTP_REFERER} https://<dns name>.<endung>/settings/ [OR]
      rewritecond %{HTTP_REFERER} https://<dns name>.<endung>/html/ [OR]
      rewritecond %{HTTP_REFERER} https://<dns name>.<endung>/plugins/ [OR]
```

```
      rewritecond %{HTTP_REFERER} https://<dns name>.<endung>/music/
      rewriterule ^/(.*) /squeeze/$1 [PT]
      rewriterule ^/clixmlbrowser/(.*) /squeeze/clixmlbrowser/$1 [PT]
      rewriterule ^/settings/(.*) /squeeze/settings/$1 [PT]
      rewriterule ^/html/(.*) /squeeze/html/$1 [PT]
      rewriterule ^/plugins/(.*) /squeeze/plugins/$1 [PT]
      rewriterule ^/music/(.*) /squeeze/music/$1 [PT]

      ProxyPass        /squeeze http://<ip des LMS>:<port>    timeout=1200
      ProxyPassReverse /squeeze http://<ip des LMS>:<port>    timeout=1200
</IfModule>
```

Alle Daten in < bla bla > sind durch eigene Daten zu ersetzen!

Nach jeder Änderung der Apache config ist diese neu einzulesen mit

```
sudo service apache2 reload
```

Proxy für Openmediavault 3

```
<IfModule mod_proxy.c>
      ProxyRequests off
      RewriteEngine On

      redirectmatch ^/omv$ /omv/

      rewritecond %{REQUEST_URI} ^/omv/
      rewriterule (.*) $1 [PT]

      rewritecond %{HTTP_REFERER} https:/<dns name>.<endung>/extjs/
      rewriterule ^/extjs/(.*) /omv/extjs/$1

      ProxyPass /omv/ http://<ip von OMV>/ timeout=1200
      ProxyPassReverse /omv/ http://<ip von OMV>/ timeout=1200
</IfModule>
```

Alle Daten in < bla bla > sind durch eigene Daten zu ersetzen!

Nach jeder Änderung der Apache config ist diese neu einzulesen mit

```
sudo service apache2 reload
```

Proxy für Homematic CCU2 Rega

Damit die Rega Schnittstelle der CCU Sicher mit https und Athentifizierung aus dem Internet genutz werden kann folgenden Block einfügen und dann ist diese erreichbar unter:

```
https://<dns name>.<endung>/rega/
```

```
<IfModule mod_proxy.c>
        ProxyRequests off
        RewriteEngine On

        redirectmatch ^/rega$ /rega/

        rewritecond %{REQUEST_URI} ^/rega/
        rewriterule (.*) $1 [PT]

        rewritecond %{HTTP_REFERER} https://<dns name>.<endung>/rega/
        rewriterule ^/(.*) /rega/$1 [PT]

        ProxyPass   /rega/  http://<ip-der-ccu>:8181/ timeout=1200
        ProxyPassReverse    /rega/  http://<ip-der-ccu>:8181/ timeout=1200
</IfModule>
```

Proxy für XEOMA Videoüberwachungsserver

es muss das apache Modul substitude geladen werden:

```
a2enmod substitute headers
```

Dann folgenden Block eintragen:

```
<IfModule mod_proxy.c>
    ProxyRequests Off
    RewriteEngine On
    ProxyPreserveHost Off
    AllowEncodedSlashes On
    KeepAlive Off

    RequestHeader unset Authorization

    redirectmatch ^/video$ /video/
    rewritecond %{REQUEST_URI} ^/video/
    rewriterule (.*) $1 [PT]

    rewritecond %{HTTP_REFERER} https://<name>.<domain>.<endung>/video/
    rewriterule ^/(.*) /video/$1 [PT]
```

```
    ProxyPass        /video  http://<ip des video servers>:10090
    ProxyPassReverse /video http://<ip des video servers>:10090

    AddOutputFilterByType SUBSTITUTE text/html
    Substitute "s|http://<ip des video
servers>:10090|https://<name>.<domain>.<endung>/video|i"
</IfModule>
```

Proxy für IP-Symcon

folgenden Block eintragen:

