Certificates

Introduction

(TODO: do these test servers still work? is wave-protocol still the right place for this?

The instructions below are for self-signed certificates, which the current test server, `initech-corp.com`, will accept, which allows for easy testing of the federation protocol. The `acmewave.com` test server has been transitioned to only accept CA-issued certificates. CA-issued certificates are better as they involve a trusted third-party, and it is expected that in production a Wave server would only accept CA-issued certificates. Changes to the test servers that affect which kind of certificate they accept will be announced on the wave-protocol mailing list.

If you want to go through the steps to generate a CA-issued certificate the instructions are at the end of this page. A server that is set up to accept self-signed certificates will also accept CA-issued certificates, so you will still be able to interop with both test servers with a CA-issued certificate.

Note that real certs will contain a critical extension that only Wave servers should accept, to prevent them being re-used as SSL server certificates.

Self-Signed Certificates

There is a script called `make_cert.sh` for generating certificates in the root directory of the repository. When you run it, you'll see roughly this:

```
$ ./make_cert.sh test
1) Generating key for test in 'test.key' ...

Generating RSA private key, 2048 bit long modulus
..............................+++
.....................................................+++
e is 65537 (0x10001)

2) Generating certificate request for test in 'test.crt' ...

You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:
State or Province Name (full name) [Some-State]:
Locality Name (eg, city) []:
Organization Name (eg, company) [Internet Widgits Pty Ltd]:
Organizational Unit Name (eg, section) []:
Common Name (eg, YOUR name) []:
Email Address []:
```

You can answer whatever you want to all questions except the Common Name question. There you should answer the DNS name of your server.

The result of this would be two files, `test.crt` and `test.key`. The certificate you can give to anyone, especially those who want to check its a known good cert. The key is your private key and should not be revealed.

The FedOne code does not support password protected private keys. This is not a concern if you used the script supplied above as the generated private key will not be password protected.

(TODO: still fedone?)

