

FHIR Proxy Installation

05 – Production Install

Version 1 – February 2021

Table of Contents

[1 Introduction 4](#_Toc65527933)

[1.1 Purpose of this Document 4](#_Toc65527934)

[2 Production Design Guidance 5](#_Toc65527935)

[2.1 Design Considerations 5](#_Toc65527936)

[2.1.1 Resilient deployment of the FHIR Proxy 5](#_Toc65527937)

[2.1.2 Resilient deployment of the database 5](#_Toc65527938)

[2.1.3 Network security and proxies 5](#_Toc65527939)

[2.1.4 SSL Termination 6](#_Toc65527940)

[2.1.5 Internal vs External connectivity 6](#_Toc65527941)

[2.2 Illustrative Deployment 7](#_Toc65527942)

[3 FHIR Proxy Installation 9](#_Toc65527943)

[3.1 Production Install Steps 9](#_Toc65527944)

[3.2 Applying Updates 13](#_Toc65527945)

[4 Technical Onboarding 14](#_Toc65527946)

[4.1 Introduction 14](#_Toc65527947)

[4.2 Firewall Configuration 14](#_Toc65527948)

[4.3 Certificates and Keys – Inbound from YHCR 15](#_Toc65527949)

[4.4 Certificates and Keys – Outbound to YHCR 17](#_Toc65527950)

**Version Control**

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Release Date** | **Released By** | **Reason for Release** |
| 1 | 22/02/2021 | Tim Davey | Preliminary Draft |
|  |  |  |  |

**Reviewers**

|  |  |  |  |
| --- | --- | --- | --- |
| **Initials** | **Name** | **Role** | **Organisation** |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

# Introduction

## Purpose of this Document

This document is part of a set which walks through the entire process of installing the FHIR Proxy and connecting to YHCR. It is assumed that the preceding document(s) have already been read, and material already covered will not be repeated.



This document looks at planning for a Production install.

* It assumes that, based on working through the “quickstart” exercise, then you already have a good understanding of the FHIR Proxy software, its configuration, and functionality
* In this document we therefore give guidance on moving from a simple demo install to a production configuration. The document collates as much real-life experience as possible to help you achieve a successful production deployment

**Note that the guidance given here cannot be entirely “step-by-step” or prescriptive. There are too many variations in local policy and practice to cover every eventuality. Based on the information provided you will need to make your own design decisions - and configure your own infrastructure in a way which combines this guidance with your own organisation’s security and infrastructure best-practices and policies.**

# Production Design Guidance

## Design Considerations

In this section we highlight some of the key questions which you will need to consider when designing your production deployment.

***Fundamentally the FHIR Proxy is similar to any other “web app” – consisting of a server which responds to HTTP(S) plus a database backend. The headings below provide more detail of these components. However if you already have an approach to deploying this kind of standard web-based system then the FHIR Proxy is likely to fit the same pattern.***

***The only real points of note are the need to use YHCR-signed certificates for TLS (including Mutual Authentication), plus the option for an “internal” vs “external” route into the service.***

### Resilient deployment of the FHIR Proxy

This is the core software provided by YHCR. It provides a stateless RESTful service responding to HTTP traffic. Resilience is therefore achieved in a similar way to any other “website” – ie by deploying in a load-balanced configuration across at least two servers.

### Resilient deployment of the database

Resilience for the database also needs to be planned for, for example:

* ***Redundancy*** – normal best-practices for avoiding a single point of failure in the database need to be considered - whether active-passive failover or an active-active cluster. It is beyond the scope of this document to cover this, however see Appendix A for references.
* ***Backups*** – again it is important that the database is backed up, and normal best-practices apply. Backups of the audit schema provide an additional function, in terms of providing assurance that a temper-proof copy of the audit trail is securely archived
* ***Management*** – the database needs to be managed and maintained in line with normal DBA best-practices
  + The ***“public” schema*** will contain record(s) for each FHIR Resource published – in many cases this is likely to consume only a small amount of storage. A fast response to queries on the indexes will be required.
  + The ***“audit” schema*** will contain record(s) for each request made to the FHIR server. Depending on traffic this could get large. It is however append-only, and with queries likely to be rare.

Note that it is possible to co-locate the FHIR Proxy and Database on a single server if desired for test deployments. However, as for any application, the different characteristics of an application server vs database mean that dedicated servers are best-practice for production.

### Network security and proxies

It is assumed that, as with any web application, the production FHIR Proxy will not be directly connected to external networks – rather it will be protected by a DMZ and Reverse Proxy layer.

