Include support for HTTPS on CoovaChilli

Introduction

- In recent years many well known websites are now by default served through https (server on port 443 and includes encryption)
- CoovaChilli was originally only capturing http traffic (served on port 80 and does not include encryption)
- There are however some advanced features included with CoovaChilli which allow it to also capture traffic going to port 443.
- This document will go through the steps you have to go through to get it working with the MESHdesk / APdesk firmware.

Requirements

- A server with a public IP Address and FQDN having NGINX configured with a signed certificate by a known Certificate Authority (CA)
- A LEDE build environment as stipulated in the page's parent document.
- In our example we use here we have a server with a FQDN of 01.wifi-dashboard.com
- We followed the standard instructions from LetsEncrypt to obtain a certificate for NGINX. (https://letsencrypt.org/)

Actions

- Copy the **fullchain.pem** and **privkey.pem** files from the public server to machine with the LEDE build environment. (For the location of these files, consult your NGINX setup)
- Copy these files to the **package/zzz-MESHdesk/files/MESHdesk/captive_portals/** folder.
- We rename **fullchain.pem** to **cert.pem**.
- We rename **privkey.pem** to **key.pem**.

```
cd ~/lede
#We assume the files are under the lede directory
cp  fullchain.pem ./source/package/zzz-
MESHdesk/files/MESHdesk/captive_portals/cert.pem
cp  privkey.pem ./source/package/zzz-
MESHdesk/files/MESHdesk/captive_portals/key.pem
```

- Edit the **package/zzz-MESHdesk/files/MESHdesk/captive_portals/common.conf** file. Adapt to your enviroment

```
#Change this to the value of the domain of the certificate we are using e.g.
wifi-dashboard.com in our case (01.wifi-dashboard.com)
domain "wifi-dashboard.com"
#Change the default to be the same as the servername where the certificate
is issued e.g. 01 in our case (01.wifi-dashboard.com)
uamaliasname "01"
```

```
redirssl
uamuissl
sslkeyfile=/etc/MESHdesk/captive_portals/key.pem
sslcertfile=/etc/MESHdesk/captive_portals/cert.pem
```

- Issue the **make menuconfig** command under the ~/lede/source directory.
- Go and select **Network** → **Captive Portal** → **coova-chilli** → **Configuration** → SSL Library → OpenSSL
- Save the changes and issue a **make** command to complete a new build with the HTTPS support included with CoovaChilli.
- Next we will look at all the technical issues around our work we just completed.

All the technical things

uamuissl

- When we specify **uamuissl** we enable the CoovaChilli Captive Portal to listen on port 4990 for requests (on top of the default of 3990)
- These requests will be served over https and is thus encrypted.
- We need to do this if we want to serve the login pages over https since many browsers does not allow you to '*step down*' e.g. going from https and then try to do Ajax request over http.
- We also specify which certificates the CoovaChilli captive portal will use for this https transactions.
- When **uamuissl** is enabled the captive portal login page will include a **ssl** item in the query string e.g. **ssl=https%3a%2f%2f01.wifi-dashboard.com%3a4990%2f**
- Our dynamic login page has some logic build in to look for this and automatically then replace the queries to the CoovaChilli controller's http port (3990) with (4990).
- You will see that the ssl item has a value of: **https://01.wifi-dashboard.com:499/**
- By default if you try and ping this FQDN if will resolve to the public IP Address the server is running on.
- Coova however pulls a fast one and intercepts DNS requests to the DNS server for **01.wifi-dashboard.com** and replaces it with the value if the UAMIP. (typically 10.100.0.1 or something similar)
- *How does it know what to intercept and replace?* It looks at the value of **uamaliasname** and **domain** and concatenate them.
- *What if I specified my own DNS server?* Then it will not work since it only replaces the reply if the query was to one of the DNS servers which Coova uses.
- *Why does it do it?* The browser will go to a FQDN in the URL and NOT an IP Address. It then determines that the certificate is belonging to a FQDN. Then it ask the DNS server what is the IP Address of this FQDN. Once it has an answer it can compare with the IP Address it is visiting and confirm that hey are the same, else it will complain.

redirssl

- **redirssl** tells CoovaChilli to also listen on port 443 for incoming traffic.
- If Coova is listening on port 443 it will answer as if it is the other side of the conversation. e.g. it will pretend to be https://www.youtube.com. \
- Most browsers however are working in such a way as to check the certificate it got during the

https transaction. If this certificate is not the same as that for which site it wants to go to it usually have a redirect to the login page mechanism.
- Alternatively it will just stop with a message saying it thinks there is a man in the middle.
- This is because in our case it expected Youtube's certificate but got WiFi-Dashboard instead.
- As we mentioned many modern gadgets and operating systems support redirecting this to a login page though there will definitely be some devices that complains outright.
- This is the trade-off between not having https support and clients assume it does not work, also this should become better supported over time.