DataONE Certificate Authority

March 18, 2015

Contents

1 Overview

This directory contains configuration files and notes on how to set up the DataONE Certificate Authority using OpenSSL as the CA application. OpenSSL generates all files needed for the CA, including certificate requests, keys, certificates, and certificate revocation lists.

The DataONE Certificate Authority is governed by a Root CA, which delegates all certificate signing and management to a Production CA. The private key for the Root CA is offline and completely protected, which protects the CA should somehow the Production CA private key be compromised. This document shows the steps used to create both the Root CA and the Production CA, as well as to perform common options such as creation and revocation of certificates with the Production CA. The operations have been encapsulated in the `ca` shell script.

There is also a Test CA, used for generating certificates for servers in the dev, staging, and sandbox environments, but this CA is completely independent of the Production CA and its certificates will not be accepted in the production environments. The Test CA has the same hierarchy as the Production CA, with a Root CA delegating certificate signing and management to an intermediate CA.

2 Key Security

The private keys associated with the DataONE certificate authorities are not stored online, nor are they made available on the network. To use the keys, it is necessary to have access to the physical media on which the keys reside, which is an encrypted volume on the device. TrueCrypt was formerly used to provide the encrypted volume, however that software is no longer considered secure and should not be used. Instead, an encrypted .DMG file should be used as the encrypted volume, using the same password as was used for the TrueCrypt volume.

New certificates created using the CA have two components: the certificate and the key. The certificate can be publicly exposed, and should be added to svn and checked in. The key MUST be kept private. A compromised key must be revoked and a replacement issued.

3 Requirements

The CA scripts rely on the BASH shell and are developed on OS X (bash 3.2.53) though should work without modification on Linux. Dependencies are:

- OpenSSL
- Standard command line tools such as `sed`, `awk`, `cut`, `sort`
- XMLStarlet is used by `cert_status` to parse SVN blame response
- Subversion client

4 Installation

Installing the DataONE CA involves the following steps. In these instructions, it is assumed that the CA software is being installed in ${HOME}/Projects/DataONE/tools, identified by ${CA_HOME} in the examples. Adjust as appropriate for your system.

1. The CA is distributed from subversion. Checkout the tool to the desired location:

```
cd ${HOME}/Projects/DataONE/tools
svn co https://repository.dataone.org/software/tools/trunk/ca
export CA_HOME="$(pwd)/ca"
```

2. Create a symbolic link for `/var/ca` to the checkout location:

   ```
   sudo ln -s ${CA_HOME}/ca /var/ca
   ```

3. Mount the encrypted volume using Finder or the command line when ready to create certificates or update the revocation list:

   ```
   hdiutil attach -agentpass /path/to/encrypted/DMG
   ```

   note

   Do not keep the encrypted volume with the keys on your laptop or any other device that is regularly connected to the Internet. Keep it on a USB stick or some other physical media that can be disconnected.

Verify the installation by running the `cert_status` script:

```
cd ${CA_HOME}
./cert_status
```

If all is good, then the script will examine the contents of the Test Environment certificates folder and report on the status of each .pem file found there. The output is lengthy, and looks something like:


```
{"what":"Certificate Status",
 "Generated":"2015-03-18T11:23:26.000+00:00",
 "content":[
{
 "File_name"   : "/Users/vieglais/Projects/DataONE_61385/svn/software/tools/trunk/ca/DataONETestIntCA/certs/cr
 "Author"      : "",
 "Serial"      : "DA3263A2A12D004B",
 "DN"          : "DC=org,DC=dataone,DC=test,CN=cn-dev-orc-1",
 "Created"     : "2012-07-24T03:39:45.000+00:00",
 "Expires"     : "2015-07-24T03:39:45.000+00:00",
 "Expires_days": 127,
 "Revocation"  : "Revoked",
 "Validity"    : "Valid"
}
,
{
 "File_name"   : "/Users/vieglais/Projects/DataONE_61385/svn/software/tools/trunk/ca/DataONETestIntCA/certs/cr
 "Author"      : "",
 "Serial"      : "DA3263A2A12D0055",
 "DN"          : "DC=org,DC=dataone,CN=cn-dev-orc-1.test.dataone.org",
 "Created"     : "2012-07-27T21:25:34.000+00:00",
 "Expires"     : "2015-07-27T21:25:34.000+00:00",
 "Expires_days": 131,
 "Revocation"  : "Revoked",
 "Validity"    : "Valid"
}
...
```


and so on.