View the certificate

```
$ openssl x509 -text -in test.crt
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number:
            e1:e7:23:24:cc:5e:71:d1
        Signature Algorithm: sha1WithRSAEncryption
        Issuer: C=AU, ST=Some-State, O=Internet Widgits Pty Ltd, CN=www.links.org
        Validity
            Not Before: Jul 17 20:59:30 2009 GMT
            Not After : Jul 17 20:59:30 2010 GMT
        Subject: C=AU, ST=Some-State, O=Internet Widgits Pty Ltd, CN=www.links.org
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
            RSA Public Key: (2048 bit)
                Modulus (2048 bit):
                    00:d9:0c:57:6b:fa:ad:b2:8f:b1:17:08:1f:d4:b1:
                    10:5a:eb:7c:35:01:02:73:3f:67:68:5d:fd:3e:4c:
                    ec:29:fa:3c:76:09:88:f5:fd:e2:ec:ad:47:44:d9:
                    6a:a9:4f:b6:2e:42:17:f3:11:b2:59:fd:2e:ab:69:
                    c6:95:a5:e2:2f:15:16:43:5f:1f:b5:c0:38:35:f0:
                    a3:db:30:19:6b:a9:b1:10:4f:e7:80:a2:a5:68:c5:
                    b5:3e:1c:81:ce:7c:98:b0:bb:8e:5b:d0:f3:21:25:
                    f7:b5:eb:d0:bf:72:f5:69:bc:24:ab:69:38:db:f5:
                    85:c9:92:98:e7:e0:a6:30:57
                Exponent: 65537 (0x10001)
        X509v3 extensions:
            X509v3 Subject Key Identifier:
                C1:B6:25:4F:F7:59:52:9C:8D:87:B9:7F:88:EC:2C:1D:3B:0F:DC:0F
            X509v3 Authority Key Identifier:
                keyid:C1:B6:25:4F:F7:59:52:9C:8D:87:B9:7F:88:EC:2C:1D:3B:0F:DC:0F
                DirName:/C=AU/ST=Some-State/O=Internet Widgits Pty Ltd/CN=www.links.org
                serial:E1:E7:23:24:CC:5E:71:D1

            X509v3 Basic Constraints:
                CA:TRUE
    Signature Algorithm: sha1WithRSAEncryption
        6d:0c:b9:a1:1e:37:9f:53:d9:bf:a1:10:21:04:46:84:27:57:
        cd:91:2a:3d:11:38:51:3e:80:ac:e0:10:d9:37:f3:27:00:20:
        04:88:2f:de:2a:54:6f:e2:f1:a5:1b:d7:54:04:4c:02:ef:6a:
        60:76:d6:68:6a:42:02:c8:ac:0f:df:16:fa:e8:b6:a6:19:8b:
        46:26:1f:bb:d6:69:6f:15:5a:43:89:ce:41:df:8b:58:74:9d:
        66:13:d9:e5:b6:9e:84:0e:fe:63:2a:d6:5c:6c:96:e7:ae:ae:
        6a:a2:a9:2e:81:98:87:2d:ce:3c:48:7c:d4:2b:71:98:97:1d:
        78:d0
-----BEGIN CERTIFICATE-----
MIIC+zCCAmSgAwIBAgIJAOHnIyTMXnHRMA0GCSqGSIb3DQEBBQUAMF0xCzAJBgNV
BAYTAkFVMRMwEQYDVQQIEwpTb21lLVN0YXRlMSEwHwYDVQQKExhJbnRlcm5ldCBX
aWRnaXRzIFB0eSBMdGQxFjAUBgNVBAMTDXd3dy5saW5rcy5vcmcwHhcNMDkwNzE3
MjA1OTMwWhcNMTAwNzE3MjA1OTMwWjBdMQswCQYDVQQGEwJBVTETMBEGA1UECBMK
U29tZS1TdGF0ZTEhMB8GA1UEChMYSW50ZXJuZXQgV2lkZ2l0cyBQdHkgTHRkMRYw
FAYDVQQDEw13d3cubGlua3Mub3JnMIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKB
gQDZDFdr+q2yj7EXCB/UsRBa63w1AQJzP2doXf0+TOwp+jx2CYj1/eLsrUdE2Wqp
T7YuQhfzEbJZ/S6racaVpeIvFRZDXx+1wDg18KPbMBlrqbEQT+eAoqVoxbU+HIHO
fJiwu45b0PMhJfe169C/cvVpvCSraTjb9YXJkpjn4KYwVwIDAQABo4HCMIG/MB0G
A1UdDgQWBBTBtiVP91lSnI2HuX+I7CwdOw/cDzCBjwYDVR0jBIGHMIGEgBTBtiVP
91lSnI2HuX+I7CwdOw/cD6FhpF8wXTELMAkGA1UEBhMCQVUxEzARBgNVBAgTClNv
bWUtU3RhdGUxITAfBgNVBAoTGEludGVybmV0IFdpZGdpdHMgUHR5IEx0ZDEWMBQG
A1UEAxMNd3d3Lmxpbmtzlm9yZ4IJAOHnIyTMXnHRMAwGA1UdEwQFMAMBAf8wDQYJ
KoZIhvcNAQEFBQADgYEAbQy5oR43n1PZv6EQIQRGhCdXzZEqPRE4UT6ArOAQ2Tfz
JwAgBIgv3ipUb+LxpRvXVARMAu9qYHbWaGpCAsisD98W+ui2phmLRiYfu9ZpbxVa
Q4nOQd+LWHSdZhPZ5baehA7+YyrWXGyW566uaqKpLoGYhy3OPEh81CtxmJcdeNA=
-----END CERTIFICATE-----
```

Check certificate and key match

```
$ openssl x509 -modulus -in test.crt -noout
Modulus=AC12A9EDA81134852DE9887BD0B4B36940B48F2520BF6970DE8854FAF4A476EAF32711C36E65DAB96729FABDDCA4531ABC3AEAD1
DD3BC0E58429CE434B070617D9065A6B7B3EBC76DE7DFBD9150DF0D27D6F5E6D6F11C7D0A4CFDCB6763BC1C01208AF184A28BC2628F195BD
75B96EB2C58F94D5EC74F7A301F2D8EB6936858B
$ openssl rsa -in test.key -modulus -noout
Modulus=AC12A9EDA81134852DE9887BD0B4B36940B48F2520BF6970DE8854FAF4A476EAF32711C36E65DAB96729FABDDCA4531ABC3AEAD1
DD3BC0E58429CE434B070617D9065A6B7B3EBC76DE7DFBD9150DF0D27D6F5E6D6F11C7D0A4CFDCB6763BC1C01208AF184A28BC2628F195BD
75B96EB2C58F94D5EC74F7A301F2D8EB6936858B
```

The two outputs should match.