```
<IfModule mod_proxy.c>
    ProxyRequests off
    RewriteEngine On

    redirectmatch ^/symcon$ /symcon/

    rewritecond %{REQUEST_URI} ^/symcon/
    rewriterule (.*) $1 [PT]

    rewritecond %{HTTP_REFERER} https://<name>.<domain>.<endung>/symcon/
    rewriterule ^/(.*) /symcon/$1 [PT]

    ProxyPass /symcon/ http://<ip vom symcon server>:<port>/ timeout=1200
    ProxyPassReverse        /rega/ http://<ip vom symcon server>:<port>/
timeout=1200
</IfModule>
```

Beispiel für die Apache SSL Konfigurationsdatei

Solltet ihr Probleme haben so vergleicht den Inhalt der Datei mit der Untenstehenden oder Übernehmt diese mit der Apassung des DNS Namens (<eigener-dns-Name> 5x) und der IP eurer CCU (<internet-ip-der-ccu> 1x) mit:

```
sudo vi /etc/apache2/sites-enabled/default-ssl.conf
```

```
  <IfModule mod_ssl.c>
    <VirtualHost _default_:443>
        ServerName <subdomain>.<domain>.<endung>
        ServerAdmin webmaster@localhost
        DocumentRoot /var/www
        <Directory />
            Options FollowSymLinks
            AllowOverride None
        </Directory>
```

```
        <Directory /var/www/>
            Options Indexes FollowSymLinks MultiViews
            AllowOverride None
            Order allow,deny
            allow from all
        </Directory>
        <Location />
                Deny from all
                AuthType basic
                AuthName "home"
                AuthUserFile  /etc/apache2/ssl/httpsusers
                Satisfy Any
                Require valid-user
        </Location>
        ScriptAlias /cgi-bin/ /usr/lib/cgi-bin/
        <Directory "/usr/lib/cgi-bin">
            AllowOverride None
            Options +ExecCGI -MultiViews +SymLinksIfOwnerMatch
            Order allow,deny
            Allow from all
        </Directory>
        SSLCertificateFile    /etc/apache2/ssl/server.crt
        SSLCertificateKeyFile /etc/apache2/ssl/server.key
        ErrorLog ${APACHE_LOG_DIR}/error.log
    <IfModule mod_proxy.c>
        ProxyRequests off
        RewriteEngine On
        redirectmatch ^/ccu$ /ccu/
        rewritecond %{REQUEST_URI} ^/ccu/
        rewriterule (.*) $1 [PT]
        rewritecond %{HTTP_REFERER} https://<eigener-dns-Name>/ccu/ [OR]
        rewritecond %{HTTP_REFERER} https://<eigener-dns-Name>/webui/ [OR]
        rewritecond %{HTTP_REFERER} https://<eigener-dns-Name>/pda/ [OR]
        rewritecond %{HTTP_REFERER} https://<eigener-dns-Name>/api/ [OR]
        rewritecond %{HTTP_REFERER} https://<eigener-dns-Name>/addons/cuxd/
[OR]
        rewritecond %{HTTP_REFERER} \?sid\=\@.+\@ [OR]
        rewritecond %{THE_REQUEST} \?sid\=\@.+\@
        rewriterule ^/(.*) /ccu/$1 [PT]
        rewriterule ^/pda/(.*) /ccu/pda/$1
        rewriterule ^/webui/(.*) /ccu/webui/$1
        rewriterule ^/addons/db/(.*) /ccu/addons/db/$1
        rewriterule ^/addons/cuxd/(.*) /ccu/addons/cuxd/$1
        ProxyPass /ccu/ http://<internet-ip-der-ccu>/ timeout=1200
        ProxyPassReverse /ccu/ http://<internet-ip-der-ccu>/ timeout=1200
    </IfModule>
        LogLevel warn
        CustomLog ${APACHE_LOG_DIR}/ssl_access.log combined
        SSLEngine on
        <FilesMatch "\.(cgi|shtml|phtml|php)$">
```

