Runtime Infrastructure - Feature #3240

setup and use of real SSL certificates (NOT self-signed)

02/06/2017 11:10 AM - Greg Shah

Status:	Closed	**Start date:**	
Priority:	Normal	**Due date:**	
Assignee:	Greg Shah	**% Done:**	100%
Category:		**Estimated time:**	0.00 hour
Target version:			
billable:	No	**vendor_id:**	GCD

Description

Related issues:

Related to Runtime Infrastructure - Bug #5663: allow webcertificates details ...	**New**

History

#1 - 02/06/2017 01:28 PM - Greg Shah

In a recent customer POC using the web client, I was implementing certificates from Let's Encrypt so that the resulting runtime installation could be used on the Internet and without any of the issues that come form self-signed certificates.

Let's Encrypt (www.letsencrypt.org) makes it pretty easy to obtain certificates, so long as you can prove that you control the web site that is associated with the domain name(s) associated with the certificate being requested. The certificates are provided at no charge and you can get them immediately using automated tools. I used the "certbot" tool (https://certbot.eff.org/) to request the certificate. The same tool will be used to renew the cert in approximately 3 months when it expires. Since the certificates are requested via an automated process, Let's Encrypt limits the period of validity (to 3 months) and makes it easy (but also required) to renew them.

The following script will request the certificate. In this example, the system is behind a firewall that exposes the web application using port 8443. The application must be off to use this method, since the certbot tool will listen on the port and respond to the Let's Encrypt authority to prove that the domain is under the control of the requester. Due to a restriction in the Let's Encrypt request process, the certbot MUST listen on port 443. This means that any intervening firewall that does port mapping, must pass port 443. In my case, I mapped the external IP address with port 443 to the internal system on port 8443. For that reason, when requesting the cert, I used redir to temporarily redirect traffic on the internal 8443 port to the certbot listening on the internal system port 443.

```
#!/bin/bash
# run this with sudo

# redirect that port to 443 because the letsencrypt client is hardcoded to listen on 443
redir --lport 8443 --cport 443 &

# store the background process' pid
bg_pid=$!

# renew the cert
letsencrypt certonly --standalone --standalone-supported-challenges tls-sni-01 -d demo.goldencode.com

# stop redir
kill -9 "$bg_pid"
```

The -d demo.goldencode.com is the part that needs to be customized to the domain name being used. Please note that you can add multiple domain names to this option like -d my.first.domain -d my.second.domain -d my.third.domain. When you do this, the resulting certificate will be valid for all of the domains. Of course, each of the domains must be configured in the public DNS to all report the same IP address. The IP address is not encoded in the certificate, just the domains.

In Ubuntu (16.04 and later), you can install the certbot code using sudo apt-get install letsencrypt. This is needed before you run the script above.

When running the tool for the first time, you will have to provide an email address associated with the certificates. That address will be used for expiration notifications at a minimum (if not other communications).

If the tool fails, it is most likely due to firewall/connectivity or DNS issues when accessed via the Internet. Read the error message and try to adjust as needed.

If the tool succeeds, then you will find the key and certs as follows:

```
/etc/letsencrypt/live/demo.goldencode.com/chain.pem
/etc/letsencrypt/live/demo.goldencode.com/fullchain.pem
/etc/letsencrypt/live/demo.goldencode.com/privkey.pem
/etc/letsencrypt/live/demo.goldencode.com/cert.pem
```

You need to carefully secure and backup all contents of /etc/letsencrypt/.

You cannot directly use the .pem files in FWD. I will post next to explain how to use these results with FWD.

#2 - 02/06/2017 02:59 PM - Greg Shah

In the directory, I added the root CA certificate, the FWD server's public certificate and the FWD server's private key. I also had to explicitly set the hostname.

1. Root CA certificate

FWD dynamically creates an in-memory trust-store. I think that at this time need to include the root CA (but I have not confirmed this). Even if it is needed right now, I haven't confirmed if this is only temporarily needed because the JVM doesn't have the Let's Encrypt root cert in the cacerts file OR if this is a permanent restriction until we properly honor the JVM cacerts.

Look in the chain.pem file and you will find something like this:

```
-----BEGIN CERTIFICATE-----
MIIEkjCCA3qgAwIBAgIQCgFBQgAAAVOFc2oLheynCDANBgkqhkiG9w0BAQsFADA/
MSQwIgYDVQQKExtEaWdpdGFsIFNpZ25hdHVyZSBUcnVzdCBDby4xFzAVBgNVBAMT
DkRTVCBSb290IENBIFgzMB4XDTE2MDMxNzE2NDA0NloXDTIxMDMxNzE2NDA0Nlow
SjELMAkGA1UEBhMCVVMxFjAUBgNVBAoTDUxldCdzIEVuY3J5cHQxIzAhBgNVBAMT
GkxldCdzIEVuY3J5cHQgQXV0aG9yaXR5IFgzMIIBIjANBgkqhkiG9w0BAQEFAAOC
AQ8AMIIBCgKCAQEAnNMM8FrlLke3cl03g7NoYzDq1zUmGSXhvb418XCSL7e4S0EF
q6meNQhY7LEqxGiHC6PjdeTm86dicbp5gWAf15Gan/PQeGdxyGkOlZHP/uaZ6WA8
SMx+yk13EiSdRxta67nsHjcAHJyse6cF6s5K671B5TaYucv9bTyWaN8jKkKQDIZ0
Z8h/pZq4UmEUEz9l6YKHy9v6Dlb2honzhT+Xhq+w3Brvaw2VFn3EK6BlspkENnWA
a6xK8xuQSXgvopZPKiAlKQTGdMDQMc2PMTiVFrqoM7hD8bEfwzB/onkxEz0tNvjj
/PIzark5McWvxI0NHWQWM6r6hCm21AvA2H3DkwIDAQABo4IBfTCCAXkwEgYDVR0T
AQH/BAgwBgEB/wIBADAOBgNVHQ8BAf8EBAMCAYYwfwYIKwYBBQUHAQEEczBxMDIG
CCsGAQUFBzABhiZodHRwOi8vaXNyZy50cnVzdGlkLm9jc3AuaWRlbnRydXN0LmNv
bTA7BggrBgEFBQcwAoYvaHR0cDovL2FwcHMuaWRlbnRydXN0LmNvbS9yb290cy9k
c3Ryb290Y2F4My5wN2MwHwYDVR0jBBgwFoAUxKexpHsscfrb4UuQdf/EFWCFiRAw
VAYDVR0gBE0wSzAIBgZngQwBAgEwPwYLKwYBBAGC3xMBAQEwMDAuBggrBgEFBQcC
ARYiaHR0cDovL2Nwcy5yb290LXgxLmxldHNlbmNyeXB0Lm9yZzA8BgNVHR8ENTAz
MDGgL6AthitodHRwOi8vY3JsLmlkZW50cnVzdC5jb20vRFNUUk9PVENBWDNDUkwu
Y3JsMB0GA1UdDgQWBBSoSmpjBH3duubRObemRWXv86jsoTANBgkqhkiG9w0BAQsF
AAOCAQEA3TPXEfNjWDjdGBX7CVW+dla5cEilaUcne8IkCJLxWh9KEik3JHRRHGJo
uM2VcGfl96S8TihRzZvoroed6ti6WqEBmtzw3Wodatg+VyOeph4EYpr/1wXKtx8/
wApIvJSwtmVi4MFU5aMqrSDE6ea73Mj2tcMyo5jMd6jmeWUHK8so/joWUoHOUgwu
X4Po1QYz+3dszkDqMp4fklxBwXRsW10KXzPMTZ+sOPAveyxindmjkW8lGy+QsRlG
PfZ+G6Z6h7mjem0Y+iWlkYcV4PIWL1iwBi8saCbGS5jN2p8M+X+Q7UNKEkROb3N6
KOqkqm57TH2H3eDJAkSnh6/DNFu0Qg==
-----END CERTIFICATE-----
```

Remove the header and footer and make the entry a single line with no spaces. Then add the /security/certificates/cas/letsencrypt-root node and make it look like this:

```
          <node class="bytes" name="letsencrypt-root">
            <node-attribute name="value" value="MIIEkjCCA3qgAwIBAgIQCgFBQgAAAVOFc2oLheynCDANBgkqhkiG9w0BAQsFAD
A/MSQwIgYDVQQKExtEaWdpdGFsIFNpZ25hdHVyZSBUcnVzdCBDby4xFzAVBgNVBAMTDkRTVCBSb290IENBIFgzMB4XDTE2MDMxNzE2NDA0NloX
DTIxMDMxNzE2NDA0NlowSjELMAkGA1UEBhMCVVMxFjAUBgNVBAoTDUxldCdzIEVuY3J5cHQxIzAhBgNVBAMTGkxldCdzIEVuY3J5cHQgQXV0aG
9yaXR5IFgzMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAnNMM8FrlLke3cl03g7NoYzDq1zUmGSXhvb418XCSL7e4S0EFq6meNQhY
7LEqxGiHC6PjdeTm86dicbp5gWAf15Gan/PQeGdxyGkOlZHP/uaZ6WA8SMx+yk13EiSdRxta67nsHjcAHJyse6cF6s5K671B5TaYucv9bTyWaN
```

```
8jKkKQDIZ0Z8h/pZq4UmEUEz9l6YKHy9v6Dlb2honzhT+Xhq+w3Brvaw2VFn3EK6BlspkENnWAa6xK8xuQSXgvopZPKiAlKQTGdMDQMc2PMTiV
FrqoM7hD8bEfwzB/onkxEz0tNvjj/PIzark5McWvxI0NHWQWM6r6hCm21AvA2H3DkwIDAQABo4IBfTCCAXkwEgYDVR0TAQH/BAgwBgEB/wIBAD
AOBgNVHQ8BAf8EBAMCAYYwfwYIKwYBBQUHAQEEczBxMDIGCCsGAQUFBzABhiZodHRwOi8vaXNyZy50cnVzdGlkLm9jc3AuaWRlbnRydXN0LmNv
bTA7BggrBgEFBQcwAoYvaHR0cDovL2FwcHMuaWRlbnRydXN0LmNvbS9yb290cy9kc3Ryb290Y2F4My5wN2MwHwYDVR0jBBgwFoAUxKexpHsscf
rb4UuQdf/EFWCFiRAwVAYDVR0gBE0wSzAIBgZngQwBAgEwPwYLKwYBBAGC3xMBAQEwMDAuBggrBgEFBQcCARYiaHR0cDovL2Nwcy5yb290LXgx
LmxldHNlbmNyeXB0Lm9yZzA8BgNVHR8ENTAzMDGgL6AthitodHRwOi8vY3JsLmlkZW50cnVzdC5jb20vRFNUUk9PVENBWDNDUkwuY3JsMB0GA1
UdDgQWBBSoSmpjBH3duubRObemRWXv86jsoTANBgkqhkiG9w0BAQsFAAOCAQEA3TPXEfNjWDjdGBX7CVW+dla5cEilaUcne8IkCJLxWh9KEik3
JHRRHGJouM2VcGfl96S8TihRzZvoroed6ti6WqEBmtzw3Wodatg+VyOeph4EYpr/1wXKtx8/wApIvJSwtmVi4MFU5aMqrSDE6ea73Mj2tcMyo5
jMd6jmeWUHK8so/joWUoHOUgwuX4Po1QYz+3dszkDqMp4fklxBwXRsW10KXzPMTZ+sOPAveyxindmjkW8lGy+QsRlGPfZ+G6Z6h7mjem0Y+iWl
kYcV4PIWL1iwBi8saCbGS5jN2p8M+X+Q7UNKEkROb3N6KOqkqm57TH2H3eDJAkSnh6/DNFu0Qg=="/>
          </node>
```

2. The FWD server's public certificate.

Look in the cert.pem and edit it the same way that was done for the root CA cert (remove the header and footer and make the entry a single line with no spaces). Then edit the FWD server's public certificate in the /security/certificates/peers/standard node. Paste it as the value:

```
          <node class="bytes" name="standard">
            <node-attribute name="value" value="MIIFCjCCA/KgAwIBAgISA8jQPH9cfzXI/qwRiPG/9m4tMA0GCSqGSIb3DQEBCw
UAMEoxCzAJBgNVBAYTAlVTMRYwFAYDVQQKEw1MZXQncyBFbmNyeXB0MSMwIQYDVQQDExpMZXQncyBFbmNyeXB0IEF1dGhvcml0eSBYMzAeFw0x
NzAyMDMxNDE2MDBaFw0xNzA1MDQxNDE2MDBaMB4xHDAaBgNVBAMTE2RlbW8uZ29sZGVuY29kZS5jb20wggEiMA0GCSqGSIb3DQEBAQUAA4IBDw
AwggEKAoIBAQC21tPcMhOjAv1IQWRiw0uQrHGwacUnDQ6albUNULxLbE1R9aejzSZ6BnHnyv/mpGrxZYGj4m4iSxIu8oGS9Eg1dRGbf5gsuZwl
IJ/YL6flUl5OUq281YFIgX6D/TcfAtIgGHUeTrsHNtl16wcl0ihNRSRen0GevG0DOVrMl7MJPrPYqyRQbnD3ViYGZhlz533Tr/DalftYkek39j
hkPbBAZTp79RAbYj0L8K6YsKMRGFPho8yal3zGHNQ+AjoyGCxzNbJahrg3iXFNKVU/efk1LOjelKqkO9NJ1EARI04e7nNo5lU7VjjX85n+mOK2
l3xWw7QYGdVlv8pYtuWQPwNJAgMBAAGjggIUMIICEDAOBgNVHQ8BAf8EBAMCBaAwHQYDVR0lBBYwFAYIKwYBBQUHAwEGCCsGAQUFBwMCMAwGA1
UdEwEB/wQCMAAwHQYDVR0OBBYEFPtHecZ2H3ap5riW//u3aknjtJTHMB8GA1UdIwQYMBaAFKhKamMEfd265tE5t6ZFZe/zqOyhMHAGCCsGAQUF
BwEBBGQwYjAvBggrBgEFBQcwAYYjaHR0cDovL29jc3AuaW50LXgzLmxldHNlbmNyeXB0Lm9yZy8wLwYIKwYBBQUHMAKGI2h0dHA6Ly9jZXJ0Lm
ludC14My5sZXRzZW5jcnlwdC5vcmcvMB4GA1UdEQQXMBWCE2RlbW8uZ29sZGVuY29kZS5jb20wgf4GA1UdIASB9jCB8zAIBgZngQwBAgEwgeYG
CysGAQQBgt8TAQEBMIHWMCYGCCsGAQUFBwIBFhpodHRwOi8vY3BzLmxldHNlbmNyeXB0Lm9yZzCBqwYIKwYBBQUHAgIwgZ4MgZtUaGlzIENlcn
RpZmljYXRlIG1heSBvbmx5IGJlIHJlbGllZCB1cG9uIGJ5IFJlbHlpbmcgUGFydGllcyBhbmQgb25seSBpbiBhY2NvcmRhbmNlIHdpdGggdGhl
IENlcnRpZmljYXRlIFBvbGljeSBmb3VuZCBhdCBodHRwczovL2xldHNlbmNyeXB0Lm9yZy9yZXBvc2l0b3J5LzANBgkqhkiG9w0BAQsFAAOCAQ
EAKkOzoqQNubu/FDMD1H9tlGQhAVERa5CN687JzYxLgXI6UGe6xsgylYi3b8YxCe4z01zBkvOdecm429WXckO7kU0rN8820Wxx2gsGb17jlQvU
q2SBim9QKIt8tREBdKx1jDDwEam2iK2kS7rup4kYsZxmgJfFCvQuQvd0L7lYPZQRkxasMPB3+rSwe/zSSsU3MZJkoawfcWXsK1jA9KmoyHczGI
3ZG1/y+ne0NW594qP6SDegAuVHTcwwkq2imFEE1jZEs2mt6rx1EDU1bq7E9GJpOiANMsiLJI/qSF5WwxPFrolXfRZJAEvzFFFJgI+6r+0Wqzx/
oVJQYzji3a8Cmw=="/>
          </node>
```

3. The FWD server's private key.

The private key is stored inside the privkey.pem file. It is in the same format as the certificates and it is NOT encrypted. This value must be encrypted for use in the directory. Remove the header and footer and make the entry a single line with no spaces as was done with the certificates above.