5 Use

Four shell scripts are included to assist with certificate management:

ca This is the main script for creating and revoking certificates.

cert_status

This script reports the status for a single certificate or all certificates in an environment.

publish_crl

Can be used to publish the certificate revocation list to the CRL servers.

publish_cert

Provides a convenient mechanism for packaging a certificate and key and placing them in a secure location for download by an authenticated user.

5.1 ca

The shell program `ca` can be used to manage certificates from both the Test CA and the Production CA. It determines which CA to use based on commandline arguments. Type `./ca -h` to see the usage help for the `ca` utility.

To install the DataONE certificate authority, simply:

1) install openssl on your machine
2) Check out a working copy of the CA from the DataONE SVN repository
3) Mount the private key encrypted volume under /Volumes/DataONE

The `ca` utility can create, revoke, and display certificates, and can generate the Certificate Revocation List (CRL) for either of the CAs. Examples follow:

To create a Production certificate for the MN with nodeid "KNB":

```
./ca -c Prod urn:node:KNB
```

To display a Production certificate for the MN with nodeid "KNB":

```
./ca -d Prod urn:node:KNB
```

To revoke a Production certificate for the MN with nodeid "KNB":

```
./ca -r Prod urn:node:KNB
```

To generate a CRL for the Prod CA:

```
./ca -g Prod
```

Any of these commands can be made to work on the Test CA instead by switching `Prod` to `Test`.

Once new CSRs, Certificates, and CRLs have been generated, they should be added to SVN and all modified files should be checked in to SVN so that others managing the CA can access all of the updated content. The only exception are the private keys that are generated, which should be given to the MN operator along with instructions on how to protect the private key. The private key should be deleted from the CA to avoid possible exposure of the keys.

5.2 cert_status

The script `cert_status` provides a mechanism to report on the status of a single certificate or all certificates within the Production or Test environments. Report output is in JSON or pipe (|) separated values and includes the attributes:

File_name Full path to the certificate

Author The name of the subversion user that chcked in the certificate

Serial The certificate serial number

DN The certificate Distinguished Name

Created Indicates when the certificate was created

Expires Indicates when the certificate will expire

Expires_days Number of days until the expiration date

Revocation Indicates if the certificate appears in the revocation list

Validity Indicates if the test `openssl verify` passes.

`cert_status` can also be used to generate VCalendar .ics files, one for Producation and one for the Test environment, that includes dates for certificate and revocation list expiry. These are checked in to Subversion and can be subscribed to using Google Calendar or iCal using the calendar locations of:

https://repository.dataone.org/software/tools/trunk/ca/Prod_events.ics

for the Production environment, and:

https://repository.dataone.org/software/tools/trunk/ca/Test_events.ics

for the Test environment.

Example Show status of a single certificate in test environment:

```
./cert_status -A DataONETestIntCA/certs/urn\:node\:mnTestGulfWatch.pem
```

Example Show status of a single certificate in production environment, using the default locations for certificates and CRL:

```
./cert_status -A -P DataONEProdCA/certs/urn\:node\:GULFWATCH.pem
```

Example Show status of a single certificate in production environment, explicitly indicating which certificates and CRL to use:

```
./cert_status -A -r DataONEProdCA/crl/DataONEProdCA_CRL.pem \
  -a DataONEProdCA/certs/DataONEProdCA.pem \
  -c DataONERootCA/certs/DataONERootCA.pem \
  DataONEProdCA/certs/urn\:node\:GULFWATCH.pem
```