The external-facing endpoint will need to present a static IP address which can be configured in the YHCR to route messages to your organisation. (YHCR does not by default use public DNS routing). Your organisation’s firewalls will also need to be opened to allow access for HTTPS traffic from the YHCR’s static IP address.

### SSL Termination

Related to the previous point is deciding where SSL from incoming YHCR connections will be terminated. Key points to consider are:

* It is required to use YHCR-signed certificates for SSL
* It is also required to TLS Mutual Authentication (client authentication) using YHCR-signed certificates

You will also need to consider whether or not SSL is furthermore needed for traffic on your own internal networks.

Planning will therefore be needed as to how these YHCR certificate requirements are incorporated. It is obviously easiest on a separate web server, however most modern web servers also provide options to configure multiple services on a single box. ***YHCR FHIR Proxy Install 99 - 3rd Party Technologies*** provides further references.

### Internal vs External connectivity

* The main “external” route to the server will come via the network security layers discussed above, and will require signed JWT tokens to be included with each message.
* There may also be a need for internal services (eg the integration engine loading the data) to connect, and therefore an option to have a simpler “internal” route for these to use.

## Illustrative Deployment

This section provides an example of how a deployment might look, based on the considerations above.

***IMPORTANT: This example is illustrative only and is not the “right” or “only” answer. You will need to design your own deployment based on knowledge of your own internal infrastructure and policies.***



Points of note in the diagram include:

* The YHCR ***FHIR Proxy*** software is hosted on Docker, via a Docker Compose file which includes an internal endpoint (listening on Port 3000), plus an external endpoint (listening on Port 3001, and with JWT validation enabled)[[1]](#footnote-2). It is deployed across two application servers for resilience.
* The ***database*** is deployed in an active-passive configuration – with a primary and log-shipping to a secondary replica. (Other options include an active-active configuration)
* A resilient ***Web Proxy*** layer is deployed in the DMZ, to provide reverse-proxy and Layer 7 Load Balancing services. It offers a static VIP for YHCR to connect to and provides TLS termination using the YHCR certificates for both encrypting the connection and TLS Mutual Authentication. (The diagram shows dedicated infrastructure, however there should also be options for segregated configuration and routing on shared proxy infrastructure).
* ***Network connections within the organisation*** are shown using HTTP. Alternatively it might be required to enable SSL for some or all of the internal connections. Additional internal SSL proxies would be required to achieve this. One option could be to use the dockerised Traefik proxy (as per the “quickstart” configuration) to enable SSL for the FHIR Proxy – but alternatively another choice of internal proxy server could be deployed. (Built-in SSL is another potential future feature for the FHIR Proxy).
* The ***integration engine*** is shown with a very simple configuration connecting to one of the FHIR Proxy instances. This assumes that resilience would be provided via queuing and retries, with application-level monitoring and error handling to trigger switching to the other server in the event of an outage. (Alternatively it could be connected to an internal load-balancer across the two application servers)
* ***Disaster Recovery*** is not explicitly shown in the diagram, but the secondary database might normally be on a remote site. The other servers are stateless and could have server images replicated for warm or cold standby to a remote site. Other more elaborate active-active arrangements would also be possible.

# FHIR Proxy Installation

Having considered the end-to-end design, this section walks through the detailed production installation of the YHCR FHIR Proxy component.

## Production Install Steps

1. **Prepare the server**

* See the instructions in document ***YHCR FHIR Proxy Install 02 - Server Preparation*** for more details, including server prerequisites and advice on installing docker

1. **Copy on production configuration**

<<TODO – confirm where to get them from. There might also be another repo for developers who want the code?>>

* If you have git installed then use the command: git clone <https://github.com/synanetics/synfhir-store.git>
* Alternatively, open in a browser to <https://github.com/synanetics/synfhir-store>, look for the green “Code” button, select “Download ZIP”, and unzip the files

The location of the install on the server is your choice

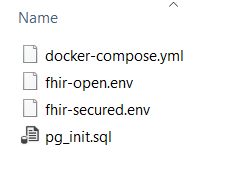
Once you have a copy of the files downloaded then look inside the “prod” folder