Then run this command:

java -classpath build/lib/p2j.jar com.goldencode.p2j.security.EncryptBase64Value <unencrypted_private_key> <32_character_password>

Yes, the password MUST be exactly 32 characters in size. The tool (which is implemented in branch 3209e revision 11198) will emit 2 lines of output:

```
BASE64 encoded encrypted value: <encrypted_private_key>
BASE64 encoded password: <encoded_password>
```

Take these values and edit the /security/certificates/private-keys/standard section and insert these as follows:

```
          <node class="container" name="standard">
            <node class="bytes" name="key-entry">
              <node-attribute name="value" value="ENCRYPTED_PRIVATE_KEY_GOES_HERE"/>
            </node>
            <node class="bytes" name="key-password">
              <node-attribute name="value" value="ENCODED_PASSWORD_GOES_HERE"/>
            </node>
          </node>
```

4. FWD server's web hostname.

Even if you have the FWD server running on an internal system (behind a firewall), as long as it is accessed via the Internet, it must know the proper public hostname on which it will be accessed. Edit the /server/default/webClient/host/ node like this (inserting the correct host name for your system):

```
          <node class="string" name="host">
            <node-attribute name="value" value="demo.goldencode.com"/>
          </node>
```

#3 - 02/06/2017 04:14 PM - Greg Shah

In addition to the directory.xml changes, we need to be able to create a Java keystore for the server's private key. At this time, this is used only for an embedded mode web application which must use the same certificate as the FWD server and web clients.

The spawner also needs to use a Java trust-store to verify the server's certificate.

Creating the keystore is a 2 step process. Step 1 is to use openssl to export from .pem to a pkcs12 format file.

```
# export cert and key from pem to pkcs12 (no encryption)
openssl pkcs12 -export -in /etc/letsencrypt/live/demo.goldencode.com/fullchain.pem -inkey /etc/letsencrypt/liv
e/demo.goldencode.com/privkey.pem -out letsencrypt_fullchain_and_key.p12 -name standard
```

Step 2, imports from the pkcs12 format into a Java keystore.

```
# import the cert and key into a JKS file so Java code can use it, encrypt with a password because private key
s require this
keytool -importkeystore -deststorepass <PASSWORD_HERE> -destkeypass <PASSWORD_HERE> -destkeystore letsencrypt-
standard-private-key.store -srckeystore letsencrypt_fullchain_and_key.p12 -srcstoretype PKCS12 -srcstorepass "
" -alias standard
```

You can then use this letsencrypt-standard-private-key.store when starting the embedded web application server. In the sample project, you would leave the p2jks, p2jksalias, p2jkspw and p2jkepw as the existing values (do not change them).

However, you must set p2jts and webks to both specify the letsencrypt-standard-private-key.store. The webksalias must be set to standard (the same value as the alias option in the 2nd command above. Finally, the p2jtspw, webkspw and webkepw would all be set to the new password created in the 2nd command above (the keytool command).

By making these changes, the embedded web server application can be made to work using the same FWD private key and certificate. FYI, if you don't use it as the p2j truststore, then the connection to the FWD server will fail.

#4 - 02/06/2017 04:23 PM - Greg Shah

The final change that was needed was to enable the spawner with a truststore that allows proper authentication of the FWD server. I don't know for sure if this is needed, it needs checking.

I tried to create a truststore (a keystore that has only the root CA cert and does not have any private keys). But I was unable to get this created. I tried this (and many other variants):

```
# export cert from pem to pkcs12 (no encryption)
openssl pkcs12 -export -nokeys -in /etc/letsencrypt/live/demo.goldencode.com/fullchain.pem -out letsencrypt_fu
llchain.p12 -name standard

# import the cert and key into a JKS file so FWD can use it
keytool -importkeystore -deststorepass PASSWORD_HERE -destkeypass PASSWORD_HERE -destkeystore letsencrypt-srv-
certs.store -srckeystore letsencrypt_fullchain.p12 -srcstoretype PKCS12 -srcstorepass "" -alias standard
```

This results in 0 records imported (an empty keystore file).

However, it turns out that the keystore created with the FWD server's private key can be used without a password as a truststore. This is a bad idea from a security perspective because it makes the encrypted private key visible in the file system to other system users. But until we can figure out the problem, this is a workaround. I just specified this file as the -Dsrv.certs= option when installing the spawner and the result worked.

Again, it is not clear how necessary this is. If the cacerts already trusts the root CA, then perhaps this is not needed?

#5 - 06/13/2017 02:20 PM - Greg Shah

FYI, when you renew Let's Encrypt certificates, the following must be re-done:

1. Items 2 and 3 from #3240-2. The root CA doesn't change (nor does the hostname), but the server's private key changes (as does the certificate based on it).

2. The changes in both #3240-3 and #3240-4 are also needed.

#6 - 06/13/2017 02:28 PM - Greg Shah

We should make changes to make this process easy.

1. We can add support for Let's Encrypt such that we can connect to the CA and obtain certificates. Using something like https://github.com/shred/acme4j, we could automate this process.

2. We could then provide a tool to import such certificates into the directory and create the necessary file system keystores like the server's private key store and the srv-certs.store (truststore). Again, it seems like this would best be automated. The idea here is that this tool would work for importing any CA's certs including Let's Encrypt.

If the user's DNS and firewall setup is correct, then we could do the rest. This would make it easy to deploy and maintain real certificates for FWD servers.

#7 - 07/04/2017 10:41 AM - Constantin Asofiei

Some utilities which will improve the cert generation and management:

1. import/export/generate certificates and private key for a specific account. Currently SSLCertGenUtil re-generates everything (except root CA, which was added in 1514a).
2. remove the dependency on the FWD server certificate for the virtual desktop HTTPS connection (and iframe for embedded mode) and replace it with a dedicated certificate. We might want to move this alias/certificate specification directly at the webClient configuration and not rely on an existing process or server account for the certificate.
3. a tool which, given a root CA (from the directory or using an external provider like Let's Encrypt), generates a certificate and private key saved externally (not in the directory); this will remove the dependency on the embedded process account to generate the web certificate for the embedded app.

Also, there are some changes in 1514a related to SSLCertGenUtil and related classes - so make sure to look at those, until we have 1514a in trunk.

The part where the certificates for the HTTPS connections are not dependent on any FWD component will make it easy to re-generate them, in case browsers change some other policy (like with subject alternate name).

If I think of something else I'll comment here.

#8 - 07/05/2017 02:06 AM - Sergey Ivanovskiy

Constantin Asofiei wrote:

> 1 import/export/generate certificates and private key for a specific account. Currently SSLCertGenUtil re-generates everything (except root CA, which was added in 1514a).

Constantin, please help what sources are needed to check.

> 2 remove the dependency on the FWD server certificate for the virtual desktop HTTPS connection (and iframe for embedded mode) and replace it with a dedicated certificate. We might want to move this alias/certificate specification directly at the webClient configuration and not rely on an existing process or server account for the certificate.

Does it mean that the web client's embedded web server must use a new independent certificate? Are we planning to use the root CA and to sign dependent certificates by this root CA?

> 3 a tool which, given a root CA (from the directory or using an external provider like Let's Encrypt), generates a certificate and private key saved externally (not in the directory); this will remove the dependency on the embedded process account to generate the web certificate for the embedded app.

Does it mean that the web server of the embedded application must use a new independent certificate?

> Also, there are some changes in 1514a related to SSLCertGenUtil and related classes - so make sure to look at those, until we have 1514a in trunk.

I compared the trunc version with 1514a for this file com/goldencode/p2j/security/SSLCertGenUtil.java and it seems they are identical.

#9 - 07/05/2017 04:10 AM - Constantin Asofiei

Sergey Ivanovskiy wrote:

> Constantin Asofiei wrote:
>
> > 1 import/export/generate certificates and private key for a specific account. Currently SSLCertGenUtil re-generates everything (except root CA, which was added in 1514a).
>
> Constantin, please help what sources are needed to check.

Sorry, forgot these were in 2683a (and not 1514a), all changes are in trunk now.

> Does it mean that the web client's embedded web server must use a new independent certificate? Are we planning to use the root CA and to sign dependent certificates by this root CA?
> Does it mean that the web server of the embedded application must use a new independent certificate?

We should let the user choose the existing root CA in the directory, Let's Encrypt or an external root CA, to generate these certificates. So we will have both current way (where a self-signed root CA is used) and external/independent certificate. When generating these certificates, add also a step to generate the .der file to be imported into the browser (for the root CA), if a custom (or directory) root CA is used.

> > Also, there are some changes in 1514a related to SSLCertGenUtil and related classes - so make sure to look at those, until we have 1514a in trunk.
>
> I compared the trunc version with 1514a for this file com/goldencode/p2j/security/SSLCertGenUtil.java and it seems they are identical.

Everything is in trunk.

#10 - 07/06/2017 12:47 AM - Sergey Ivanovskiy

Thank you, it seems that the new feature should be to automate a support for Let's Encrypt and to add management of external CA certificates to the existing functionality that supports self-signed CA certificates.

Does it make sense to expose these functionality via new administration GWT client? In this case this task should be started from 3222a branch and committed in the trunk repo after 3222a.

#11 - 07/06/2017 07:47 AM - Greg Shah

Yes, it does make sense to integrate this into the admin interface. Please note that everything that can be done via the admin interface must also be available via a non-interactive command line tool. This is needed so that we can properly and fully automate all certificate processing.

#12 - 07/12/2017 02:25 AM - Sergey Ivanovskiy

Greg, Constantin, com/goldencode/p2j/security/SSLCertGenUtil.java has unnamed input arguments that can get its usage to be more complicated. I would like to add named parameters to this util. There is an interesting tool http://args4j.kohsuke.org/ under the MIT license that automates command line parameters for beans and more. Otherwise I can add our named parameters inline into this tool without any generalizations.

#13 - 07/13/2017 03:22 AM - Sergey Ivanovskiy

Committed revision 11157 added a certificate chain to the existing SSLCertGenUtil tool. I will add named parameters to this tool if there are no objections.

#14 - 07/13/2017 04:29 AM - Greg Shah

I'm OK with the args4j usage so long as it can be properly managed using gradle dependencies and the standard maven repositories.

Constantin: any opinions?

#15 - 07/14/2017 05:58 AM - Constantin Asofiei

Greg Shah wrote:

> I'm OK with the args4j usage so long as it can be properly managed using gradle dependencies and the standard maven repositories.
>
> Constantin: any opinions?

I'm OK if it can work cleanly and not complicate things too much.

#16 - 07/14/2017 12:36 PM - Sergey Ivanovskiy

This library is clear to use via Option annotation, http://args4j.kohsuke.org/args4j/apidocs/org/kohsuke/args4j/Option.html and its command line parser http://args4j.kohsuke.org/args4j/apidocs/org/kohsuke/args4j/CmdLineParser.html. Since SSLCertGenUtil tool can ask a user for the required input, boolean type for options is useless. If option is set, then its value is true, otherwise false. If it is false, we can't detect via boolean option if a user must be queried for the input. But it is not a problem, since enumerations or strings and even maps can be parsed and filled through command line parameters.