Now you are done and can add the key and cert to you Wave server and interop with other Wave servers that accept self-signed certificates. If you want to generate a CA-issued Certificate follow the directions in the next section.

Getting a CA-issued Certificate

You will need access to the email account `webmaster@yourdomain.com`, `postmaster@yourdomain.com`, or `hostmaster@yourdomain.com`.

First generate an encrypted private key:

```
$ openssl genrsa -des3 -out example.com.encrypted.key 2048
```

You will be asked for passphrase, make sure it is at least 10 characters.

Then generate a certificate request:

```
$ openssl req -new -key example.com.encrypted.key -out example.com.csr
```

You will be asked for passphrase from above. After that you will be asked to fill in a bunch of details.

> The *Common Name* should be in the form `wave.example.com`

You can use this certificate request with your Certificate Authority of choice. Below are instructions on getting a free Class 1 CA-issued certificate from StartSSL. Using StartSSL is not required, but is documented here because it is one of the CAs that provide free CA certs.

StartSSL instructions

1. Go to `https://www.startssl.com`.
2. Sign in, or sign up. To sign up you will need to provide email that you can validate, then log out and log in again - click in Authenticate - you will be asked for (email) certificate that was generated in the sign up process.
3. Go to *Control Panel*.
4. Click on the *Validations Wizard*.
5. Choose *Domain Name Validation* where you have to validate you domain, i.e. `example com`.
6. Go to *Certificates Wizard*.
7. Choose XMPP certificate.
8. In the private key generation step click on "*skip*".
9. In the next step paste the certificate request that was generated earlier (the contents of the `example.com.csr`).
10. Choose your domain, i.e. `example.com`. In the subdomain you need to enter "wave", i.e. `http://wave.example.com`.
11. Click on *Continue* until *Finish*.
12. After that you will have your signed certificate. Save it as **`example.com.crt`**.
13. Go to *ToolBox*, choose *StartCom CA Certificates*. Download your intermediate certificate **`sub.class1.server.ca.pem`** and the Certification Authority certificate **`ca.pem`**.

So by now you should have 5 files:

- `example.com.encrypted.key`
- `example.com.crt`
- `example.com.csr`
- `sub.class1.server.ca.pem`
- `ca.pem`

Make sure to backup the private key and signed certificate (`example.com.encrypted.key example.com.crt`) and put it somewhere in a safe place.

Now let's remove the passphrase from the private key with:

```
$ openssl rsa -in example.com.encrypted.key -out example.com.nonencrypted.key
```

then convert the key to a different format with:

```
$ openssl pkcs8 -topk8 -nocrypt -in example.com.nonencrypted.key -out example.com.key
```

Now we have the private key we can use with waveinabox server and a certificate signed by StartCom.
You can test your certificate using the `openssl` command line tool. If you get a CA-issued cert for the domain `example.com` then you can test the cert with:

```
$ openssl verify -CAfile sub.class1.server.ca.pem example.com.crt
example.com.crt: OK
```

⚠ (TODO: Make this refer to server.federation.config instead)

To enable the certs you will need to make some changes to `run-config.sh`. Enable certs, and add the intermediate cert to the list of certificates:

```
# Set true to disable the verification of signed deltas
WAVESERVER_DISABLE_VERIFICATION=false

# Set true to disable the verification of signers (certificates)
WAVESERVER_DISABLE_SIGNER_VERIFICATION=false

CERTIFICATE_FILENAME_LIST=${WAVE_SERVER_DOMAIN_NAME}.crt,sub.class1.server.ca.pem
```

Note: Some people have found that they need to include both the `sub.class1.server.pem` and the `ca-bundle` cert in the chain as follows:

```
CERTIFICATE_FILENAME_LIST=$\{WAVE_SERVER_DOMAIN_NAME}.crt,sub.class1.server.ca.pem,ca.pem
```

The order of the certificates listed in the `CERTIFICATE_FILENAME_LIST` is important, with your certificate going first, and intermediate certs following.

Check Certificates

⚠ (TODO: change FedOne to WIAB?)

The `check-certificates.sh` script included in the FedOne? source will do all of the above checks for you. Make sure `run-config.sh` is configured first then run `check-certificates.sh`. If the certificates are valid and configured correctly you will see:

```
SUCCESS: The certificates have been verified and are working correctly
```

Otherwise and error message will be printed pointing to the cause of the error.