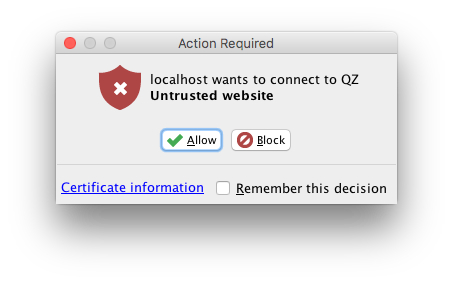
# The ARK QZ Tray setup (Optional)

This document describes the process to set up barcode printing in The ARK, utilizing [QZ Tray](https://qz.io/download/). Each instance of The ARK should have its own Certificate Authority and accompanying QZ Tray installer, with the appropriate bundled root CA certificate.

If the following steps have not been completed, barcode printing will continue to function, however each time The ARK attempts to connect to QZ Tray a popup similar to the following will be shown:



The “Remember this decision” checkbox does not work for untrusted websites.

Easy Instructions

Use the script located at #find somewhere to put it in git# and follow the prompts.

## Manual Instructions

### Pre-WAR compilation

Before compiling and deploying The ARK war file, you’ll need to generate the certificates used to authenticate QZ Tray with your instance of The ARK. This requires OpenSSL.

* Create a directory to store all the keys and certificates used in this process. For this example, we will use /root/ca.
* Populate /root/ca with the required base contents:

**cd** /root/ca

**mkdir** certs crl newcerts private

**chmod** 700 private

**touch** index.txt

**echo** 1000 **>** serial

* Copy the root OpenSSL configuration file from <URL OF FILE HERE> to /root/ca/openssl.cnf, and replace the directory /root/ca with the directory you created earlier.
* Now you can create the root key, and enter a pass phrase for it:

**cd** /root/ca

**openssl** genrsa –aes256 –out private/ca.key.pem 4096

**chmod** 400 private/ca.key.pem

* Using the root key, we now create a root certificate. We will be giving the root certificate a long expiry date, as when it expired all certificates signed by the CA become invalid. You’ll need to provide the private key pass phrase created earlier, and you’ll be asked to specify information that will be incorporated into the certificate:

**cd** /root/ca

**openssl** req –config openssl.cnf \

-key private/ca.key.pem \

-new –x509 –days 3650 –sha256 –extensions v3\_ca \

-out certs/ca.cert.pem

**chmod** 444 certs/ca.cert.pem

* Now we are ready to create the intermediate pair. We’ll first create the directory to store the intermediate pair, and the same setup that we created before:

**mkdir** /root/ca/intermediate

**cd** /root/ca/intermediate

**mkdir** certs crl csr newcerts private

**chmod** 700 private

**touch** index.txt

**echo** 1000 > serial

**echo** 1000 > crlnumber

* We’ll also copy the root OpenSSL configuration file to the intermediate directory, and update the settings shown below:

**cp** /root/ca/openssl.cnf /root/ca/intermediate/openssl.cnf

**[** CA\_default **]**

dir **=** /root/ca/intermediate

private\_key **=** $dir/private/intermediate.key.pem

certificate **=** $dir/certs/intermediate.cert.pem

crl **=** $dir/crl/intermediate.crl.pem

policy **=** policy\_loose

* Once the intermediate OpenSSL configuration has been updated correctly, we can create the intermediate key. You’ll be asked to specify another pass phrase for the intermediate key, this should be different to the root pass phrase.

**cd** /root/ca

**openssl** genrsa –aes256 \

-out intermediate/private/intermediate.key.pem 2048

**chmod** 400 intermediate/private/intermediate.key.pem

* Create a certificate signing request (CSR) using the intermediate key. You’ll be asked to specify information again, and these fields should be the same as the root certificate however the Common Name should be the URL of your ARK instance:

**cd** /root/ca

**openssl** req –config intermediate/openssl.cnf –new –sha256 \

-key intermediate/private/intermediate.key.pem \

-out intermediate/csr/intermediate.csr.pem

* With the CSR created, we now create the intermediate CA. This should be valid for an equal or shorter time than the root CA:

**cd** /root/ca

**openssl** ca –config openssl.cnf –extensions v3\_intermediate\_ca \

-days 3650 –notext –md sha256 \

-in intermediate/csr/intermediate.csr.pem \

-out intermediate/certs/intermediate.cert.pem

**chmod** 444 intermediate/certs/intermediate.cert.pem

* Now we have all three of the required components; intermediate.cert.pem, intermediate.key.pem and ca.cert.pem, we can place them in the correct locations. intermediate.cert.pem and intermediate.key.pem go to ark-container/src/main/resources/ as digital-certificate.txt and private-key.pem

**cd** /root/ca

**cp** intermediate/certs/intermediate.cert.pem \

ark-container/src/main/resources/digital-certificate.txt

**cp** intermediate/certs/intermediate.key.pem \

ark-container/src/main/resources/private-key.pem

* We also need to add to ark-container/src/main/resources/application.properties such that the barcode.privatekey.password property has the correct pass phrase for the intermediate private key.
* With the intermediate key and certificates in place, and the application.propeties file updated, you are ready to compile and deploy The ARK.

### Compiling custom QZ Tray

To compile QZ Tray, you need the following requirements:

* Git
* The root CA certificate from the previous step
* Ant

And depending on which platform you intend to support, at least one of the following:

* NSIS (Windows)
* makeself (Linux)
* Obtain the QZ Tray source code from git or from a release download.
  + Clone the QZ Tray git repository: <https://github.com/qzind/tray>

**git** clone https://github.com/qzind/tray

* + Download from GitHub releases (update for appropriate version number)

**wget** https://github.com/qzind/tray/archive/v2.0.1.zip

**unzip** v2.0.1.zip

**mv** tray-2.0.1 tray

* Then compile the source, ensuring you specify the –Dauthcert.use flag to package our root CA certificate inside the installer. The different ant commands listed below will compile the installer for each platform.

**cd** tray

**ant** nsis –Dauthcert.use=/root/ca/certs/ca.cert.pem #Windows

**ant** pkgbuild –Dauthcert.use=/root/ca/certs/ca.cert.pem #macOS

**ant** makeself –Dauthcert.use=/root/ca/certs/ca.cert.pem #Linux

* Compiled installers are available at tray/out/