#17 - 07/15/2017 05:08 AM - Sergey Ivanovskiy

Committed rev 11159. Working to fix this program's flaw if the previous fresh run encrypted the CA private key, then running the second time with the previously logged password

```
Initializing service for directory /home/sbi/projects/3240a/tests/directory.xml...
Reading company configuration...
Using 'US' for [C] Country Code.
Using 'Alpharetta' for [L] Locality (city, etc).
Using 'GoldenCode' for [O] Organization.
Using 'Hotel' for [OU] Organization Unit.
Using 'GA' for [ST] State or Province Name.
Using 10 years for validity.
Enter RSA key strength (in bits), greater or equal than 2048 (blank for default 2048): 
Enter RSA public exponent, blank for default (65337): 
Adding account /security/accounts/processes/standard
Adding account /security/accounts/processes/ehotel
Adding account /security/accounts/processes/embedded
Adding account /security/accounts/users/bogus
WARNING: alias shared is set for multiple accounts!
Account /security/accounts/users/hotel ignored - no alias is defined
Account /security/accounts/users/serg ignored - no alias is defined
Account /security/accounts/users/qzln774g238k9z1y ignored - no alias is defined
Account /security/accounts/users/8k98qoge27mp87d5 ignored - no alias is defined
Account /security/accounts/users/j0ztj9p882f7x92m ignored - no alias is defined
Account /security/accounts/users/0l36qo3u9gn5m95x ignored - no alias is defined
Account /security/accounts/users/39n197j0xujr92wy ignored - no alias is defined
Account /security/accounts/users/4z1pwpq0v06919vi ignored - no alias is defined
Account /security/accounts/users/5md83y45qer07f7i ignored - no alias is defined
Account /security/accounts/users/7a9nny998w6gy01z ignored - no alias is defined
Re-use the encryption passwords currently in the directory (yes/no): yes
Re-use the root CA currently in the directory (yes/no): yes
Enter the root CA private-key encryption password: E4GdVNY8zy@vRT7Z7@0h9(0=qcXSrUA4aePK
Re-using root CA using [hotel-root] alias...
Exception in thread "main" com.goldencode.p2j.security.SSLCertGenException: com.goldencode.p2j.security.SSLCer
tGenException: The AES key must have 128, 192 or 256 bits!
    at com.goldencode.p2j.security.SSLCertGenUtil.generateRootCA(SSLCertGenUtil.java:1003)
    at com.goldencode.p2j.security.SSLCertGenUtil.generate(SSLCertGenUtil.java:348)
    at com.goldencode.p2j.security.SSLCertGenUtil.main(SSLCertGenUtil.java:1558)
Caused by: com.goldencode.p2j.security.SSLCertGenException: The AES key must have 128, 192 or 256 bits!
    at com.goldencode.p2j.security.BCCertFactory.cipherBytes(BCCertFactory.java:185)
    at com.goldencode.p2j.security.BCCertFactory.decryptPrivateKey(BCCertFactory.java:267)
    at com.goldencode.p2j.security.SSLCertGenUtil.generateRootCA(SSLCertGenUtil.java:989)
    ... 2 more
```

#18 - 07/15/2017 06:01 AM - Sergey Ivanovskiy

The incorrect password was used. The key store entry 36 chars password was used instead of hotel-root alias private key 32 chars protection password. So the previous test case is passed too.

#19 - 07/15/2017 07:24 AM - Sergey Ivanovskiy

SSLCertGenUtil input parameters were changed by this commit 11160.
It seems that the old input string was compact

```
directory.xml 2048 65337 yes yes fBMaU6k(r6<G^tUeZEg=BW48z3mM4M29 yes no yes no yes yes srv-certs.store yes ro
ot-ca-pk.store
```

but the current input parameters must be formatted but their positions in the input are not important now

```
-dir directory.xml
--rsa-key-size 2048
--rsa-public-exponent 65337
--reuse-passwords yes
--reuse-ca yes
--private-key-password "fBMaU6k(r6<G^tUeZEg=BW48z3mM4M29"
--export-account-keys yes
--use-common-account-store no
--export-server-keys yes
--use-common-server-store no
--export-server-certs yes
--include-ca yes
--server-certs-store srv-certs.store
--export-ca-key yes
--root-private-keys-store root-ca-pk.store
```

#20 - 07/17/2017 03:02 AM - Constantin Asofiei

Sergey Ivanovskiy wrote:

> SSLCertGenUtil input parameters were changed by this commit 11160.
> It seems that the old input string was compact
> [...]
> but the current input parameters must be formatted but their positions in the input are not important now
> [...]

What defaults do you have for these parameters? What happens if a parameter is missing - do you still rely on STDIN?

#21 - 07/17/2017 04:22 AM - Sergey Ivanovskiy

Constantin Asofiei wrote:

> Sergey Ivanovskiy wrote:
>
> > SSLCertGenUtil input parameters were changed by this commit 11160.
> > It seems that the old input string was compact
> > [...]
> > but the current input parameters must be formatted but their positions in the input are not important now
> > [...]
>
> What defaults do you have for these parameters? What happens if a parameter is missing - do you still rely on STDIN?

The default values are not set for these parameters and a user is requested for missing parameters. I didn't change the business logic, there are only additional changes: load external certificates and named options.

#22 - 07/17/2017 08:29 AM - Sergey Ivanovskiy

There are new options to load Let's Encrypt certificate and to use it for signing accounts and server certificates:

```
-L -cert "/etc/letsencrypt/live/demo.goldencode.com/cert.pem" -chain "/etc/letsencrypt/live/demo.goldencode.co
m/fullchain.pem" -key "/etc/letsencrypt/live/demo.goldencode.com/privkey.pem"
```

But I have not tested it yet. Another remaining work is to use acme4j client to get this Let's Encrypt certificate.

#23 - 10/25/2017 06:26 PM - Sergey Ivanovskiy

3240a is rebased over 11182. The current rev is 11187.

#24 - 10/26/2017 11:42 AM - Sergey Ivanovskiy

Constantin Asofiei wrote:

> Some utilities which will improve the cert generation and management:
>
> 1. import/export/generate certificates and private key for a specific account. Currently SSLCertGenUtil re-generates everything (except root

CA, which was added in 1514a).
2. remove the dependency on the FWD server certificate for the virtual desktop HTTPS connection (and iframe for embedded mode) and replace it with a dedicated certificate. We might want to move this alias/certificate specification directly at the webClient configuration and not rely on an existing process or server account for the certificate.
3. a tool which, given a root CA (from the directory or using an external provider like Let's Encrypt), generates a certificate and private key saved externally (not in the directory); this will remove the dependency on the embedded process account to generate the web certificate for the embedded app.

To these issues it is required to automate getting Let's Encrypt certificate.

Constantin, the 2 option is not clear for me now. It seems that the web client uses srv-certs.store as its key store and its trusted store. The current configuration seems simple and it doesn't need to know about this subject. Do we need to support the different web client certificate and private key pair for web clients embedded web server?

#25 - 10/26/2017 12:50 PM - Constantin Asofiei

Sergey Ivanovskiy wrote:

> Constantin Asofiei wrote:
>
> > Some utilities which will improve the cert generation and management:
> >
> > 1. import/export/generate certificates and private key for a specific account. Currently SSLCertGenUtil re-generates everything (except root CA, which was added in 1514a).
> > 2. remove the dependency on the FWD server certificate for the virtual desktop HTTPS connection (and iframe for embedded mode) and replace it with a dedicated certificate. We might want to move this alias/certificate specification directly at the webClient configuration and not rely on an existing process or server account for the certificate.
> > 3. a tool which, given a root CA (from the directory or using an external provider like Let's Encrypt), generates a certificate and private key saved externally (not in the directory); this will remove the dependency on the embedded process account to generate the web certificate for the embedded app.
>
> To these issues it is required to automate getting Let's Encrypt certificate.
>
> Constantin, the 2 option is not clear for me now. It seems that the web client uses srv-certs.store as its key store and its trusted store.

No, srv-certs.store contains only server certificates plus the root CA certificate.

> The current configuration seems simple and it doesn't need to know about this subject. Do we need to support the different web client certificate and private key pair for web clients embedded web server?

The idea here is this: the web client's HTTPS connection relies on the server's certificate, usually named 'standard'. We need to decouple this, and have separate certificate for it.

#26 - 10/26/2017 03:43 PM - Constantin Asofiei

Segey, do you need help with any of this stuff?

#27 - 10/26/2017 03:54 PM - Sergey Ivanovskiy

Constantin Asofiei wrote:

> Segey, do you need help with any of this stuff?

The current state is that the external root CA can be loaded and the other dependent certificates are generated. Next planning to automate getting Let's Encrypt certificate since this task can be potentially useful, then planning to test certificate generation from the given Let's Encrypt certificate, then planning to do the 1 import/export/generate certificates and private key for a specific account, then the 2 or 3. According to Greg letters it is not required to complete this task to 30 Oct. The 2 task probably requires the server side changes, but now this task only is assumed to fix helper tools. I think the 2 part can wait.It can help to concentrate on the other tasks.

#28 - 10/27/2017 07:03 PM - Sergey Ivanovskiy

According to https://shredzone.org/maven/acme4j/challenge/tls-sni-01.html acme4j client with a tunable web server can help to automate getting of Let's Encrypt certificates. Planning to use https Jetty web server that is started with a self-signed certificate and private key pair generated for the subject subjectAltName from the Let's Encrypt ACME server. It seems that jetty supports the required SNI requests https://webtide.com/jetty-9-3-features/. The controlled web server should respond on the Let's Encrypt request with this generated certificate.

#29 - 10/29/2017 04:19 AM - Sergey Ivanovskiy

3240a was rebased rev. 11189.

#30 - 10/29/2017 01:11 PM - Sergey Ivanovskiy

Please review the committed revision 11190 that added new option --build-alias-certs broker1 to rebuild certs for the given alias.

```
java -cp ...  com.goldencode.p2j.security.SSLCertGenUtil
-dir /home/sbi/projects/3240a/tests/directory.xml
--rsa-key-size 2048
--rsa-public-exponent 65337
--reuse-passwords yes
--reuse-ca yes
--private-key-password "^u0mrGn)o2X5nor109+KAvt&teTCn2v7"
--export-account-keys yes
--use-common-account-store no
--export-server-keys yes
--use-common-server-store no
--export-server-certs yes
--include-ca yes
--server-certs-store srv-certs.store
--export-ca-key yes
--root-private-keys-store root-ca-pk.store
--build-alias-certs  broker1
```

#31 - 10/29/2017 05:36 PM - Sergey Ivanovskiy

Committed rev. 11191 simplified -L option to load external CA certificates.

```
-dir /home/sbi/projects/3240a/tests/directory.xml
--rsa-key-size 2048
--rsa-public-exponent 65337
--reuse-passwords yes
--private-key-password "^u0mrGn)o2X5nor109+KAvt&teTCn2v7"
--export-account-keys yes
--use-common-account-store no
--export-server-keys yes
--use-common-server-store no
--export-server-certs yes
--include-ca yes
--server-certs-store srv-certs.store
--export-ca-key yes
--root-private-keys-store root-ca-pk.store
-L cert=root-ca-pk.crt
-L key=root-ca-pk.key
```

The case was checked for the self-signed certificates. The rest part is getting Let's Encrypt certificates. Postponed this task for the next day 30 October 2017.

#32 - 10/30/2017 08:33 AM - Constantin Asofiei

Review for branch 3240a revision 11192:

1. please add history entries
2. SSLCertFactory is having a dependency on BouncyCastle - can you rework it so that is removed?
3. SSLCertGenUtil - change imports to be with .*
4. SSLCertGenUtil.saveCertificatePairInExternalStore is missing javadoc

Otherwise, please check if normal operation works OK (i.e. full re-generation including root CA).

#33 - 10/30/2017 10:28 AM - Sergey Ivanovskiy

Committed rev. 11193 fixed these issues and tested certificates regeneration with different settings using a common master password or reuse passwords from the directory.

#34 - 10/30/2017 02:56 PM - Sergey Ivanovskiy

Planning to utilize SecureWebServer and SSLCertFactory in order to start the target managed web server to prove that the ACME client is the owner of the target web server. The code will be ready for the review today (~ 2 hours). Planning to add new dependencies

```
   fwdClientServer group: 'org.shredzone.acme4j', name: 'acme4j-client', version: '0.10'
   fwdClientServer group: 'org.shredzone.acme4j', name: 'acme4j-utils', version: '0.10'
   -fwdClientServer group: 'org.slf4j', name: 'slf4j-jdk14', version: '1.6.1'-
```

#35 - 10/30/2017 05:13 PM - Sergey Ivanovskiy

Greg, this new functionality to get Let's Encrypt signed certificates is not ready yet. It takes more time than I expected although it seems there are no additional issues during this implementation. Can it be later?

#36 - 10/30/2017 05:30 PM - Sergey Ivanovskiy

Is it required to add this license note at the header

```
/*
*
* acme4j - Java ACME client
*
* Copyright (C) 2015 Richard "Shred" Körber
*   http://acme4j.shredzone.org
*
* Licensed under the Apache License, Version 2.0 (the "License");
* you may not use this file except in compliance with the License.
*
* This program is distributed in the hope that it will be useful,
* but WITHOUT ANY WARRANTY; without even the implied warranty of
* MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.
*/
```

if we are using the code of org.shredzone.acme4j package and an example of ACME client?

#37 - 10/30/2017 05:36 PM - Greg Shah

Sergey Ivanovskiy wrote:

> Greg, this new functionality to get Let's Encrypt signed certificates is not ready yet. It takes more time than I expected although it seems there are no additional issues during this implementation. Can it be later?

Yes.

#38 - 10/30/2017 05:37 PM - Greg Shah

Using (calling) a package from a library should not require a different copyright notice in our code.

Do not copy any sample code that is copyrighted.

#39 - 10/31/2017 08:42 AM - Greg Shah

Is there a safe version of this branch that can be merged to trunk? What testing is needed? It is OK to finish the work in another branch.

#40 - 10/31/2017 09:05 AM - Sergey Ivanovskiy

I added Acme4J dependencies and it used this version '1.56' of Bouncy Castle transitively, but the current version is used
fwdServer group: 'org.bouncycastle', name: 'bcprov-jdk15on', version: '1.50'
fwdServer group: 'org.bouncycastle', name: 'bcpkix-jdk15on', version: '1.50'.
Don't know https://shredzone.org/maven/acme4j/acme4j-example/license.html is fit us.

#41 - 10/31/2017 09:06 AM - Sergey Ivanovskiy

It looks safe. Planning to rebase over the current trunc. Now the rev. is 11202.

#42 - 10/31/2017 09:07 AM - Greg Shah

Acme4j is Apache 2.0 which is fine. Can you split off the acme4j changes into a separate branch and just make this branch safe for the trunk?

#43 - 10/31/2017 09:07 AM - Greg Shah

My thinking is that the acme4j stuff should be looked at more carefully, but the other utility changes are safer.

#44 - 10/31/2017 09:09 AM - Sergey Ivanovskiy

Yes, planning to move AcmeClient and ManagedWebServer into 3240b and their dependencies.

#45 - 10/31/2017 09:14 AM - Sergey Ivanovskiy

Planning to move this bzr diff -r11197..11202 > 3240b_1.txt into 3240b.

#46 - 10/31/2017 09:16 AM - Sergey Ivanovskiy

- File 3240b_1.txt added

This diff is moved out.