Example Generate a pipe delimited text file reporting on all the test certificates:

```
./cert_status -S > testcerts.csv; \
for f in $(find DataONETestIntCA/certs -name *.pem); \
  do ./cert_status -A -s $f >> testcerts.csv; done
```

or:

```
./cert_status -s -A DataONETestIntCA/certs
```

Example Generate a pipe delimited text file reporting on all the production certificates:

```
./cert_status -H > testcerts.csv; \
for f in $(find DataONETestIntCA/certs -name *.pem); \
do ./cert_status -A -s \
 -r DataONEProdCA/crl/DataONEProdCA_CRL.pem \
 -a DataONEProdCA/certs/DataONEProdCA.pem \
 -c DataONERootCA/certs/DataONERootCA.pem \
$f >> prodcerts.csv; done
```

or:

```
./cert_status -s -A -P DataONEProdCA/certs
```

Example Generate a calendar of events in .ics format for production environment certificate expirations and the next update time for the CRL. Output is to the file "Prod_events.ics" for the production environment or "Test_events.ics" for the test environment. The calendar can be subscribed to using the respective SVN URL:

```
./cert_status -P -L
```

5.3 publish_crl

The certificate revocation list (CRL) is a signed document that contains a list of certificates that have been revoked. The CRL has a relatively short life (typically 30 days) and MUST be updated regularly even if no more certificates have been revoked. The CRL is updated using the `ca` tool:

```
./ca -g Prod
```

for the Production environment, and:

```
./ca -g Test
```

for the Test environment.

After generation, the CRL must be uploaded to the locations specified within the certificates. Since the CRL publish locations can change over time, it is necessary to examine every certificate to ensure that the complete list of CRL locations is determined. The `publish_crl` script simplifies this task by examining the advertised CRL locations in each certificate and publishing the CRL to each expected location.

`publish_crl` uses scp to copy the CRL to each host, hence it is necessary for the user to have SSH access to the host, and write access to the file system folder where the CRL is located (`/var/www/crl`).

Example Publish the CRL for the Test Environment:

```
./publish_crl
```

Example Show what will happen when run for Production Environment:

```
./publish_crl -D -P
```

Example Publish the CRL for the Production Environment, and be verbose:

```
./publish_crl -V -P
```

5.4 publish_cert

The script `publish_cert` provides a convenience mechanism to package a certificate, its key, and the CSR used to generate the certificate into a .zip file and upload it to the distribution server (currently https://project.dataone.org/).

The script accepts two arguments, the LDAP uid of the user that will retrieve the package and the path to the certificate. The certificate is expected to be located in the `certs` folder of the respective CA.

> **note**
>
> The resulting file names have the ":" character replaced with "_".

The script uses ssh to connect to the distribution host, create a target folder if necessary, and upload the package .zip file. As such, it is necessary for the user running the script to have SSH access to the distribution host and write access to the destination folder (`/var/www/users`).

Example Share a certificate and key for user vieglais:

```
./publish_cert vieglais DataONETestIntCA/certs/urn:node:ATestCert.pem
```

The resulting package would be downloadable from:

```
https://project.dataone.org/~vieglais/urn_node_ATestCert.zip
```

After unzipping, the result would be:

```
urn_node_ATestCert/
  info.txt
  urn_node_ATestCert.pem
  urn_node_ATestCert.csr
  private/
    urn_node_ATestCert.key
```

The file `info.txt` contains general information about the certificate generated by the cert_status program.

6 Appendix: Additional notes on OpenSSL setup and usage

OpenSSL was used to create the various CA files and operate the CA. The following sections are a synopsis of how all of the CAs were created and how various CA functions can be run using OpenSSL alone. The `ca` shell script automates most of these functions, so their inclusion here is mainly as a reference and not intended for typical usage.

6.1 New DN formats

CA:

```
DC=org, DC=dataone, CN=DataONE Root CA
DC=org, DC=dataone, CN=DataONE Production CA
DC=org, DC=dataone, CN=DataONE Test CA
```

Nodes:

```
DC=org, DC=dataone, CN=urn:node:SOMENODE
```

CA Certificate validity:

```
100 years
```

Node Certificate validity:

```
3 years
```

6.2 Creating the Root CA

```
mkdir /var/ca
cd /var/ca
mkdir DataONERootCA
cd DataONERootCA
mkdir certs crl newcerts private req
touch index.txt
# Edit the openssl.cnf config file
openssl req -new -newkey rsa:4096 -keyout /Volumes/DataONE/DataONERootCA.key \
  -out req/DataONERootCA.csr -config ./openssl.cnf
openssl ca -create_serial -out certs/DataONERootCA.pem -days 36500 \
  -keyfile /Volumes/DataONE/DataONERootCA.key -selfsign -config ./openssl.cnf \
  -extensions v3_ca -infiles req/DataONERootCA.csr
cp serial crlnumber
# Edit crlnumber to be a different hex number
openssl ca -config ./openssl.cnf -gencrl -out crl/DataONERootCA_CRL.pem
```