```
            SSLOptions +StdEnvVars
        </FilesMatch>
        <Directory /usr/lib/cgi-bin>
            SSLOptions +StdEnvVars
        </Directory>
        BrowserMatch "MSIE [2-6]" \
            nokeepalive ssl-unclean-shutdown \
            downgrade-1.0 force-response-1.0
        # MSIE 7 and newer should be able to use keepalive
        BrowserMatch "MSIE [17-9]" ssl-unclean-shutdown
    </VirtualHost>
    </IfModule>
```

Homedroid Einstellungen für den Zugriff auf die CCU über diesen Reverse Proxy

Geben sie in der Konfiguration von Homedroid folgende Daten ein:

- Serveradresse name.dynamicdns.com/ccu/
- Unter Netzwerkeinstellungen den Hacken bei **HTTPS verwenden** setzen.
- Unter „Experimental Settings“ nach Bestätigung das sie wissen was sie tun einen Hacken bei **Enable Http Auth** setzen und die Zugangsdaten bei **Benutzername** und **Kennwort** eingeben die sie vergeben haben!

Problembehebung

Basic Authentication Header einer weitergeleiteten Seite entfernen

Wenn sie das Problem haben das eine Seite aus ihrem Netzwerk nach der Apache Reverse Proxy Authentication erneut nach einem Kennwort Fragt können sie diese Abfrage entfernen indem sie noch folgendes Modul im Apache Proxy laden:

```
sudo a2enmod header
```

und folgenden Eintrag für diesen Reverse Proxy in die Apache Konfig schreiben:

```
<IfModule mod_proxy.c>
    ProxyRequests Off
    ProxyPreserveHost Off
    AllowEncodedSlashes On
    KeepAlive Off
    RequestHeader unset Authorization
```

```
    ProxyPass        /<seite>  http://<ip>:<port>
    ProxyPassReverse /<seite> http://<ip>:<port>
</IfModule>
```

Alle Daten in < bla bla > sind durch eigene Daten zu ersetzen!

Nach jeder Änderung der Apache config ist diese neu einzulesen mit

```
sudo service apache2 reload
```

Weitere Seiten einbinden

Wenn auch andere Seiten wie die einer Webcam (z.B. Foscam) durch den Reverse Proxy verfügbar gemacht werden können sie das einfacher haben solange das Webinterface nicht zu viele Unterseiten hat so wie bei meiner Foscam Webcam durch änderen der Datei:

```
sudo vi /etc/apache2/sites-enabled/default-ssl.conf
```

und fügen am ende des **<VirtualHost _default_:443>** Tag den folgenden angepassten Block ein:

```
ProxyPass        /cam01/ http://<ip>:<port>/
ProxyPassReverse /cam01/ http://<ip>:<port>/
```

Absichern des Zugangs mit fail2ban

Mit dem im standard Repository von Debian vorhandenen paket fail2ban kann einfach das HTTPS Login überwacht und effektiv vor „Brute Force“ Attacken geschützt werden. Wenn also jemand öfter als 6 mal einen Fehlerhaftes Login versucht wird er (die IP Adresse) für x Minuten in der Firewall komplett gesperrt und kann keine weiteren Versuche machen bis diese Zeit abgelaufen ist.

Installation

```
aptitude install fail2ban
```

Einrichten Raspbian / Debian 9

für https apache muss in der Datei

```
sudo vi /etc/fail2ban/jail.d/defaults-debian.conf
```

Und die Apache Auth Rule aktivieren indem folgende Zeilen dort angefügt werden damit die Datei so aussieht:

```
[sshd]
enabled = true

[apache-auth]
enabled = true

[apache-noscript]
enabled = true

[apache-overflows]
enabled = true

[apache-nohome]
enabled = true

[apache-botsearch]
enabled = true

[apache-fakegooglebot]
enabled = true

[apache-modsecurity]
enabled = true

[apache-shellshock]
enabled = true
```

Dann den fail2ban Dienst neustarten mit:

```
service fail2ban restart
```

Nun kann im fail2ban log gesehen werden wenn jemand ge„banned“ wurde

```
tail -f /var/log/fail2ban.log
```

Einrichten Raspbian / Debian 7 oder 8

Dann muss nur noch der apache aktiviert werden und in der log Datei