#47 - 10/31/2017 09:23 AM - Sergey Ivanovskiy

Committed revision 11203 of 3240a is ready. Greg, can I merge 3240a after rebasing with trunc 11191 or wait the rev. 11192?

#48 - 10/31/2017 09:25 AM - Greg Shah

Please rebase to 11192. I'm doing a code review of 3240a right now. Wait to merge until I confirm.

#49 - 10/31/2017 09:26 AM - Sergey Ivanovskiy

OK. Today I found https://letsencrypt.org/2017/10/17/acme-support-in-apache-httpd.html that Apache module automates ACME certificates.

#50 - 10/31/2017 09:41 AM - Sergey Ivanovskiy

3240a was rebased to 11207 from trunc revision 11192. It is ready for review.

#51 - 10/31/2017 09:56 AM - Greg Shah

Code Review Task branch 3240a Revision 11207

build.gradle has the args4j change that may need some testing. It also does need a history entry.

How safe is this addition (does it pull any other libraries in or change any existing library versions)?

#52 - 10/31/2017 09:57 AM - Greg Shah

Constantin: What is your opinion of the changes?

#53 - 10/31/2017 09:57 AM - Greg Shah

FYI the args4j is licensed under the MIT license, which is permissive and is allowed for us.

#54 - 10/31/2017 09:59 AM - Sergey Ivanovskiy

It is safe and has no transitive dependencies that can brake another parts. http://args4j.kohsuke.org/dependency-convergence.html

#55 - 10/31/2017 09:59 AM - Constantin Asofiei

Sergey Ivanovskiy wrote:

> It is safe and has no transitive dependencies that can brake another parts. http://args4j.kohsuke.org/dependency-convergence.html

To be sure, please compare the build/lib contents for trunk and 3240a.

#56 - 10/31/2017 10:20 AM - Sergey Ivanovskiy

I can confirm that only args4j-2.33.jar is added to the build/lib.

#57 - 10/31/2017 10:20 AM - Constantin Asofiei

Greg Shah wrote:

> Constantin: What is your opinion of the changes?

The changes look OK to me.

Sergey: add a history entry to build.gradle and you can release it.

#58 - 10/31/2017 10:29 AM - Sergey Ivanovskiy

Committed revision 11208 added history, planning to merge into trunc.

#59 - 10/31/2017 11:01 AM - Sergey Ivanovskiy

3240a was committed into the trunc rev 11193 and archived. New task branch 3240b was created.

#60 - 10/31/2017 11:41 AM - Sergey Ivanovskiy

Committed revision 11194 added acme client as a standalone tool yet. It can be used as this one on devsrv01 with 3240b.

```
java -cp ... com.goldencode.p2j.security.AcmeClient

-server "acme://letsencrypt.org/staging"
-domains "ssltest.goldencode.com"
-host "72.17.138.126"
-port 10443
```

#61 - 11/01/2017 12:40 PM - Greg Shah

A report of some results from testing the ACME Java code on a demo.goldencode.com:

On 11/01/2017 12:19 PM, Serg Ivanovskiy wrote:

> Greg,
>
> OK, this program works correctly with production server, but the Let's Encrypt test server doesn't provide real certificates
>
> java -cp ./lib/p2j.jar com.goldencode.p2j.security.AcmeClient -host <ip_address_redacted> -port 8443 -server "acme://letsencrypt.org/" -domains "demo.goldencode.com" -S C="US" -S S="GA" -S L="Alpharetta" -S O="GoldenCode" -S OU="Demo"
> SLF4J: Class path contains multiple SLF4J bindings.
> SLF4J: Found binding in [jar:file:/opt/tests/lib/slf4j-jdk14-1.6.1.jar!/org/slf4j/impl/StaticLoggerBinder.class]
> SLF4J: Found binding in [jar:file:/opt/tests/lib/slf4j-simple-1.6.1.jar!/org/slf4j/impl/StaticLoggerBinder.class]
> SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
> SLF4J: Actual binding is of type [org.slf4j.impl.JDK14LoggerFactory]
> Nov 01, 2017 12:09:09 PM com.goldencode.p2j.security.AcmeClient main
> INFO: Starting up...
> Nov 01, 2017 12:09:09 PM com.goldencode.p2j.security.AcmeClient readUserAccount
> INFO: Can't get the client registration URI from user.reg'
> java.io.FileNotFoundException: user.reg (No such file or directory)
> at java.io.FileInputStream.open0(Native Method)
> at java.io.FileInputStream.open(FileInputStream.java:195)
> at java.io.FileInputStream.<init>(FileInputStream.java:138)
> at com.goldencode.p2j.security.AcmeClient.readUserAccount(AcmeClient.java:430)
> at com.goldencode.p2j.security.AcmeClient.main(AcmeClient.java:669)
>
> Nov 01, 2017 12:09:12 PM com.goldencode.p2j.security.AcmeClient main
> INFO: The client account is https://acme-v01.api.letsencrypt.org/acme/reg/<acct_redacted>
> Nov 01, 2017 12:09:13 PM com.goldencode.p2j.security.AcmeClient authorize
> INFO: Authorization for domain demo.goldencode.com
> Nov 01, 2017 12:09:14 PM com.goldencode.p2j.security.AcmeClient tlsSniChallenge
> INFO: Please configure your web server.
> Nov 01, 2017 12:09:14 PM com.goldencode.p2j.security.AcmeClient tlsSniChallenge
> INFO: It must return the certificate 'tlssni.crt' on a SNI request to:
> Nov 01, 2017 12:09:14 PM com.goldencode.p2j.security.AcmeClient tlsSniChallenge
> INFO: <text_redacted>.acme.invalid
> Nov 01, 2017 12:09:14 PM com.goldencode.p2j.security.AcmeClient tlsSniChallenge
> INFO: The matching keypair is available at 'tlssni.key'.
> Nov 01, 2017 12:09:14 PM com.goldencode.p2j.security.AcmeClient tlsSniChallenge
> INFO: If you're ready, dismiss the dialog...
> Nov 01, 2017 12:09:14 PM com.goldencode.p2j.security.AcmeClient tlsSniChallenge
> INFO: Please use 'tlssni.key' and 'tlssni.crt' cert for SNI requests to:
>
> https://<url_redacted>.acme.invalid
>
> Nov 01, 2017 12:09:14 PM org.eclipse.jetty.util.log.Log initialized

```
INFO: Logging initialized @4959ms
Nov 01, 2017 12:09:14 PM org.eclipse.jetty.server.Server doStart
INFO: jetty-9.3.z-SNAPSHOT
Nov 01, 2017 12:09:14 PM org.eclipse.jetty.server.handler.ContextHandler doStart
INFO: Started o.e.j.s.h.ContextHandler@5542c4ed{/,null,AVAILABLE,<text_redacted>.acme.invalid}
Nov 01, 2017 12:09:14 PM org.eclipse.jetty.server.handler.ContextHandler doStart
INFO: Started o.e.j.s.h.ContextHandler@1573f9fc{/,null,AVAILABLE}
Nov 01, 2017 12:09:14 PM org.eclipse.jetty.server.AbstractConnector doStart
INFO: Started ServerConnector@4116aac9{SSL,[ssl, http/1.1]}{<ip_address_redacted>:8443}
Nov 01, 2017 12:09:14 PM org.eclipse.jetty.server.Server doStart
INFO: Started @5065ms
Nov 01, 2017 12:09:14 PM GenericWebServer.startup
INFO: {main} Server URL: https://&lt;ip_address_redacted&gt;:8443/
Nov 01, 2017 12:09:19 PM org.eclipse.jetty.server.AbstractConnector doStop
INFO: Stopped ServerConnector@4116aac9{SSL,[ssl, http/1.1]}{<ip_address_redacted>:8443}
Nov 01, 2017 12:09:19 PM org.eclipse.jetty.server.handler.ContextHandler doStop
INFO: Stopped o.e.j.s.h.ContextHandler@5542c4ed{/,null,UNAVAILABLE,<text_redacted>.acme.invalid}
Nov 01, 2017 12:09:19 PM org.eclipse.jetty.server.handler.ContextHandler doStop
INFO: Stopped o.e.j.s.h.ContextHandler@1573f9fc{/,null,UNAVAILABLE}
Nov 01, 2017 12:09:20 PM com.goldencode.p2j.security.AcmeClient askCertificate
INFO: Success! The certificate for domains [demo.goldencode.com] has been generated!
Nov 01, 2017 12:09:20 PM com.goldencode.p2j.security.AcmeClient askCertificate
INFO: Certificate URI: https://acme-v01.api.letsencrypt.org/acme/cert/&lt;url_redacted>
```

There are in

```
ll
total 56
drwxr-xr-x 3 sbi  sbi   4096 Nov  1 12:09 ./
drwxr-xr-x 7 root root  4096 Nov  1 11:09 ../
~~rw-rw-r-~~ 1 sbi  sbi   3452 Nov  1 12:09 domain-chain.crt
~~rw-rw-r-~~ 1 sbi  sbi   1805 Nov  1 12:09 domain.crt
~~rw-rw-r-~~ 1 sbi  sbi   1082 Nov  1 12:09 domain.csr
~~rw-rw-r-~~ 1 sbi  sbi   1675 Nov  1 12:09 domain.key
drwxrwx--- 2 sbi  sbi  12288 Nov  1 09:22 lib/
~~rw-rw-r-~~ 1 sbi  sbi   2143 Nov  1 12:09 managed-server.store
~~rw-rw-r-~~ 1 sbi  sbi   1115 Nov  1 12:09 tlssni.crt
~~rw-rw-r-~~ 1 sbi  sbi   1675 Nov  1 12:09 tlssni.key
~~rw-rw-r-~~ 1 sbi  sbi   1679 Nov  1 11:58 user.key
~~rw-rw-r-~~ 1 sbi  sbi    121 Nov  1 12:09 user.reg
```

domain* were signed by Let's Encrypt and registered accout key and url are in user.key and user.reg. The private certificate user.key must be in secret it is related to registration of Let's Encrypt account and domain.key is secret too.

#62 - 11/01/2017 12:45 PM - Greg Shah

> this program works correctly with production server

Do you mean the Let's Encrypt production certificate server?

> but the Let's Encrypt test server doesn't provide real certificates

Do we care about this? With demo.goldencode.com you can get a real certificate from the production server. If that works well, then I think it is fine. Unless I am misunderstanding.

> INFO: Please configure your web server.

Is this an interactive process? We are wanting a fully automated ACME registration process.

> domain* were signed by Let's Encrypt and registered accout key and url are in user.key and user.reg. The private certificate user.key must be in secret it is related to registration of Let's Encrypt account and domain.key is secret too.

Do we have a way to easily put all of these into the right keystores/directory entries to make a real FWD server work?

#63 - 11/01/2017 12:57 PM - Sergey Ivanovskiy

Yes, I meant that this option -server "acme://letsencrypt.org/" must point to the Let's Encrypt production certificate server in order to get signed certificates. For me it is not obvious that the server accessed by "acme://letsencrypt.org/staging" shouldn't return certificates. This server "acme://letsencrypt.org/staging" returns INVALID status when someone tries to get signed certificates and it means that the ACME client works fine.

> Do we have a way to easily put all of these into the right keystores/directory entries to make a real FWD server work?

Yes, I am planning to add new options to the base tool com.goldencode.p2j.security.SSLCertGenUtil in order that it gets signed certificates and loads them into directory and updates all dependent certificates. AcmeClient works properly without user dialog prompts.

#64 - 11/01/2017 04:07 PM - Sergey Ivanovskiy

Tested new Let's Encrypt certificates to be fed into com.goldencode.p2j.security.SSLCertGenUtil with these options

```
-dir /home/sbi/projects/3240b/tests/directory.xml
--rsa-key-size 2048
--rsa-public-exponent 65337
--reuse-passwords no
--private-key-password "****************************"
--export-account-keys yes
--use-common-account-store no
--export-server-keys yes
--use-common-server-store no
--export-server-certs yes
--include-ca yes
--server-certs-store srv-certs.store
--export-ca-key yes
--root-private-keys-store root-ca-pk.store
-L cert=./certs/domain.crt
-L key=./certs/domain.key
-L chain=./certs/domain-chain.crt
```

and then exported hotel-root.der and investigated it manually. It looks that we got the base part of this task. This certificate is issued by

```
CN = Let's Encrypt Authority X3
O = Let's Encrypt
C = US
```

and has this key usage

```
Critical
Signing
Key Encipherment
```

and is valid for 3 months. There are unimplemented use cases related to renew the outdated certificate, revocation of this certificate, and ACME account and authorization management.

#65 - 11/02/2017 10:48 AM - Greg Shah

Code Review Task Branch 3240b Revision 11196

The changes are OK.

1. build.gradle and SSLCertGenUtil need history entries.

2, The logging in AcmeClient.tlsSniChallenge() is confusing. It suggests an interactive process but you said previously that it is not interactive.

#66 - 11/02/2017 10:50 AM - Greg Shah

> There are unimplemented use cases related to renew the outdated certificate, revocation of this certificate, and ACME account and authorization management.

Do you anticipate any issues implementing these features? The renewal in particular is very important.

If the other items are difficult, then we may be able to get away with using another client (e.g. the letsencrypt package) for those. Getting and renewing certs is the most important part since the standard clients don't integrate nicely with Java keystores or our directory.xml.

Still, if the other features are easy it is best to add them to our utility to make it complete.

#67 - 11/02/2017 11:08 AM - Sergey Ivanovskiy

Renew is the same process, it requires to run this utility again. Planning to fix issues from #3240-65, and to do it complete today.

#68 - 11/03/2017 11:40 AM - Sergey Ivanovskiy

I encountered some kind of content conflicts due to the previous merge (forgot bzr bind after bzr rebase). But the content of my changes is saved. Please review the committed revision 11197. My previous sequence of commits and merge were removed. The 3240b is rebased over trunc rev. 11194.

#69 - 11/03/2017 11:59 AM - Greg Shah

Code Review Task Branch 3240b Revision 11197

I'm OK with the changes.

#70 - 11/03/2017 12:00 PM - Greg Shah

What is left to do in this task?