6.3 Creating the Production CA

```
cd ..
mkdir DataONEProdCA
cd DataONEProdCA
mkdir certs crl newcerts private req
touch index.txt
#  Edit openssl.cnf
openssl req -new -newkey rsa:4096 -keyout /Volumes/DataONE/DataONEProdCA.key \
  -out req/DataONEProdCA.csr -config ../DataONERootCA/openssl.cnf
cd ../DataONERootCA
openssl ca -out ../DataONEProdCA/certs/DataONEProdCA.pem -days 36500 \
  -keyfile /Volumes/DataONE/DataONERootCA.key -config ./openssl.cnf \
  -extensions v3_ca -infiles ../DataONEProdCA/req/DataONEProdCA.csr
```

6.4 Create the Certificate Chain File

```
cd ..
cat DataONERootCA/certs/DataONERootCA.pem \
  DataONEProdCA/certs/DataONEProdCA.pem > DataONECAChain.crt
```

6.5 Creating and Signing Node Requests

```
cd DataONEProdCA
openssl genrsa -passout pass:temp -des3 -out private/NodeNPass.key 2048
openssl rsa -passin pass:temp -in private/NodeNPass.key -out private/NodeN.key
rm private/NodeNPass.key
openssl req -config ./openssl.cnf -new -key private/NodeNPass.key -out req/NodeN.csr
openssl ca -config ./openssl.cnf  -create_serial -days 1095 \
  -out certs/NodeN.pem -infiles req/NodeN.csr
```

6.6 To revoke a certificate

```
openssl ca -config ./openssl.cnf -revoke certs/NodeN.pem
openssl ca -config ./openssl.cnf -gencrl -out crl/DataONEProdCA_CRL.pem
```

6.7 Creating the Test CA

```
mkdir /var/ca
cd /var/ca
mkdir DataONETestCA
cd DataONETestCA
mkdir certs crl newcerts private req
touch index.txt
  # Edit the openssl.cnf config file
openssl req -new -newkey rsa:4096 -keyout /Volumes/DataONE/DataONETestCA.key \
  -out req/DataONETestCA.csr -config ./openssl.cnf
openssl ca -create_serial -out certs/DataONETestCA.pem -days 36500 \
  -keyfile /Volumes/DataONE/DataONETestCA.key -selfsign -config ./openssl.cnf \
  -extensions v3_ca -infiles req/DataONETestCA.csr
cp serial crlnumber
# Edit crlnumber to be a different hex number
openssl ca -config ./openssl.cnf -gencrl -out crl/DataONETestCA_CRL.pem
```

6.8 Creating the Test Intermediate CA

This is the equivalent of the Production CA except for the test environments:

```
cd /var/ca
mkdir DataONETestIntCA
cd DataONETestIntCA
mkdir certs crl newcerts private req
touch index.txt
# Edit the openssl.cnf config file
openssl req -new -newkey rsa:4096  -keyout /opt/DataONE/DataONETestIntCA.key \
```

```
  -out req/DataONETestIntCA.csr -config ../DataONETestCA/openssl.cnf
cd ../DataONETestCA
openssl ca -out ../DataONETestIntCA/certs/DataONETestIntCA.pem -days 36500 \
  -keyfile /opt/DataONE/DataONETestCA.key -config ./openssl.cnf \
  -extensions v3_ca  -verbose -infiles ../DataONETestIntCA/req/DataONETestIntCA.csr
# Create DataONETestIntCA/serial with serial number of the DataONETestIntCA.pem + something
```

6.9 Creating the Test Certificate Chain File

```
cd /var/ca
cat DataONETestCA/certs/DataONETestCA.pem \
  DataONETestIntCA/certs/DataONETestIntCA.pem > DataONETestCAChain.crt
```