```
vi /etc/fail2ban/jail.conf
```

folgender Block

```
#
# HTTP servers
#

[apache]

enabled  = true
port     = http,https
filter   = apache-auth
logpath  = /var/log/apache*/*error.log
maxretry = 6
```

dann noch den Dienst mit der geänderten Konfiguration neu laden mit:

```
service fail2ban reload
```

Überprüfen

Das log in /var/log/fail2ban.log sollte nun so aussehen:

```
09:34:52,594 fail2ban.server : INFO   Changed logging target to
/var/log/fail2ban.log for Fail2ban v0.8.6
09:34:52,605 fail2ban.jail   : INFO   Creating new jail 'ssh'
09:34:52,611 fail2ban.jail   : INFO   Jail 'ssh' uses poller
09:34:52,901 fail2ban.filter : INFO   Added logfile = /var/log/auth.log
09:34:52,912 fail2ban.filter : INFO   Set maxRetry = 6
09:34:52,929 fail2ban.filter : INFO   Set findtime = 600
09:34:52,938 fail2ban.actions: INFO   Set banTime = 600
09:34:53,868 fail2ban.jail   : INFO   Jail 'ssh' started
09:39:17,379 fail2ban.server : INFO   Stopping all jails
09:39:18,316 fail2ban.jail   : INFO   Jail 'ssh' stopped
09:39:18,345 fail2ban.server : INFO   Changed logging target to
/var/log/fail2ban.log for Fail2ban v0.8.6
09:39:18,366 fail2ban.jail   : INFO   Creating new jail 'ssh'
09:39:18,467 fail2ban.jail   : INFO   Jail 'ssh' uses Gamin
09:39:18,724 fail2ban.filter : INFO   Added logfile = /var/log/auth.log
09:39:18,742 fail2ban.filter : INFO   Set maxRetry = 6
09:39:18,774 fail2ban.filter : INFO   Set findtime = 600
09:39:18,788 fail2ban.actions: INFO   Set banTime = 600
09:39:19,298 fail2ban.jail   : INFO   Creating new jail 'apache'
09:39:19,301 fail2ban.jail   : INFO   Jail 'apache' uses Gamin
09:39:19,329 fail2ban.filter : INFO   Set maxRetry = 6
09:39:19,364 fail2ban.filter : INFO   Set findtime = 600
09:39:19,381 fail2ban.actions: INFO   Set banTime = 600
09:39:19,602 fail2ban.jail   : INFO   Jail 'ssh' started
09:39:19,752 fail2ban.jail   : INFO   Jail 'apache' started
```

Reverse Proxy Paralell mit Authentifizierung und ohne einrichten (owncloud)

Wenn neben dem Reverse Proxy auch andere Webseiten wie z.B. owncloud ohne Authentifizierung betrieben werden sollen dann empfiehlt sich die Einrichtung eines weiteren Virtualhost unter dem die Dienste ohne Authentifizierung erreichbar sind. Also wird z.B. unter zuhause.<domain>.<endung> immer die Authentifizierung für alle Anfragen benötigt aber für cloud.<domain>.<endung> nicht.

Dies ist nur eine Möglichkeit mit der dies Realisiert werden kann, Alternativ kann auch im Bestehenden Proxy für jede Seite die Athentifizierung aktivert / deaktivert werden, nur finde ich die Lösung mit der eigenen Domain / Subdomain für Gesicherten Zugang und einer anderen für den normalen „sauberer“.

Domains / Subdomains

Damit dies Funktionieren kann müsst ihr zwei domains (subdomains) für die ip Adresse eures Apache Servers einrichten es wird also unter z.B.:

```
zuhause.<domain>.<endung>
```

Die Athentifizierung des Reverse Proxy kommen und dahinter könnt ihr sicher auf die ansonsten ungeschützen Inhalte zugreifen, aber unter z.B.:

```
cloud.<domain>.<endung>
```

Kommt ihr direkt ohne Apache Authentifizierung zu owncloud.

Owncloud

Die Installation von Onwcloud ist am besten über die von Ownloud verfügbaren Repositorys durchzuführen, diese und die Anleitung findet ihr hier.

Apache2 Module

folgende Module sollten aktiviert sein, zum Auflisten der aktuell geladenen Module einfach:

```
sudo apache2ctl -M
```

ausführen, dies sollte folgende Module ausgeben.