#71 - 11/03/2017 12:11 PM - Sergey Ivanovskiy

I forgot to add these new options to com.goldencode.p2j.security.SSLCertGenUtil in order to use one tool

```
--get-lets-encrypt userkey=./certs/user.key
--get-lets-encrypt userreg=./certs/user.reg
--get-lets-encrypt domains="test8.acme.com test7.acme.com"
--lets-encrypt-client-host host_ip
--lets-encrypt-client-port 9999
--lets-encrypt-server "acme://letsencrypt.org/staging"
-S O="Org" -S OU="Dev" -S C="RU" -S L="Tver" -S S="RU"
```

Please wait I will commit it.

#72 - 11/03/2017 05:13 PM - Sergey Ivanovskiy

Greg, please review these changes rev 11199. New option to request Let's Encrypt certificates and to rebuild all directory certificates was added.

```
-dir /home/sbi/projects/3240b/tests/directory.xml
--rsa-key-size 2048
--rsa-public-exponent 65337
--reuse-passwords no
--private-key-password "*************************"
--export-account-keys yes
--use-common-account-store no
--export-server-keys yes
--use-common-server-store no
--export-server-certs yes
--include-ca yes
--server-certs-store srv-certs.store
--export-ca-key yes
--root-private-keys-store root-ca-pk.store
-domains "test8.acme.com    test7.acme.com  "
-host "192.168.1.37"
-port 9999
-server "acme://letsencrypt.org/staging"
-S O="Org" -S OU="Dev" -S C="**" -S L="****" -S S="**"
-acme REQUEST
```

Now the task is complete and ready for review.

#73 - 11/04/2017 10:48 AM - Greg Shah

Code Review Task Branch 3420b Revision 11200

Generally, I'm OK with the changes. I've checked in some minor formatting changes.

1. Can we set some safe defaults for the SSLCertGenUtil.InputParameters? If there are some safe values that are almost always used, then I'd like to default them so that the command line usage is simpler for the common case. This should take in account that the tool will be commonly used for both self-signed and ACME use cases, so we don't want to force one or the other by default.

2. Please put all the private methods in AcmeClient at the bottom of the file (above main).

Constantin: please review.

#74 - 11/05/2017 11:02 AM - Sergey Ivanovskiy

Yes, it seems that these options can be set to these default values:

```
--rsa-key-size 2048
--rsa-public-exponent 65537
--export-account-keys yes
--use-common-account-store no
--export-server-keys yes
--use-common-server-store no
--export-server-certs yes
--include-ca yes
--export-ca-key yes
--root-private-keys-store root-ca-pk.store
--server-certs-store srv-certs.store
```

Committed revision 11204 fixed #3240-73.

#75 - 11/06/2017 10:08 AM - Constantin Asofiei

Greg, I'm OK with the changes in 3240b rev 11204.

#76 - 11/06/2017 10:12 AM - Greg Shah

Code Review Task Branch 3420b Revision 11204

I'm fine with the changes.

Since the changes are just for utility code that is not used at conversion or runtime, no regression testing is needed. Please merge 3240b to trunk.

#77 - 11/06/2017 11:12 AM - Sergey Ivanovskiy

3240b was merged into the trunc as rev 11196 and archived.

#78 - 11/06/2017 11:18 AM - Greg Shah

- % Done changed from 0 to 100

- Status changed from WIP to Closed

#79 - 11/08/2017 11:06 AM - Constantin Asofiei

Sergey, please let me know what value should I use to test Let's Encrypt, for these options. Is it possible to do this without using demosrv01?

```
--private-key-password "*************************"
-domains "test8.acme.com    test7.acme.com  "
-host "192.168.1.37"
-port 9999
-server "acme://letsencrypt.org/staging"
-acme REQUEST
```

#80 - 11/08/2017 11:25 AM - Sergey Ivanovskiy

It is possible to use another host for which there is a registered domain name that is accessible by some external IP address. These options should work

```
java -cp ../../p2j/build/lib/p2j.jar com.goldencode.p2j.security.SSLCertGenUtil \
 -dir ./directory.xml \
 --reuse-passwords yes \
 -domains "test8.acme.com test7.acme.com" \
 -host "Host_IP_Address" \
 -port 443_Redirected_Port \
 -server "acme://letsencrypt.org/" \
 -S O="Organization" -S OU="Organization_Unit" -S C="Country_Code" -S L="Locality_city" -S S="State_province" 
\
 -acme REQUEST
```

#81 - 11/08/2017 11:27 AM - Constantin Asofiei

Can you explain how -host and -port work? Should these be accessible via the internet? Are they the host/port from the machine where the SSLCertGenUtil command is ran?

#82 - 11/08/2017 11:33 AM - Sergey Ivanovskiy

Yes, it is the host and port on the machine on which this tool is executed and its host IP should be accessible at the port 443. Thus the -port option

should point to 443 or another local port on which one the traffic from 443 is redirected.
(redir --lport 8443 --cport 443 &)
Please take into account that acme://letsencrypt.org/staging doesn't provide any certificates and can be used only for testing the client functionality. But this
acme://letsencrypt.org/ returns valid certificates.

#83 - 11/08/2017 11:39 AM - Constantin Asofiei

Sergey Ivanovskiy wrote:

> Yes, it is the host and port on the machine on which this tool is executed and its host IP should be accessible at the port 443. Thus the -port option should point to 443 or another local port on which one the traffic from 443 is redirected.
> (redir --lport 8443 --cport 443 &)

Why would you need this port redirection on the machine? Just not to expose 443 to the internet?

> Please take into account that acme://letsencrypt.org/staging doesn't provide any certificates and can be used only for testing the client functionality. But this
> acme://letsencrypt.org/ returns valid certificates.

Have you tested acme://letsencrypt.org/ ?

#84 - 11/08/2017 11:46 AM - Sergey Ivanovskiy

Because 443 port is a privilege and this tool can be executed without sudo rights. 443 must be open to the Internet. Yes, I tested it on demsrv01. Do you have any issues? I don't know if dynamic domains can be used or not because I tested this functionality with acme://letsencrypt.org/staging and decided that they couldn't be used.

#85 - 11/08/2017 11:46 AM - Greg Shah

> Why would you need this port redirection on the machine? Just not to expose 443 to the internet?

There are multiple possible reasons:

1. More than one HTTPS server on the same machine.
2. Avoidance of the restricted 443 (as a port below 1024) and thus an application without root privs can listen on it.
3. The machine is behind a firewall which port maps to something other than 443.

Since the Let's Encrypt ACME process can only operate with 443, the use of redirection in the above cases is needed.

#86 - 11/12/2017 04:47 AM - Sergey Ivanovskiy

Created task branch 3240c to setup a key store for P2J web server and local web clients based on Let's Encrypt certificates.

#87 - 11/12/2017 05:14 AM - Sergey Ivanovskiy

Link to the task #3366-82.

#88 - 11/12/2017 06:48 AM - Sergey Ivanovskiy

Committed rev. 11200 added certificate suite and changed accordingly. Planning to use this CertificateSuite to setup web servers. This usage example is in ManagedWebServer.

#89 - 11/12/2017 04:48 PM - Sergey Ivanovskiy

Let me consider Hotel GUI and its certificates usages.

1) The "standard" process and P2J web server both use the same private and public certificates to serve clients via application protocol over secure sockets and via https correspondingly. The standard-private-key.store contains the private key and srv-certs.store contains its public key and can contain CA public key.
2) The web clients use the same certificates too via ServerKeyStore.

3)The next case is the embedded web application that uses ehotel-private-key.store for its embedded P2J client for each https request to the embedded web application but
the web server of the embedded web application uses embedded-private-key.store to setup secure socket connection. And created embedded web clients use ServerKeyStore thus
they should be the same standard-private-key.store for their https connections and srv-certs.store for their P2J clients connections.

Correct?

If it is correct, then it follows that now P2J web server and web clients setup their web services via ServerKeyStore and the last one serves the P2J server certificates.
Now we can only force P2J server use the Let's Encrypt private and public certificates via standard-private-key.store and srv-certs.store, but it means that internal secure sockets will use the same keys.

Greg, what do you think about how to improve this design in order to use Let's Encrypt certificates independently from internal P2J certificates? Please share your design.

#90 - 11/13/2017 04:05 PM - Sergey Ivanovskiy

3240c is rebased over 11200 trunc. Planning to use different public and private pairs of certificates for P2J server and its web services including web clients. It seems that ServerKeyStore should implement byte[] getWebStore().

#91 - 11/14/2017 09:05 AM - Sergey Ivanovskiy

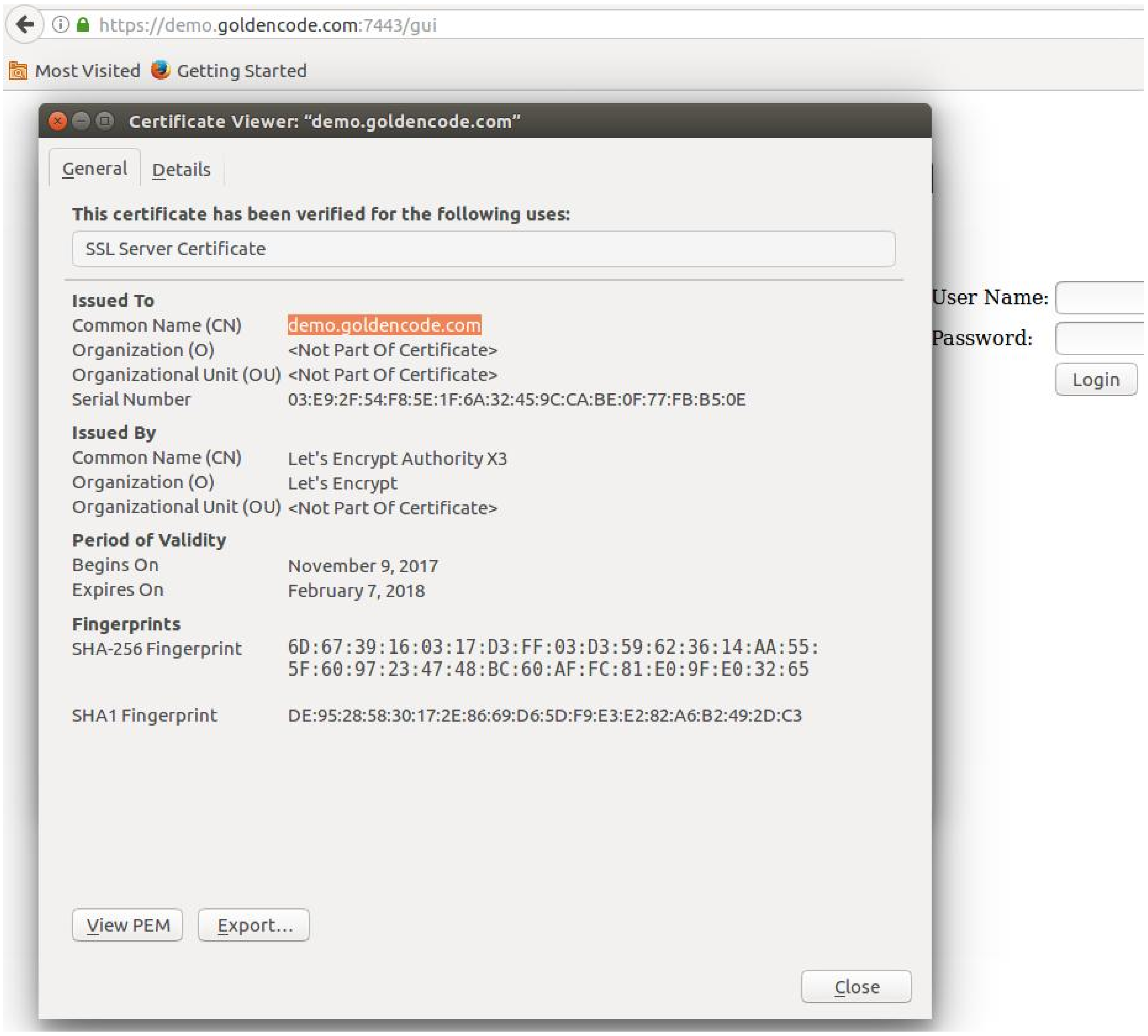
Planning to add this new container webCertificates to the directory under server/default

```
        <node class="boolean" name="adminEnabled">
          <node-attribute name="value" value="TRUE"/>
        </node>
        <node class="integer" name="adminPort">
          <node-attribute name="value" value="7443"/>
        </node>
        <node class="container" name="webCertificates">
          <node class="string" name="alias">
            <node-attribute name="value" value="web-alias"/>
          </node>
          <node class="string" name="certificate">
            <node-attribute name="value" value="domain.crt"/>
          </node>
          <node class="string" name="chain">
            <node-attribute name="value" value="domain-chain.crt"/>
          </node>
          <node class="string" name="key">
            <node-attribute name="value" value="domain.key"/>
          </node>
        </node>
```

and to create SSLCertGenUtil task to build web-private-key.store containing private and public certificates from Let's Encrypt to be used by the embedded web application server. Is it OK if the private key domain.key will be stored in the disk file and its name will be in the directory?

#92 - 11/14/2017 09:12 AM - Sergey Ivanovskiy

- File LetsEncryptCerts.png added



https://demo.goldencode.com:7443/gui

Certificate Viewer: "demo.goldencode.com"

This certificate has been verified for the following uses:

SSL Server Certificate

Issued To

Common Name (CN)	demo.goldencode.com
Organization (O)	<Not Part Of Certificate>
Organizational Unit (OU)	<Not Part Of Certificate>
Serial Number	03:E9:2F:54:F8:5E:1F:6A:32:45:9C:CA:BE:0F:77:FB:B5:0E

Issued By

Common Name (CN)	Let's Encrypt Authority X3
Organization (O)	Let's Encrypt
Organizational Unit (OU)	<Not Part Of Certificate>

Period of Validity

Begins On	November 9, 2017
Expires On	February 7, 2018

Fingerprints

SHA-256 Fingerprint	6D:67:39:16:03:17:D3:FF:03:D3:59:62:36:14:AA:55: 5F:60:97:23:47:48:BC:60:AF:FC:81:E0:9F:E0:32:65
SHA1 Fingerprint	DE:95:28:58:30:17:2E:86:69:D6:5D:F9:E3:E2:82:A6:B2:49:2D:C3

#93 - 11/15/2017 08:45 AM - Sergey Ivanovskiy

I changed the web certificate configuration to this one in order to support private and public certificate pairs under this node /security/certificates

```
        <node class="container" name="webCertificates">
          <node class="string" name="alias">
            <node-attribute name="value" value="web-alias"/>
          </node>
          <node class="bytes" name="certificate">
            <node-attribute name="value"  value="" />
          </node>
          <node class="container" name="chain">
           <node class="bytes" name="web-alias-1">
            <node-attribute name="value"  value="" />
           </node>
           <node class="bytes" name="web-alias-2">
            <node-attribute name="value"  value="" />
           </node>
          </node>
          <node class="container" name="key">
            <node class="bytes" name="key-entry">
              <node-attribute name="value" value=""/>
            </node>
            <node class="bytes" name="key-password">
              <node-attribute name="value" value=""/>
            </node>
          </node>
<!--
          <node class="string" name="certificate">
            <node-attribute name="value" value="domain.crt"/>
          </node>
          <node class="string" name="chain">
            <node-attribute name="value" value="domain-chain.crt"/>
          </node>
          <node class="string" name="key">
            <node-attribute name="value" value="domain.key"/>
          </node>
-->
        </node>
```

It requires to develop the code to read, to write and to update this configuration.

