Ruby and iControl: Automating Private Key and Certificate Signing Request Creation

George Watkins, 2011-01-04

Overview

If you’re like most of our customers, you’re probably offloading SSL on your BIG-IPs. There is a large performance benefit to this PKI management strategy, but there is also the simplicity of being able to create, update, and manage SSL certificates at one strategic point of control. Using Ruby and iControl, we can take this one step further by allowing creation or updating of private keys and CSRs (certificate signing request) from one’s workstation. This can be a great time saver for the administrator who is generating lots of keys and signing certificates

Generating Private Keys and CSRs

The heart of this application is the key_generate method in the Management::KeyCertificate interface. The key_generate method is used to generate private keys as well as optionally create an associated CSR. In order to make it work, it will need five arguments: mode, keys, x509_data, create_optional_cert_csr, and overwrite. The ‘mode’ parameter designates whether this key will be used in an SSL profile, for the management UI, Enterprise Manager, or iQuery. In this case, we will be using the ‘MANAGEMENT_MODE_DEFAULT’ mode, which is used to create keys, certificates, CSRs for use in SSL profiles. Secondly, we’ll provide an array of keys to generate. The ‘Key’ structure provides a number of attributes such as key type, bit length, passphrase protection, etc. The next parameter, which is associated with each key via the array offset, is the X.509 subject data. The ‘X509Data’ structure provides subject values for the key and CSR such as location and organization information and common names associated with the key.

Let’s take a look at a sample implementation for Example Company, Inc. located in San Francisco:

```
key_data  = { "id" => "www.example.com",
              "key_type" => "KTYPE_RSA_PRIVATE", \
              "bit_length" => 2048, \
              "security" => "STYPE_NORMAL", }

x509_data = { "common_name" => "www.example.com", \
              "organization_name" => "Example Company, Inc.", \
              "division_name" => "Information Technology", \
              "locality_name" => "San Francisco", \
              "state_name" => "California", \
              "country_name" => "US" }

bigip["Management.KeyCertificate"].key_generate('MANAGEMENT_MODE_DEFAULT', [key_data], [x509_data], t
```

The result of this method call will be a private key file named ‘www.example.com.key’ placed in the /config/ssl/ssl.key/ directory on the target BIG-IP. A CSR file named “www.example.com.csr” will also be placed in /config/ssl/ssl.csr/.

Retrieving the CSR

In order to have as little interaction with the BIG-IPs UI as possible, we will also need to retrieve the CSR and output it to STDOUT or a file for our administrator. This is done through another method, certificate_request_export_to_pem, which is in the same Management::KeyCertificate interface mentioned earlier. This is a simple method with two parameters: mode and csr_ids. The mode will be the same as earlier (MANAGEMENT_MODE_DEFAULT) and the CSR ID will be the same as the ID we used when we created the private key. Making the following method call will result in the base-64 encoded CSR being returned:

```
csr = bigip["Management.KeyCertificate"].certificate_request_export_to_pem('MANAGEMENT_MODE_DEFAULT',
```

The Application

Now if we take those concepts and wrap them up into a pretty application, we arrive at a tool that can be given to any BIG-IP administrator. The script requires a minimum of three command-line options: BIG-IP address, password, and a key ID. The rest of the values will either be taken from defaults (RSA key type, 2048-bit keys, no passphrase, etc.). In the case of the X.509 subject data, the user will be prompted for responses if they are not provided. Finally, we can provide an option to write the private key and CSR to local files. This can be handy if you are storing the keys in a configuration management system or elsewhere. For a full list of all available options see below:

```
ssl-key-and-csr-creator.rb -b <BIG-IP address> -u <BIG-IP user> -i <key ID>

BIG-IP connection parameters
--------------------
  -b  (--bigip-address)  BIG-IP management-accessible address
  -u  (--bigip-user)     BIG-IP username
  -p  (--bigip-pass)     BIG-IP password (will prompt if left blank

Private key parameters
--------------------
  -i  (--key-id)         key ID: must be unique and should be indicative of the purpose (required)
  -t  (--key-type)       key type: [RSA|DSA] (default is 'RSA')
  -l  (--key-bit-length) key bit length: should be a minimum of 1024-bit (default is 2048; most CAs
  -s  (--key-security)   key security: [normal|fips|password] (default is 'normal' with no passphras

X.509 data parameters (if blank, you'll be prompted for the answers)
--------------------
      (--common-name)    common name: FQDN for virtual server (www.example.com)
      (--country)        country: two letter country abbreviation (US, CN, etc.)
      (--state)          state: two letter state abbreviation (WA, OR, CA, etc.)
      (--locality)       locality: locality or city name (Seattle, Portland, etc.)
      (--organization)   organization: organization or company name (F5 Networks, Company XYZ, etc.)
      (--division)       division: department or division name (IT, HR, Finance, etc.)

Output options
--------------------
  -o  (--output-dir)     CSR/key output directory: location to output private key and CSR files (def
  -k  (--key-output)     key output: save private key to a local file (saved as key_id.key)
  -c  (--csr-output)     CSR output: save certificate signing request to a local file (saved as key_

Help and usage
--------------------
  -h  (--help)           shows this help/usage dialog
```

F5 Networks, Inc. | 401 Elliot Avenue West, Seattle, WA 98119 | 888-882-4447 | f5.com

F5 Networks, Inc. Corporate Headquarters info@f5.com	F5 Networks Asia-Pacific apacinfo@f5.com	F5 Networks Ltd. Europe/Middle-East/Africa emeainfo@f5.com	F5 Networks Japan K.K. f5j-info@f5.com