```
Loaded Modules:
 core_module (static)
 log_config_module (static)
 logio_module (static)
 version_module (static)
 mpm_prefork_module (static)
 http_module (static)
 so_module (static)
 alias_module (shared)
 auth_basic_module (shared)
 authn_file_module (shared)
 authz_default_module (shared)
 authz_groupfile_module (shared)
 authz_host_module (shared)
 authz_user_module (shared)
 autoindex_module (shared)
 cgi_module (shared)
 deflate_module (shared)
 dir_module (shared)
 env_module (shared)
 mime_module (shared)
 negotiation_module (shared)
 php5_module (shared)
 proxy_module (shared)
 proxy_html_module (shared)
 proxy_http_module (shared)
 reqtimeout_module (shared)
 rewrite_module (shared)
 setenvif_module (shared)
 ssl_module (shared)
 status_module (shared)
Syntax OK
```

Sollten Module bei euch nicht geladen sein diese einfach mit:

```
sudo a2enmod <modulname>
```

aktivieren.

SSL Zertifikat für die zweite domain / subdomain erstellen

Dies müsst ihr wie hier erneut aber mit der anderen domain / subdomain und einem anderen dateinamen (im Beispiel unten ist es /etc/apache2/ssl/cloud.server.crt und /etc/apache2/ssl/cloud.server.key) für die Zertifikate ausführen!

Apache2 sites-enabled

Die https Konfiguration des webservers in:

```
/etc/apache2/sites-enabled/default-ssl.conf
```

sollte so aussehen:

```
<IfModule mod_ssl.c>
<VirtualHost *:443>
        ServerName cloud.<domain>.<endung>

        DocumentRoot /var/www/owncloud

        <Directory /var/www/owncloud/>
                AllowOverride All
        </Directory>

        #   SSL Engine Switch:
        #   Enable/Disable SSL for this virtual host.
        SSLEngine on

        #   /usr/share/doc/apache2.2-common/README.Debian.gz for more info.
        #   If both key and certificate are stored in the same file, only
the
        #   SSLCertificateFile directive is needed.
        SSLCertificateFile    /etc/apache2/ssl/cloud.server.crt
        SSLCertificateKeyFile /etc/apache2/ssl/cloud.server.key

</VirtualHost>

<VirtualHost *:443>
        ServerName zuhause.<domain>.<endung>
        ServerAdmin webmaster@localhost

        DocumentRoot /var/www/home
        <Directory />
                Options FollowSymLinks
                AllowOverride None
        </Directory>
        <Directory /var/www/home/>
                Options Indexes FollowSymLinks MultiViews
                AllowOverride None
                Order allow,deny
                allow from all
        </Directory>

        ScriptAlias /cgi-bin/ /usr/lib/cgi-bin/
        <Directory "/usr/lib/cgi-bin">
                AllowOverride None
```

```
                Options +ExecCGI -MultiViews +SymLinksIfOwnerMatch
                Order allow,deny
                Allow from all
        </Directory>

        <Location />
            Deny from all
            AuthType basic
            AuthName "zuhause"
            AuthUserFile  /etc/apache2/ssl/httpsusers
            Satisfy Any
            Require valid-user
        </Location>

        ErrorLog ${APACHE_LOG_DIR}/error.log
        LogLevel warn

        CustomLog ${APACHE_LOG_DIR}/ssl_access.log combined
        SSLEngine on
        SSLCertificateFile    /etc/apache2/ssl/server.crt
        SSLCertificateKeyFile /etc/apache2/ssl/server.key
        ####################################
        # PROXY's START
        # ...
        # PROXY's ENDE
        ####################################
       </VirtualHost>
</IfModule>
```

Alle in **<**stehenden**>** Texte müssen sowie die domain / subdomain namen und Pfade and die eigenen angepasst werden!

Nach den Änderungen nicht vergessen die config zu testen mit:

```
apache2ctl -S
```

und den Apache neu zu laden:

```
service apache2 reload
```