#94 - 11/15/2017 09:16 AM - Greg Shah

I'm OK with your directory changes. I prefer them to be in the certificates branch.

Constantin: Any objections?

> Is it OK if the private key domain.key will be stored in the disk file and its name will be in the directory?

For the web clients, we have been trying to send down an in-memory key store with the certs. This avoids having to store it in the file system. By putting it in the directory and then sending it down to the client over the FWD session, it makes it easier to deploy the client (less configuration).

Are there any benefits to storing it in the file system? I'm OK with making utility code changes to load it into the directory.

#95 - 11/15/2017 07:44 PM - Sergey Ivanovskiy

Decided to use separate configuration webCertificates because the trusted chain certificates can be easily retrieved from webCertificates/chain and there is possibility to use server process certificates from the directory to set up web server store.

Please review the committed revision 11202. After java -cp path to p2j.jar com.goldencode.p2j.security.SSLCertGenUtil has been executed with

```
-dir ./directory.xml 
--rsa-key-size 2048
--rsa-public-exponent 65337
--reuse-passwords yes
--export-account-keys yes
--use-common-account-store no
--export-server-keys yes
--use-common-server-store no
--export-server-certs yes
--include-ca yes
--server-certs-store srv-certs.store
--export-ca-key yes
--root-private-keys-store root-ca-pk.store
--reuse-ca
--private-key-password password for CA private key
```

, new webCertificates configuration is added to the directory and standard-web-key.store is created. This store can be used for the embedded application web server.

```
        <node class="container" name="webcertificates">
          <node class="string" name="alias">
            <node-attribute name="value" value="standard"/>
          </node>
          <node class="bytes" name="certificate">
            <node-attribute name="value" value="REDACTED"/>
          </node>
          <node class="container" name="chain">
            <node class="bytes" name="standard-1">
              <node-attribute name="value" value="REDACTED"/>
            </node>
          </node>
          <node class="container" name="key">
            <node class="bytes" name="key-entry">
              <node-attribute name="value" value="REDACTED"/>
            </node>
            <node class="bytes" name="key-password">
              <node-attribute name="value" value="REDACTED"/>
          </node>
        </node>
      </node>
    </node>
```

The alias

```
        <node class="string" name="alias">
          <node-attribute name="value" value="standard"/>
        </node>
```

can be updated manually to any another server process alias and after SSLCertGenUtil has been executed, the rest webCertificates is updated accordingly and can be used to setup the web server and web clients. Also it is possible to set up this configuration manually from the existing private and public certificates from the directory.

In order to get independent Let's Encrypt certificate it needs to execute this command

```
fwd@demosrv01:/opt/app/deploy/server$ java -classpath ../lib/p2j.jar com.goldencode.p2j.security.SSLCertGenUti
l -dir directory.xml -host 192.168.1.39 -port 8443 -server "acme://letsencrypt.org/" -domains "demo.goldencode
.com" -S C="US" -S S="GA" -S L="Alpharetta" -S O="GoldenCode" -S OU="Demo" --reuse-passwords yes -acme REQUEST
```

#96 - 11/16/2017 08:35 AM - Constantin Asofiei

Sergey, this is a review for 3240c rev 11202:

1. placing the webCertificates node under the server/default is not OK. Please move it under the certificates node; as a directory.xml can have multiple servers, you can make it something like and change the SSLCertGenUtil code to specify the server name for which the web certificate should be generated.   Just to note, the webCertificates/<server-name>/alias/ value is a certificate alias which may be different than the server name (as you mentioned).   Also, you will need to change the search for the webCertificates node to use the certificates/webCertificates/<server-name>/ in SecurityCache or any other place where you search for it.

```
       <node class="container" name="webCertificates">
          <node class="container" name="<server-name>">
             <node class="string" name="alias">
               <node-attribute name="value" value="standard"/>
             </node>
             <node class="bytes" name="certificate">
               <node-attribute name="value" value="REDACTED"/>
             </node>
             <node class="container" name="chain">
               <node class="bytes" name="standard-1">
                 <node-attribute name="value" value="REDACTED"/>
               </node>
             </node>
             <node class="container" name="key">
```

```
              <node class="bytes" name="key-entry">
                <node-attribute name="value" value="REDACTED"/>
              </node>
              <node class="bytes" name="key-password">
                <node-attribute name="value" value="REDACTED"/>
              </node>
            </node>
          </node>
        </node>
      </node>
```

2. SSLCertGenUtil.saveToWebStore needs javadoc
3. SecurityCache and everything else: webCertificates (and the other web-store specific stuff) must be optional otherwise it will break with existing installations which don't require a web certificate
4. what testing did you do? Does it work 'out-of-the-box', directly running the @SSLCertGenUtil ... -acme REQUEST" to get the Let's Encrypt certificates? Do you need to run the first command in #3240-95 to prepare the directory?

#97 - 11/16/2017 09:06 AM - Sergey Ivanovskiy

Constantin Asofiei wrote:

> Sergey, this is a review for 3240c rev 11202:
> placing the webCertificates node under the server/default is not OK. Please move it under the certificates node; as a directory.xml can have multiple servers, you can make it something like and change the SSLCertGenUtil code to specify the server name for which the web certificate should be generated.   Just to note, the webCertificates/<server-name>/alias/ value is a certificate alias which may be different than the server name (as you mentioned).   Also, you will need to change the search for the webCertificates node to use the certificates/webCertificates/<server-name>/ in SecurityCache or any other place where you search for it.

Please correct me. The webCertificates must be per a server name under /security/certificates/. For default server it must be

```
<node class="container" name="webCertificates">
          <node class="container" name="default">
             <node class="string" name="alias">
               <node-attribute name="value" value="standard"/>
             </node>
             <node class="bytes" name="certificate">
               <node-attribute name="value" value="REDACTED"/>
             </node>
             <node class="container" name="chain">
               <node class="bytes" name="standard-1">
                 <node-attribute name="value" value="REDACTED"/>
               </node>
             </node>
             <node class="container" name="key">
               <node class="bytes" name="key-entry">
                 <node-attribute name="value" value="REDACTED"/>
               </node>
               <node class="bytes" name="key-password">
                 <node-attribute name="value" value="REDACTED"/>
               </node>
             </node>
           </node>
         </node>
      </node>
```

Does it make sense to change web certificates for a different server?

> SSLCertGenUtil.saveToWebStore needs javadoc

OK, planning to fix it.

> SecurityCache and everything else: webCertificates (and the other web-store specific stuff) must be optional otherwise it will break with existing installations which don't require a web certificate

It should be optional, because if web store is not set, then another settings should be used to setup its web server and web clients.

> what testing did you do? Does it work 'out-of-the-box', directly running the @SSLCertGenUtil ... -acme REQUEST" to get the Let's Encrypt certificates? Do you need to run the first command in #3240-95 to prepare the directory?

I tested rev 11202 locally with already received Let's Encrypt certificates from demo.goldencode.com. It should work to run SSLCertGenUtil ... -acme REQUEST directly. No, the first command is not required at all. The server should work without adding this new settings under /server/default/webCertificates

#98 - 11/16/2017 09:16 AM - Constantin Asofiei

Sergey Ivanovskiy wrote:

> Please correct me. The webCertificates must be per a server name under /security/certificates/. For default server it must be
> Does it make sense to change web certificates for a different server?

So, the idea here is this: SSLCertGenUtil, when generating certificates, it will ask for a server name (i.e. standard). After this, it will create a security/certificates/webCertificates/<server-name> node and place all the configuration there. There is no webCertificates/default. At runtime, you have the SecurityCache.serverId which identifies the currently running server.

> > SecurityCache and everything else: webCertificates (and the other web-store specific stuff) must be optional otherwise it will break with existing installations which don't require a web certificate

It should be optional, because if web store is not set, then another settings should be used to setup its web server and web clients.

I mean this code in SecurityCache, it assumes the webCertificates is always there; please double-check all runtime code that this is optional.

```
      try
      {
         webStore = SSLCertGenUtil.createEmptyStore();

         webServerAlias = Utils.getDirectoryNodeString(ds,
                                                       "webCertificates/alias",
                                                       "web-cert",
                                                       false);

         SSLCertFactory factory = getSSLCertFactory();

         CertificateFactory cf = CertificateFactory.getInstance("X.509");

         CertificateSuite suite = SSLCertGenUtil.readWebCertificates(ds, factory, cf);

         String keyPassword = SSLCertGenUtil.createAES256BitKey();

         byte[] encrypted = factory.encryptPrivateKey(suite.privateKey, keyPassword);

         privateKeys.put(webServerAlias, encrypted);

         privateKeyPasswords.put(webServerAlias, keyPassword);

         webStore.setKeyEntry(webServerAlias,
                              suite.privateKey,
                              keyPassword.toCharArray(),
                              suite.getFullChain());
      }
```

#99 - 11/16/2017 09:34 AM - Sergey Ivanovskiy

OK, understand. Planning to do changes accordingly.

#100 - 11/17/2017 06:28 AM - Sergey Ivanovskiy

3240c was rebased over 11201. Please review 11205 that fixed all issues found in the review #3240-96.

#101 - 11/17/2017 09:40 AM - Sergey Ivanovskiy

Constantin, the tool SSLCertGenUtil can build now the web key store that can be used for the embedded application. There is a script to start Hotel GUI embedded application and these parameters are used

```
                            -webks ../../deploy/server/embedded-private-key.store \
                            -webkspw 87ViGTQVp\<07zPw\(3i2A4%uCHYx3mIgSt\>Tk \
                            -webkepw h8jNMO\>Vbn0\`cL\)b95dVn\$w03CePIB6eItL3 \
```

How was this script created? The question is due to that passwords created by SSLCertGenUtil are only in this tool log. If this tool produces this web key store alias-web-key.store for the embedded application web server with the known passwords for its private key and its store, then its logs must be saved to store a password for CA private key and passwords for web key store. If passwords are reused from the directory, then passwords for web key store can be read from the directory since they are the same by the current implementation.

#102 - 11/20/2017 02:43 AM - Sergey Ivanovskiy

Should this branch 3240c be checked with regression tests on devsrv01? It seems that the changes are related to web server and web clients SSL configurations.

#103 - 11/20/2017 11:02 AM - Constantin Asofiei

I'm OK with the changes in 3240c rev 11205.

Sergey Ivanovskiy wrote:

> Should this branch 3240c be checked with regression tests on devsrv01? It seems that the changes are related to web server and web clients SSL configurations.

Yes, we should look for connection errors related to this. One main part run is enough, if you have doubts about the results, please notify me.

#104 - 11/20/2017 11:27 AM - Constantin Asofiei

Sergey Ivanovskiy wrote:

> Constantin, the tool SSLCertGenUtil can build now the web key store that can be used for the embedded application. There is a script to start Hotel GUI embedded application and these parameters are used
> [...]
>
> How was this script created?

It was created manually.

> The question is due to that passwords created by SSLCertGenUtil are only in this tool log.

We usually save the output to have the passwords in a nicer format. If they are saved in directory.xml, they are base64-encoded.

> If this tool produces this web key store alias-web-key.store for the embedded application web server with the known passwords for its private key and its store, then its logs must be saved to store a password for CA private key and passwords for web key store. If passwords are reused from the directory, then passwords for web key store can be read from the directory since they are the same by the current implementation.

Until we move the embedded app as a default module in FWD project, we need to explicitly set them. No other automation is needed at this time for embedded app.

So, please point me to the command which can be used to test the Let's Encrypt certificates on our demo server, I'd like to check them myself.

#105 - 11/20/2017 11:40 AM - Sergey Ivanovskiy

Constantin, please test this command #3240-80 (#3240-95) with these additional parameters --web-alias and --server-id that can be entered later

```
fwd@demosrv01:/opt/<REDACTED>/deploy/server$ java -classpath ../lib/p2j.jar com.goldencode.p2j.security.SSLCertGenUtil -dir directory.xml -host 192.168.1.39 -port 8443 -server "acme://letsencrypt.org/" -domains "demo.goldencode.com" -S C="US" -S S="GA" -S L="Alpharetta" -S O="GoldenCode" -S OU="Demo" --reuse-passwords yes -acme REQUEST --web-alias webServerAlias --server-id standard
```

#106 - 11/20/2017 12:47 PM - Constantin Asofiei

Sergey, what directory preparations are there required? I assume all Jetty web servers will be using the same alias (i.e. embedded app and virtual desktop)? I see the --web-alias webServerAlias setting but where is the webServerAlias coming from?

#107 - 11/20/2017 01:04 PM - Sergey Ivanovskiy

This command should change directory.xml and add downloaded certificates to directory.xml under /security/certificates/webCertificates/[--server-id]/ and webServerAlias will be the alias for the target downloaded certificates, where [--server-id] is substituted by --server-id value. Then this webServerAlias can be used with generated key store webServerAlias-web-key.store for the embedded application. This web alias name can be changed but if it is not given in the command line, then by default this name will be used for the web alias. Yes, it is supposed that all servers use this generated key store and the provided web alias. The key store has this pattern [--web-alias]-web-key.store, where [--web-alias] is substituted by the --web-alias value.