1. **Install other servers and components**

* You will need to provision other servers and install any other 3rd party components – eg database, web proxy. It is beyond the scope of this document to provide detailed instructions for these non-YHCR components – but see “***YHCR FHIR Proxy Install 99 - 3rd Party Technologies***” for links and tips which may be useful.
* Note that you will need to have at least a working database connection to run the FHIR Proxy for an initial internal test. This includes:
  + Creating a database called “fhirstore”
  + Creating a logon for the FHIR Proxy to use, and setting as owner of the fhirstore database (eg “iamonfhir”)
  + Running the database initialisation script (ie pg\_int.sql – see below)
  + Opening firewalls and database connectivity between the FHIR Proxy and database servers

See “***YHCR FHIR Proxy Install 99 - 3rd Party Technologies***” for further tips on these tasks

1. **Review configuration files**

****

* ***pg\_init.sql*** – this contains a SQL script to initialise the postgress database tables.
* ***Manually run this script on the database server***
* ***fhir-open.env and fhir-secured.env*** – configuration files for each of the FHIR Proxy instances. The database connection string will need to be set:
  + ***PG\_CONNECTION*** – the Postgres database connection string in format ***postgresql://[USER]:[PASSWORD]@[DB\_HOST]:[DB\_HOST\_PORT]/fhirstore***

For example:

*postgresql://iamonfhir:password123@10.1.3.8:5432/fhirstore*

***TIP****: Even if it is a test install and Postgres is on the same box, use a “proper” IP address here and not loopback*

There are many other settings in this file which you should check through – see ***YHCR FHIR Proxy Install 03 - Quickstart Install*** for further details

* ***docker-compose.yaml*** – this is the main Docker Compose configuration file which describes the environment. It is configured to spin up two instances of the FHIR Proxy – one secured with JWT validation, and the other “open” for internal use.
  + ***Choose your flavour*** – you will see the “image:” details, with an obvious choice between Linux or Windows 2019 for each instance. Comment in/out the appropriate lines to make your choice
  + ***Confirm ports*** – by default the endpoints are exposed on ports 3000 and 3001. However this is easy to change if desired (simply search and replace)

1. **Check internal firewalls**

* You will need to make sure that any internal and/or server firewalls are open for the necessary connections – eg to access the FHIR Proxy ports, and for the FHIR Proxy to connect to the database. (For example Windows Firewall will block all of these by default)

1. **Run up the system**

* Ensure your current directory is the “prod” directory where the “docker-compose.yml” file is located
* Enter the command to spin up the installation: ***docker-compose up***. (On Linux this may need to be prefixed with sudo).
  + As usual, the first time will be slow as it downloads all of the docker images. Subsequent runs will be much faster

1. **Smoke Test**

* The most obvious basic test is to try ***<http://localhost:3000/fhir/stu3/metadata>.***
* You can also try[***http://localhost:3001/fhir/stu3/metadata***](http://localhost:3001/fhir/stu3/metadata) – which should return an Operation Outcome stating “Unauthorised request”
* You can also try just [***http://localhost:3000***](http://localhost:3000) and [***http://localhost:3001***](http://localhost:3001) – both of which should return simply “Ping”. This can be useful for healthchecks - as this check is not affected by JWT verification.
* See ***YHCR FHIR Proxy Install 03 - Quickstart Install*** for more details of initial testing and troubleshooting

1. **Configure Autostart**

The FHIR Proxy needs to autostart when the server reboots.

* **On Linux**

1. Ensure the Docker Compose file is configured with ***restart: always*** for the containers which you want to autostart. (This is already the case in the provided docker compose files)
2. Set the service running in the background with docker-compose up -d
3. Nothing else is necessary on Linux – the docker daemon and containers will autostart when the server boots.
4. Test to make sure – eg reboot and check ***docker ps***

* **On Windows Server**

The steps above do not appear to be sufficient on Windows Server. It appears to be necessary to run a script on system startup.

1. Use Control Panel / Services to check that the Windows docker service itself is configured to autostart (by default it should be)
2. Create a start up script eg a .bat file containing:

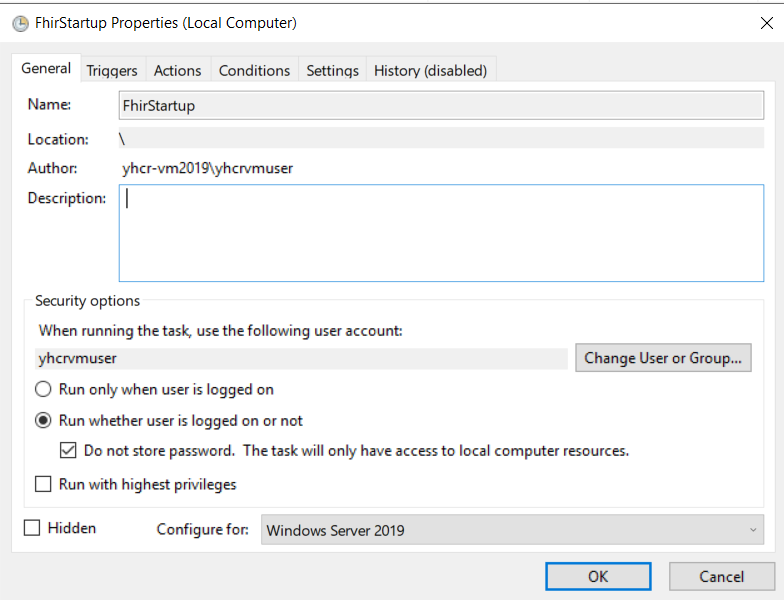
***cd*** *<the “prod” directory containing the prod docker-compose file>*

***docker-compose down***

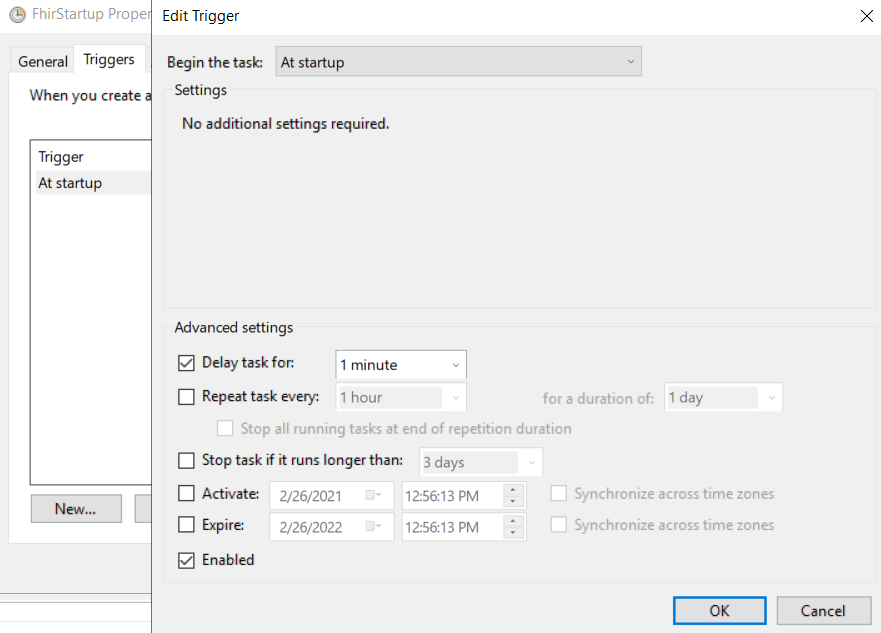
***docker-compose up -d***

(The ***docker-compose down*** line was added based on experience that sometimes debris can remain when the server is switched off. Startup was found to be more reliable with this precaution)

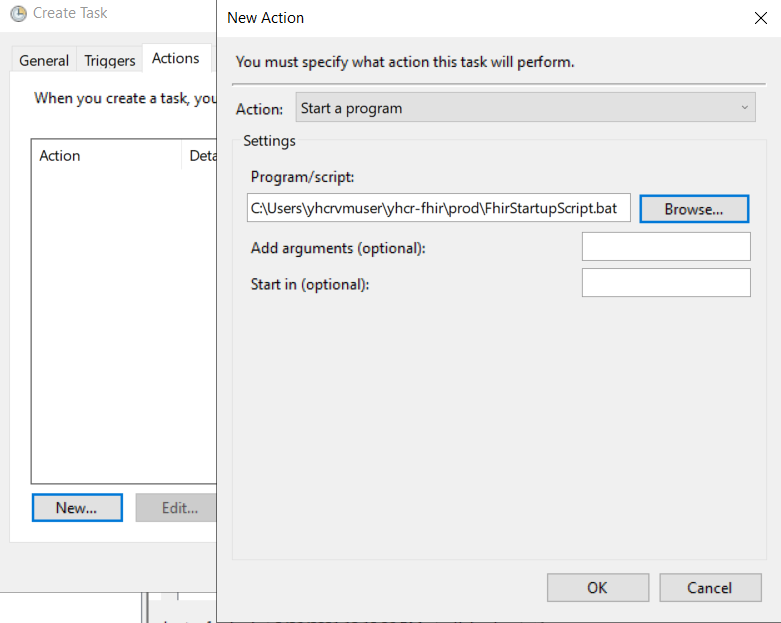
1. Test the script by running it manually to make sure that it works.
2. This script then needs to be scheduled on system startup – there are several ways to do this and you may have your own preferences. One method which the author tested successfully was using Windows Task Scheduler:
   * Create a new task
     + Needs to run whether a user is logged on or not (but does not seem to require a password)
     + Configure for the correct operating system



* + Trigger “At startup”
    - **The 1 minute delay appears to be very important**. If the task attempts to run too soon – eg before other services have started, then it will fail.