#108 - 11/20/2017 01:11 PM - Sergey Ivanovskiy

There are no another requirements to run the command except the P2J and embedded application servers should be stopped.

#109 - 11/20/2017 11:35 PM - Sergey Ivanovskiy

The main-regression part of the run-time tests is ready and available at devsrv01 in the results folder 20171120_160355.zip. In this run there are 6 failed tests:

```
gso_190 ...
gso_238 ...
tc_dc_slot_024 ...
tc_job_002 ...
tc_job_clock_004 ...
tc_pay_wr_inq_001 ...
```

Please look at the report results. I found that the failed tests are printed twice in the index.html main page and first time they are printed in the order listed above, but then

```
gso_238 ... gso_190 ..
tc_dc_slot_024 ...
tc_job_002 ...
tc_job_clock_004 ...
tc_pay_wr_inq_001 ...
```

Looks like a bug in the report. Now running ctrl-c tests.

#110 - 11/21/2017 04:23 AM - Sergey Ivanovskiy

The ctrl-c-regression part has been finished with these results 20171120_231820.zip, where gso_ctrlc_tests are passed and gso_ctrlc_3way_tests are failed.

#111 - 11/21/2017 08:39 AM - Constantin Asofiei

Sergey, I'm getting this on demosrv01:

```
java -classpath ../lib/p2j.jar com.goldencode.p2j.security.SSLCertGenUtil -dir directory.xml -host 192.168.1.3
9 -port 8443 -server "acme://letsencrypt.org/" -domains "demo.goldencode.com" -S C="US" -S S="GA" -S L="Alphar
etta" -S O="GoldenCode" -S OU="Demo" --reuse-passwords yes -acme REQUEST --web-alias webServerAlias --server-i
d standard
Initializing service for directory directory.xml...
Reading company configuration...
Using 'US' for [C] Country Code.
Using 'Alpharetta' for [L] Locality (city, etc).
Using 'GoldenCode' for [O] Organization.
Using 'Hotel' for [OU] Organization Unit.
Using 'GA' for [ST] State or Province Name.
Using 10 years for validity.
Using 65337 for RSA public exponent.
Adding account /security/accounts/processes/standard
Adding account /security/accounts/processes/ehotel
Adding account /security/accounts/processes/embedded
Adding account /security/accounts/users/bogus
WARNING: alias shared is set for multiple accounts!
Account /security/accounts/users/hotel ignored - no alias is defined
Account /security/accounts/users/3b8wi23f6414zmrv ignored - no alias is defined
Account /security/accounts/users/gcy8jo4ai3656o13 ignored - no alias is defined
Account /security/accounts/users/3dp61i2na56zw9w8 ignored - no alias is defined
Account /security/accounts/users/w30x6rwv64hf581s ignored - no alias is defined
Account /security/accounts/users/oc6w268mb4j3t89n ignored - no alias is defined
Account /security/accounts/users/4tt7s16wckb05j08 ignored - no alias is defined
Account /security/accounts/users/41bfjem218q8n18z ignored - no alias is defined
Account /security/accounts/users/c5qb218pde6w97m9 ignored - no alias is defined
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.slf4j.impl.JDK14LoggerFactory]
Nov 21, 2017 8:39:21 AM com.goldencode.p2j.security.AcmeClient getCertificates
INFO: The client account is https://acme-v01.api.letsencrypt.org/acme/reg/24633622
Nov 21, 2017 8:39:21 AM com.goldencode.p2j.security.AcmeClient authorize
INFO: Authorization for domain demo.goldencode.com
Nov 21, 2017 8:39:21 AM com.goldencode.p2j.security.AcmeClient tlsSniChallenge
INFO: In order to pass tls-sni-01 challenge the web server must return the certificate 'tlssni.crt' on a SNI r
equest to:
Nov 21, 2017 8:39:21 AM com.goldencode.p2j.security.AcmeClient tlsSniChallenge
INFO: 098d2d5f6f84fbf089eba40bdef08e6c.5dd4349210c4db03abd753ef4050e059.acme.invalid
Nov 21, 2017 8:39:21 AM com.goldencode.p2j.security.AcmeClient tlsSniChallenge
INFO: The matching keypair must be available at 'tlssni.key'.
Nov 21, 2017 8:39:21 AM com.goldencode.p2j.security.AcmeClient tlsSniChallenge
INFO: Please use 'tlssni.key' and 'tlssni.crt' cert for SNI requests to:

https://098d2d5f6f84fbf089eba40bdef08e6c.5dd4349210c4db03abd753ef4050e059.acme.invalid

Nov 21, 2017 8:39:21 AM com.goldencode.p2j.security.AcmeClient getCertificates
SEVERE: Failed to get a certificate for domains [demo.goldencode.com]
org.shredzone.acme4j.exception.AcmeException: Managed web server 192.168.1.39:8443 can't be started
        at com.goldencode.p2j.security.AcmeClient.acceptTlsSniChallenge(AcmeClient.java:652)
        at com.goldencode.p2j.security.AcmeClient.authorize(AcmeClient.java:527)
        at com.goldencode.p2j.security.AcmeClient.askCertificates(AcmeClient.java:216)
        at com.goldencode.p2j.security.AcmeClient.getCertificates(AcmeClient.java:338)
        at com.goldencode.p2j.security.SSLCertGenUtil.generate(SSLCertGenUtil.java:379)
        at com.goldencode.p2j.security.SSLCertGenUtil.main(SSLCertGenUtil.java:1996)
Caused by: java.lang.IndexOutOfBoundsException: Index: 0, Size: 0
        at java.util.LinkedList.checkElementIndex(LinkedList.java:555)
        at java.util.LinkedList.get(LinkedList.java:476)
        at com.goldencode.p2j.security.BCCertFactory.loadCertificateSuite(BCCertFactory.java:591)
        at com.goldencode.p2j.security.ManagedWebServer.<init>(ManagedWebServer.java:151)
        at com.goldencode.p2j.security.AcmeClient.acceptTlsSniChallenge(AcmeClient.java:646)
        ... 5 more

Exception in thread "main" com.goldencode.p2j.security.SSLCertGenException: The certificate request is failed
```

```
        at com.goldencode.p2j.security.SSLCertGenUtil.generate(SSLCertGenUtil.java:414)
        at com.goldencode.p2j.security.SSLCertGenUtil.main(SSLCertGenUtil.java:1996)
```

demosrv01 is configured with 3240c - please check and let me know when you have a successful generation with Let's Encrypt.

#112 - 11/21/2017 09:52 AM - Greg Shah

Please re-run the main regression testing. You don't have to re-do the CTRL-C tests.

#113 - 11/21/2017 10:17 AM - Sergey Ivanovskiy

Constantin Asofiei wrote:

> Sergey, I'm getting this on demosrv01:
> [...]
>
> demosrv01 is configured with 3240c - please check and let me know when you have a successful generation with Let's Encrypt.

I didn't test this version with demosrv01. According to this exception the certificate chain file is absent for unknown reasons.

#114 - 11/21/2017 10:21 AM - Sergey Ivanovskiy

Sergey Ivanovskiy wrote:

> Constantin Asofiei wrote:
>
> > Sergey, I'm getting this on demosrv01:
> > [...]
> >
> > demosrv01 is configured with 3240c - please check and let me know when you have a successful generation with Let's Encrypt.
>
> I didn't test this version with demosrv01. According to this exception the certificate chain file is absent for unknown reasons.

No, it is related to the different usage and I changed the code but didn't test it. Please check up this hot fix. (Committed revision 11206.)

```
=== modified file 'src/com/goldencode/p2j/security/BCCertFactory.java'
--- src/com/goldencode/p2j/security/BCCertFactory.java    2017-11-16 00:13:45 +0000
+++ src/com/goldencode/p2j/security/BCCertFactory.java    2017-11-21 15:21:52 +0000
@@ -588,7 +588,7 @@
              }

              // chainList can be a full chain
-             if (chainList.get(0).equals(publicKey))
+             if (!chainList.isEmpty() && chainList.get(0).equals(publicKey))
              {
                chainList.remove(0);
            }
```

#115 - 11/21/2017 10:34 AM - Sergey Ivanovskiy

Greg Shah wrote:

> Please re-run the main regression testing. You don't have to re-do the CTRL-C tests.

Yes, the next main-regression tests were done with these results 20171121_050940.zip. In this run there are different failed tests except tc_job_002 and tc_job_clock_004

```
1. 1. navigation: 1. login_to_main_menu: failure in step 2: 'timeout before the specific screen buffer became available'
2. basic_menuing: failure in step 1: 'timeout before the specific screen buffer became available (Mismatched data at line 0, column 0. Expected ' ' (0x0020 at relative Y 0, relative X 0) and found '^' (0x005E at relative Y 0, relative X 0), template: screens/navigation/syman_menu.txt.)'
3. function_browser: failure in step 1: 'timeout before the specific screen buffer became available (Mismatched data at line 0, column 0. Expected ' ' (0x0020 at relative Y 0, relative X 0) and found '^' (0x005E at relative Y 0, relative X 0), template: screens/navigation/syman_menu.txt.)'
4. user_and_dev_help: failure in step 1: 'timeout before the specific screen buffer became available (Mismatched data at line 0, column 0. Expected ' ' (0x0020 at relative Y 0, relative X 0) and found '^' (0x005E at relative Y 0, relative X 0), template: screens/navigation/syman_menu.txt.)'
5. logout_from_main_menu: failure in step 1: 'timeout before the specific screen buffer became available (Mismatched data at line 0, column 0. Expected ' ' (0x0020 at relative Y 0, relative X 0) and found '^' (0x005E at relative Y 0, relative X 0), template: screens/navigation/syman_menu.txt.)'
2. gso_221_test: 1. gso221_session1: failure in step 5: 'timeout before the specific screen buffer became available'
2. gso221_session2: failure in step 1: 'Timeout while waiting for event semaphore to be posted.'
3. tc_tests: 1. tc_dc_slot_024: failure in step 8: 'timeout before the specific screen buffer became available (Mismatched data at line 4, column 9. Expected ' ' (0x0000 at relative Y 4, relative X 9) and found '0' (0x0030 at relative Y 4, relative X 9), template: screens/tc/dc/slot/024/dc_slot_024_step1.txt.)'
2. tc_dc_slot_025: failure in step 8: 'timeout before the specific screen buffer became available (Mismatched data at line 4, column 9. Expected ' ' (0x0000 at relative Y 4, relative X 9) and found '0' (0x0030 at relative Y 4, relative X 9), template: screens/tc/dc/slot/025/dc_slot_025_step1.txt.)'
3. tc_job_002: failure in step 41: 'Line size mismatch (line # = 11, base = 0, actual = 91).'
4. tc_job_clock_001: failure in step 9: 'timeout before the specific screen buffer became available (Mismatched data at line 5, column 18. Expected ' ' (0x0000 at relative Y 5, relative X 18) and found '┌' (0x250C at relative Y 5, relative X 18), template: screens/tc/job/clock/001/job_clock_001_step3.txt.)'
5. tc_job_clock_002: failure in step 20: 'timeout before the specific screen buffer became available (Mismatched data at line 5, column 18. Expected ' ' (0x0000 at relative Y 5, relative X 18) and found '┌' (0x250C at relative Y 5, relative X 18), template: screens/tc/job/clock/002/job_clock_002_step6_2.txt.)'
6. tc_job_clock_003: dependency chain: tc_job_clock_001
7. tc_job_clock_004: failure in step 3: 'Timeout while waiting for event semaphore to be posted.'
8. tc_job_clock_005: failure in step 23: 'timeout before the specific screen buffer became available (Mismatched data at line 5, column 18. Expected ' ' (0x0000 at relative Y 5, relative X 18) and found '┌' (0x250C at relative Y 5, relative X 18), template: screens/tc/job/clock/005/job_clock_005_step7.txt.)'
9. tc_job_clock_006: dependency chain: tc_job_clock_001 --> tc_job_clock_003
10. tc_dc_slot_023: dependency chain: tc_job_clock_001 --> tc_job_clock_003 --> tc_job_clock_006navigation: 1. login_to_main_menu: failure in step 2: 'timeout before the specific screen buffer became available'
2. basic_menuing: failure in step 1: 'timeout before the specific screen buffer became available (Mismatched data at line 0, column 0. Expected ' ' (0x0020 at relative Y 0, relative X 0) and found '^' (0x005E at relative Y 0, relative X 0), template: screens/navigation/syman_menu.txt.)'
3. function_browser: failure in step 1: 'timeout before the specific screen buffer became available (Mismatched data at line 0, column 0. Expected ' ' (0x0020 at relative Y 0, relative X 0) and found '^' (0x005E at relative Y 0, relative X 0), template: screens/navigation/syman_menu.txt.)'
4. user_and_dev_help: failure in step 1: 'timeout before the specific screen buffer became available (Mismatched data at line 0, column 0. Expected ' ' (0x0020 at relative Y 0, relative X 0) and found '^' (0x005E at relative Y 0, relative X 0), template: screens/navigation/syman_menu.txt.)'
5. logout_from_main_menu: failure in step 1: 'timeout before the specific screen buffer became available (Mismatched data at line 0, column 0. Expected ' ' (0x0020 at relative Y 0, relative X 0) and found '^' (0x005E at relative Y 0, relative X 0), template: screens/navigation/syman_menu.txt.)'
2. gso_221_test: 1. gso221_session1: failure in step 5: 'timeout before the specific screen buffer became available'
2. gso221_session2: failure in step 1: 'Timeout while waiting for event semaphore to be posted.'
3. tc_tests: 1. tc_dc_slot_024: failure in step 8: 'timeout before the specific screen buffer became available (Mismatched data at line 4, column 9. Expected ' ' (0x0000 at relative Y 4, relative X 9) and found '0' (0x0030 at relative Y 4, relative X 9), template: screens/tc/dc/slot/024/dc_slot_024_step1.txt.)'
```

```
2. tc_dc_slot_025: failure in step 8: 'timeout before the specific screen buffer became available (Mismatched data at line 4, column 9. Expected ' ' (0x0000 at relative Y 4, relative X 9) and found '0' (0x0030 at relative Y 4, relative X 9), template: screens/tc/dc/slot/025/dc_slot_025_step1.txt.)'
3. tc_job_002: failure in step 41: 'Line size mismatch (line # = 11, base = 0, actual = 91).'
4. tc_job_clock_001: failure in step 9: 'timeout before the specific screen buffer became available (Mismatched data at line 5, column 18. Expected ' ' (0x0000 at relative Y 5, relative X 18) and found '┌' (0x250C at relative Y 5, relative X 18), template: screens/tc/job/clock/001/job_clock_001_step3.txt.)'
5. tc_job_clock_002: failure in step 20: 'timeout before the specific screen buffer became available (Mismatched data at line 5, column 18. Expected ' ' (0x0000 at relative Y 5, relative X 18) and found '┌' (0x250C at relative Y 5, relative X 18), template: screens/tc/job/clock/002/job_clock_002_step6_2.txt.)'
6. tc_job_clock_003: dependency chain: tc_job_clock_001
7. tc_job_clock_004: failure in step 3: 'Timeout while waiting for event semaphore to be posted.'
8. tc_job_clock_005: failure in step 23: 'timeout before the specific screen buffer became available (Mismatched data at line 5, column 18. Expected ' ' (0x0000 at relative Y 5, relative X 18) and found '┌' (0x250C at relative Y 5, relative X 18), template: screens/tc/job/clock/005/job_clock_005_step7.txt.)'
9. tc_job_clock_006: dependency chain: tc_job_clock_001 --> tc_job_clock_003
10. tc_dc_slot_023: dependency chain: tc_job_clock_001 --> tc_job_clock_003 --> tc_job_clock_006
```

#116 - 11/21/2017 10:39 AM - Sergey Ivanovskiy

Planning to run main-regression tests for rev. 11206.

#117 - 11/21/2017 10:50 AM - Constantin Asofiei

Sergey, OK, the certificate is generated and is trusted by the browser. Issues:

1. although I specified --web-alias embedded, I still got The web certificate suite [webServerAlias] was saved ... why is webServerAlias forced instead of embedded alias?
2. the O and OU certificate settings, although specified in the command, weren't saved in the certificate - you can check demosrv01 servers to see what I mean, they are up now, with the Let's Encrypt certificates.

#118 - 11/21/2017 11:32 AM - Sergey Ivanovskiy

Constantin Asofiei wrote:

> Sergey, OK, the certificate is generated and is trusted by the browser. Issues:
> 1. although I specified --web-alias embedded, I still got The web certificate suite [webServerAlias] was saved ... why is webServerAlias forced instead of embedded alias?

Constantin, thank you, committed rev 11207 should fix this issue,

```
=== modified file 'src/com/goldencode/p2j/security/SSLCertGenUtil.java'
--- src/com/goldencode/p2j/security/SSLCertGenUtil.java    2017-11-17 19:54:39 +0000
+++ src/com/goldencode/p2j/security/SSLCertGenUtil.java    2017-11-21 16:19:47 +0000
@@ -383,6 +383,11 @@
                                                              AcmeClient.USER_REG_FILE);
                   if (suite != null && suite.publicKey != null)
                   {
```

```
+                       if (inputParameters.webAlias != null)
+                       {
+                          webAlias = inputParameters.webAlias.trim();
+                       }
+
                        if (webAlias == null || webAlias.isEmpty())
                        {
                           webAlias = "webServerAlias";
```

If the provided alias is already present in the directory, then the program should ask a different name for the web certificate alias.

```
                      if (inputParameters.webAlias != null)
                      {
                         webAlias = inputParameters.webAlias.trim();
                      }

                      if (webAlias == null || webAlias.isEmpty())
                      {
                         webAlias = "webServerAlias";
                      }

                      while ((webAlias != null) && (aliases.contains(webAlias)))
                      {
                         System.out.print("The following certificates can't be used as a new web" +
                                          " certificate alias: " + (new ArrayList<>(aliases)));
                         webAlias = readLine("Enter new web certificate alias: ");
                      }
```

> 2. the O and OU certificate settings, although specified in the command, weren't saved in the certificate - you can check demosrv01 servers to see what I mean, they are up now, with the Let's Encrypt certificates.

Checked P2J self-signed certificates and found that they don't have organization and organization unit too. It seems that we set up them, correct?

#119 - 11/21/2017 11:37 AM - Sergey Ivanovskiy

May be we should check up these use cases for AcmeClient: revoke certificate, delete account, then register new account with provided contact information. I didn't test them but it seems that organization unit and organization should be set.

#120 - 11/21/2017 11:41 AM - Constantin Asofiei

Sergey Ivanovskiy wrote:

> Checked P2J self-signed certificates and found that they don't have organization and organization unit too. It seems that we set up them, correct?

The embedded certificate for i.e. Hotel GUI embedded app has the O and OU set and reported by the browser. So, are you saying that with 3240c, the self-signed certificates no longer have O and OU ?

#121 - 11/21/2017 01:42 PM - Sergey Ivanovskiy

Constantin Asofiei wrote:

> Sergey Ivanovskiy wrote:
>
>> Checked P2J self-signed certificates and found that they don't have organization and organization unit too. It seems that we set up them, correct?
>
> The embedded certificate for i.e. Hotel GUI embedded app has the O and OU set and reported by the browser. So, are you saying that with 3240c, the self-signed certificates no longer have O and OU ?

No, I don't think that with 3240c there is this bug. Constantin, could I look at domain.csr?

#122 - 11/21/2017 02:14 PM - Constantin Asofiei

Sergey Ivanovskiy wrote:

> No, I don't think that with 3240c there is this bug. Constantin, could I look at domain.csr?

Yes, go ahead and use demosrv01 - everything I generated is there.

#123 - 11/21/2017 02:56 PM - Sergey Ivanovskiy

domain.csr has all required fields

```
sbi@demosrv01:/opt/app/deploy/server$ openssl req -noout -text -in domain.csr 
Certificate Request:
    Data:
        Version: 0 (0x0)
        Subject: C=US, L=Alpharetta, O=GoldenCode, OU=Demo, ST=GA, CN=demo.goldencode.com
.................................................................................
```

But the signed certificate domain.crt has only these fields

```
sbi@demosrv01:/opt/app/deploy/server$ openssl x509 -in domain.crt -text -noout
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number:
            03:70:fb:d0:2f:24:bb:e0:7d:5f:5e:60:9f:9a:06:2e:dd:d3
    Signature Algorithm: sha256WithRSAEncryption
        Issuer: C=US, O=Let's Encrypt, CN=Let's Encrypt Authority X3
        Validity
            Not Before: Nov 21 14:43:01 2017 GMT
            Not After : Feb 19 14:43:01 2018 GMT
        Subject: CN=demo.goldencode.com
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
                Public-Key: (2048 bit)
                Modulus:
                    00:b0:21:32:7c:6f:af:c8:32:a9:48:08:f8:10:32:
                    2a:bf:8a:ce:b5:4c:7c:fe:c1:aa:a6:08:eb:31:c8:
                    c2:d0:f2:97:c0:5a:84:a6:cf:4d:3d:92:34:90:ed:
                    e9:90:a6:21:f1:9a:e1:af:36:56:1b:19:ee:99:0e:
                    be:ef:57:8b:fb:02:a5:05:d2:88:f2:fc:79:d5:76:
                    7b:d0:d8:0c:7b:0b:53:4c:8b:95:2d:04:c5:cb:2b:
                    77:e3:9d:03:b7:45:79:b0:24:34:18:2c:90:40:ed:
                    38:0b:ea:66:93:a1:de:b5:dd:d1:c9:cf:f0:57:ab:
                    66:b2:23:0f:2c:c3:67:9d:a0:f8:f0:37:66:6a:cd:
                    eb:ea:1b:5d:1f:a7:d7:65:93:e4:a6:10:aa:fe:3e:
                    ee:be:ea:ce:79:9d:63:20:fc:4e:13:70:96:12:57:
                    ba:e5:81:4e:db:64:c4:db:46:3e:b6:0d:aa:02:f7:
                    19:3d:c8:ed:e5:70:2e:e8:94:0b:23:f8:e5:e0:d5:
                    12:52:57:82:5b:1c:d4:da:10:6e:6c:a5:ad:64:70:
                    ad:fc:db:d0:54:5c:fc:66:03:ee:4f:90:bb:8b:68:
                    5d:79:89:83:83:81:af:04:7e:c9:4d:74:82:43:44:
                    fd:27:0f:7f:30:ad:5f:67:d9:30:14:e5:16:67:3f:
                    46:3d
                Exponent: 65537 (0x10001)
        X509v3 extensions:
            X509v3 Key Usage: critical
                Digital Signature, Key Encipherment
            X509v3 Extended Key Usage: 
                TLS Web Server Authentication, TLS Web Client Authentication
            X509v3 Basic Constraints: critical
                CA:FALSE
            X509v3 Subject Key Identifier: 
                E4:7D:BD:FA:2B:72:35:0A:BA:44:57:B2:C7:09:92:2B:1D:0F:B5:1B
            X509v3 Authority Key Identifier: 
                keyid:A8:4A:6A:63:04:7D:DD:BA:E6:D1:39:B7:A6:45:65:EF:F3:A8:EC:A1

            Authority Information Access: 
                OCSP - URI:http://ocsp.int-x3.letsencrypt.org
                CA Issuers - URI:http://cert.int-x3.letsencrypt.org/

            X509v3 Subject Alternative Name: 
                DNS:demo.goldencode.com
            X509v3 Certificate Policies: 
                Policy: 2.23.140.1.2.1
                Policy: 1.3.6.1.4.1.44947.1.1.1
                  CPS: http://cps.letsencrypt.org
                  User Notice:
                    Explicit Text: This Certificate may only be relied upon by Relying Parties and only in acc
```

```
ordance with the Certificate Policy found at https://letsencrypt.org/repository/

    Signature Algorithm: sha256WithRSAEncryption
         64:9f:ae:a6:c7:6f:90:bd:e5:ac:af:19:b5:03:82:da:65:a9:
         c2:6b:eb:aa:8a:c0:ec:53:5b:78:ea:03:81:21:23:02:c8:a0:
         85:d8:ef:92:cf:10:d0:07:8d:53:c6:14:4a:0c:4a:1b:b1:6c:
         81:2d:b0:9b:30:9b:dd:50:39:8f:30:65:d3:dd:60:00:53:dd:
         69:7a:54:f6:15:9f:63:c0:33:1d:e4:0d:b7:4b:c0:b2:fc:dd:
         2a:2a:67:f2:18:b0:bf:67:fb:9c:8c:d6:ba:9b:58:37:48:af:
         aa:88:21:c7:53:57:7b:c7:4c:90:ef:98:72:fc:a0:8f:b6:c9:
         f5:61:72:87:74:98:39:bf:e6:92:e2:23:46:cb:21:f1:ee:e1:
         20:a7:bd:33:d0:b5:07:da:16:05:c5:d1:d8:bd:2b:95:b5:bc:
         3a:b3:be:83:9f:89:9a:19:5f:97:09:1a:9e:46:4c:7d:3b:ed:
         66:7e:aa:82:bf:49:67:db:b5:cb:24:e5:0c:a0:e7:3f:f4:dd:
         aa:c5:d2:ef:27:34:76:9e:20:27:21:d0:1a:9d:c8:81:44:a6:
         45:ca:b4:7f:78:81:51:c9:18:0c:a4:07:d1:a1:32:6b:ca:f6:
         ab:a7:58:35:fa:44:65:49:b9:08:c9:c7:dd:76:21:17:83:a4:
         91:f4:aa:08
```

It looks that these fields weren't taken into account C=US, L=Alpharetta, O=GoldenCode, OU=Demo. May be the user certificates have been taken into account. They were generated by this method KeyPair loadOrCreateKeyPair(File file) of AcmeClient?

#124 - 11/21/2017 03:00 PM - Sergey Ivanovskiy

I checked that the main web site www.goldencode.com has these target fields unset too.

#125 - 11/21/2017 03:13 PM - Sergey Ivanovskiy

Found the following unrelated issue that if the directory doesn't have a company configuration, then generated certificates have no company configuration too. Please review committed revision 11208 that fixed this bug and improved to parse company configuration parameters into EnumMap.

#126 - 11/21/2017 04:23 PM - Sergey Ivanovskiy

The last main-regression tests were failed without writing reports. Planning to restart for rev 11208.

#127 - 11/21/2017 05:19 PM - Greg Shah

> In this run there are different failed tests except tc_job_002 and tc_job_clock_004

I think this is enough to tell us it is OK. You don't have to do another run.

When Constantin believes this is complete, you can merge to trunk.

#128 - 11/21/2017 05:45 PM - Sergey Ivanovskiy

OK, stopped this testing.

#129 - 11/22/2017 08:24 AM - Constantin Asofiei

Sergey, looks like I'm not allowed to use embedded alias - is it because there is a process using the same certificate alias? What are the constraints?

```
The following certificates can't be used as a new web certificate alias: [ehotel, embedded, shared, standard]E
nter new web certificate alias: webCertificatesAlias
```

Otherwise, I'm good with the changes, I've checked both self-signed and Let's Encrypt and they work.

#130 - 11/22/2017 10:56 AM - Sergey Ivanovskiy

I did this constraint in order that this new web alias has a unique name within the directory. If the web alias is from this set, then SSLCertGenUtil.this.updateWebCertificates reuses the existing certificates from the directory and saves them in the web key store. It seems that this constraint that aliases are unique is reasonable.

#131 - 11/22/2017 01:18 PM - Sergey Ivanovskiy

Can I merge 3240c into the trunk?

#132 - 11/23/2017 06:00 AM - Sergey Ivanovskiy

Sergey Ivanovskiy wrote:

> Can I merge 3240c into the trunk?

3240c was rebased over trunc rev. 11202. The current rev 11210 is ready to merge into the trunc.

#133 - 11/24/2017 06:10 AM - Greg Shah

Please merge 3240c to trunk.

#134 - 11/24/2017 06:35 AM - Sergey Ivanovskiy

3240c was merged into the trunc rev. 11203.

#135 - 09/11/2021 07:05 AM - Constantin Asofiei

- Related to Bug #5663: allow webcertificates details to be specified via files added

Files

3240b_1.txt	35.5 KB	10/31/2017	Sergey Ivanovskiy
LetsEncryptCerts.png	98.2 KB	11/14/2017	Sergey Ivanovskiy