* + Action - to run your start-up script



1. Test this configuration – eg reboot and check ***docker ps*** to confirm the services are running

## Applying Updates

New versions of the software will be notified via the normal release process. You can also proactively check for updates. Updates are applied by pulling the latest docker image using the “build” flag of *docker-compose up* ie ***docker-compose up -d --no-deps --build <container name>***

For example:

1. Change your current directory to the location of the docker-compose file
2. ***docker-compose up -d --no-deps --build fhir-open***
3. ***docker-compose up -d --no-deps --build fhir-secured***

(The “no-deps” setting ensures that just the one specified container is rebuilt)

***Once you are satisfied with your installation from an internal perspective then it is time to move to the final step of connecting to the YHCR via the Onboarding process.***

# Technical Onboarding

## Introduction

Once installation is complete then the final step is to connect with the YHCR via the Onboarding process. The full onboarding process covers people, process, and technology and is described in more detail elsewhere. The purpose of this section is purely to give further advice on some of the technical aspects of getting connected.

The Onboarding Suite website is provided to assist with these technical steps. In essence the process involves registering for an account, and providing various information – including contact details, plus technical details such as IP addresses. You are then provided with details of the relevant YHCR endpoints, plus credentials to connect. This document is not intended to be a user guide for the Onboarding Suite itself – but rather to highlight the local technical configuration needed as part of using this tool.

* ***NB1: Test locally first*** – it is strongly advised to follow the steps suggested in previous sections and to test your installation locally first. This can help give confidence in the installation before introducing the additional complexity of configuring an external connection.
* ***NB2: Keep track of certificates*** – much of the technical onboarding process is about configuring certificates. This is not difficult per-se, but there are several different certificates and it is easy to get confused! Once you have downloaded all the certificates, it may be useful to take a moment to get organised and to be absolutely clear about which certificate is going where. (A diagram or checklist may help – examples are given below).
* ***NB3 – Onboarding is per environment*** – the details of IP addresses and Certificates are different for each of the YHCR Environments (Sandpit, Staging, Production). For the rest of this chapter we will consider connecting to a single environment - but it is important to remember that each task will apply three times.

## Firewall Configuration

You will provide your environment’s IP address to the Onboarding Suite – which will be used for routing and/or to open any necessary firewalls on YHCR side

The Onboarding Suite will provide details of the YHCR’s IP address and endpoints – which will similarly need to be used to open relevant firewalls for the YHCR to gain access into your organisation’s infrastructure.

## Certificates and Keys – Inbound from YHCR

The following are relevant for receiving inbound connections from YHCR:

|  |  |  |  |
| --- | --- | --- | --- |
| **Item** | **How generated** | **Purpose** | **Where it goes** |
| **YHCR SSL Root CA** | Provided by YHCR (generic) | 1. To verify the certificate chain of the YHCR-provided SSL certificate 2. To verify incoming TLS Mutual Authentication client keys | On the web proxy – likely to be part of the configuration of SSL and Mutual Authentication.  Also, on Windows, loaded into Trusted RootCA store |
| **SSL Key** | Self-generated (eg using OpenSSL, based on details in YHCR provided cnf file) | To establish SSL connection | On the web proxy, to configure SSL |
| **SSL Certificate** | Use Onboarding Suite to exchange Certificate Signing Request for this Certificate | To establish SSL connection (including proving server identity) | On the web proxy, to configure SSL  *(Note – may be combined with the previous two items into a pfx file)* |
| **YHCR IAM Root CA** | Provided by YHCR (generic) | To verify the JWT on incoming messages | Pasted into the .env configuration file for the FHIR Proxy |

The diagram below illustrates this further:



1. Enter organisation and connection details into YHCR Onboarding Suite
2. Download from Onboarding Suite a configuration (cnf) file containing these details
3. Use OpenSSL and the information in the cnf file to generate an SSL private key plus a certificate signing request (csr) for the SSL Certificate - containing the public key plus organisation details. Example command:

***openssl req -new -config <downloaded\_filename>.cnf -keyout <new\_filename>.key -out <new\_filename>.csr***

1. Upload the csr to YHCR Onboarding Suite, and download in return a fully signed SSL certificate
2. Download from the YHCR Onboarding Suite the YHCR’s SSL Root CA certificate (This does not need to be secret and is freely available)
3. Download from the YHCR Onboarding Suite the YHCR’s IAM Root CA certificate (This does not need to be secret and is freely available)

And then:

1. The SSL private key and signed SSL certificate need placing on the web server and configuring as the basis of enabling SSL

*In some cases (eg Microsoft IIS) then you may need to combine these two along with the Root CA certificate into a single pfx file. Example command:*

***openssl pkcs12 -export -out YHCRServerSSL.pfx -inkey YHCRServerSSL.key -in YHCRServerSSL.crt -certfile YHCRRootCA.crt name YHCRServerSSLCertificate***

1. The SSL Root CA certificate needs placing on the web server and configuring as the basis of enabling SSL Mutual Authentication (ie incoming client keys can be verified against it)
2. The IAM Root CA needs configuring into the FHIR Proxy where it will be used to verify incoming JWT tokens

Finally note that the cnf and csr files are transitory steps on the way to generating a certificate and are no longer needed once this process is complete.

## Certificates and Keys – Outbound to YHCR

The following are relevant for making outbound connections into YHCR:

|  |  |  |  |
| --- | --- | --- | --- |
| **Item** | **How generated** | **Purpose** | **Where it goes** |
| **Client Key** | Self-generated (eg using OpenSSL, using details in YHCR provided cnf file)  Use Onboarding Suite to send CSR containing public key to YHCR | To authenticate SSL connection into YHCR | Wherever the https connection to YHCR is opened (eg Client app) |
| **YHCR SSL Root CA** | Provided by YHCR (generic) | To verify SSL certificate from YHCR server | Wherever the https connection to YHCR is opened (eg Client app) |
| **Signing Key** | Self-generated (eg using OpenSSL, using details in YHCR provided cnf file)  Use Onboarding Suite to send CSR containing public key to YHCR | To sign tokens for sending to YHCR IAM | Wherever messages are created and tokens assigned (eg Client app) |
| **Client ID** | Provided by YHCR | To connect to YHCR IAM | Wherever messages are created and tokens assigned (eg Client app) |
| **Client Secret** | Self-generated, and entered into Onboarding Suite | To connect to YHCR IAM | Wherever messages are created and tokens assigned (eg Client app) |

The diagram below illustrates this further:



**To secure the SSL connection to YHCR:**

1. Similarly to the inbound process – download a cnf file containing details about your organisation, use this with OpenSSL to generate a Client key and csr, then upload the csr to YHCR
2. Download from the YHCR Onboarding Suite the YHCR’s SSL Root CA certificate (This does not need to be secret and is freely available)

And then:

1. Configure the Client key to be presented when attempting to establish an SSL connection to YHCR

*Notice that although it is possible to download the Client certificate file, this is primarily needed by the YHCR itself – to verify incoming connections which present your Client key*

1. Configure the SSL Root Certificate to use for checking the server identity of the YHCR server when establishing a connection to YHCR

**To generate JWT requests to YHCR IAM:**

1. Similarly to the inbound process once again – download a cnf file containing details about your organisation, use this with OpenSSL to generate a Signing key and csr, then upload the csr to YHCR
2. Get your Client ID, and enter a corresponding Client Secret into the Onboarding Suite

And then:

1. Go through the steps to establish an SSL connection to YHCR (including presenting your Client Key, as above
2. Present your Client ID and Client Secret as additional Basic Authentication credentials to IAM
3. Present to IAM a JWT token that is signed with your Signing Key

*Again, notice that although it is possible to download the Signing certificate file, this is primarily needed by the YHCR itself – to verify incoming JWT requests which present your Signing key*

1. IAM will check your request and, if valid, will return a corresponding JWT token signed by the YHCR. This can then be presented for further calls into YHCR.

One last diagram may be useful to further illustrate this process of gaining a token and making a request to YHCR. (See also <https://yhcr.org/wp-content/uploads/2020/11/YHCR_Design_Paper_005__Identity_and_Access_Management_v1.1.docx>)



1. At the time of writing then two instances of the FHIR Proxy are necessary to achieve these two endpoints. It is hoped that a future version of the FHIR Proxy might offer this within one instance. [↑](#footnote-ref-2)