Building an OpenSSL Certificate Authority - Introduction and Design Considerations for Elliptical Curves

Chase Abbott, 2017-06-11

Introduction

Building a certificate authority (CA) for your lab environment is a pain. Until recently I relied on Jamielinux's OpenSSL Certificate Authority but it slowly lost relevance due to evolutions in Public Key Infrastructure (PKI) requirements, specifically ECC, hash, critical certificate extensions, and revokation changes. This guide adheres to current PKI needs namely RFC 5759 NSA's Suite B Certificate and Certificate Revocation List (CRL) Profile. Before you start burning effigies in the comment section, yes I know ECC curves NIST P-256 and NIST P-384 are not considered "secure" by SafeCurves related to rigidity[1], ladder[2], completeness[3], and indistinguishability[4]. Modern PKI ends up being a slightly religious debate divided across admins who practice provable security versus evangelists preaching mathematically absolute cryptography. I err towards the side of practical and provable security. If we adjusted cryptological practices to comply with every cryptographic sermon, we'd have nothing left to provide our infrastructure that a reasonable number of clients could connect to; forget if my infrastructure can even support it.

What Suite B does offer is compatibility with browsers and applications. Complying with popular standards allows us to learn about some of the more overlooked PKI requirements mentioned above. You don't need to fully understand the mathematical intricacies of elliptical curves but you do need to know what your applications and clients can use, especially when certain internet entities quietly drop support for unpopular curves "cough.... Chrome". So many caveats to PKI raise a question of why we're using OpenSSL and why are we building a CA compliant to NSA Suite B (specifically NIST P-384).

- OpenSSL is free and used by a majority of OS and application vendors. Many admins have familiarity with it's various commands
- My lab is in AWS and their provided Active Directory offerings do not allow us Enterprise Admin roles; a requirement for setting up Microsoft Certificate Authorities and I am not going to spin up independent AD infrastructure for isolated VPCs
- We learn OpenSSL commands related to elliptical curves (some of the many)
- We must pay attention to certificate extension attributes
- We must create a revocation policy which many existing guides gloss over
- We must pay attention to OpenSSL's caveat's and limitations to modern PKI requirements, including version support for various features.

I tested this series against OpenSSL v1.0.2g (default on Ubuntu 16.04 LTS), and v1.1.0f (manual upgrade to OpenSSL current release). There are some featuers within OpenSSL I am eagerly waiting but are not available as of the current public release. I will keep updating this as requirements and feature become mainstream. This guide is not intended for the hemmoraging edge admin who lives a brazen alpha code lifestyle.

Suite B PKI Requirements and CA Design Considerations

This guide assists the reader to build a ***lab-ready*** CA using elliptical curves for cryptographic signatures and hash functions. RFC 5759 states:

- Every Suite B cert MUST use the x.509 v3 format and contain
  - An ECDSA-capable signing key, using curves P-256 or P-384 OR
  - An ECDH-capable key establishment key, using curve P-256 or P-384
- Every Suite B certificate and CRL MUST be signed using ECDSA.
- The CA's (Root and Intermediary) MUST be on the curve P-256 or P-384 if the CA contains a key on the curve P-256

256.

- If the certificate contains a key on the curve P-384, the signing CA's Key MUST be on the curve **P-38**4.
- Any certificate and CRL MUST be hashed using SHA-256 or SHA-384, matched to the size of the signing CA's key.
- Suite B PKI follows RFC5280 for marking extensions as critical and non critical. Microsoft helps distill the extension requirements to:
    - subjectKeyIdntifier (SKI), keyUsage, and basicConstraints MUST exist
    - keyUsage MUST be marked as critical
    - keyCertSign and CRLSign MUST be set
    - All other bits (with exceptions for digital signature and non-repudiation) MUST NOT BE SET

This guide uses a common and recommended best practiced two-tier PKI hierarchy certificate authority. A two-tier certificate authority consists of:

- A self-signed root certificate
- An intermediary certificate signed by the root (completing the two tiers of the certificate authority)
- A CRL distribution point and OCSP Resolver URI for the intermediary and subsequent client & server certificates (allowing the root to revoke an intermediary certificate)
- Client and server certificates signed by the intermediary certificate

This guide is also designed for lab environments because it do not follow recommended security practices for root CA protection or private key protection. More sensitive or public CA's should follow recommended seucirty practices by storing root certificate and key offline or in an IT security-owned HSL (both recommended options for certificate and key protection). Using a dedicated lab CA reduce the exposure of corporate internal resources and limits risk impact to a small subset of machines and should not comprimise sensitive data. As you build CA's in other environments like staging or test environments where corporate data may reside, you should increase the security accordingly.

Let's go build a CA!

Notes:

(1) NIST P-384 is not considered rigid, instead considered manipulatable. Coefficients generated by hashing the unexplained seed a335926a a319a27a 1d00896a 6773a482 7acdac73. The curve-generation process has a large unexplained input, giving the curve generator a large space of curves to choose from. Consider, for example, a curve-generation process that takes y^2=x^3-3x+H(s) meeting various security criteria, where s is a large random "seed" and H is a hash function. No matter how strong H is, a malicious curve generator can search through many choices of s, checking each y^2=x^3-3x+H(s) for vulnerability to a secret attack; this works if the secret attack applies to (e.g.) one curve in a billion.

(2) NIST P-384 does not apply ladder algorithms (in this case the Montgomery ladder) to compute variations in fixed time, preventing timing and/or power information leakage (side-channel attack). SafeCurves requires curves to support simple, fast, constant-time single-coordinate single-scalar multiplication, avoiding conflicts between simplicity, efficiency, and security. This is not a requirement specifically to use Montgomery curves: there are other types of curves that support simple, fast, constant-time ladders. "Fast" means that implementations of scalar multiplication for the same curve cannot be much faster, and "simple" means that reasonably fast implementations of scalar multiplication for the same curve cannot be much more concise. At this time there are no examples close enough to the edge to warrant quantification of "much".

(3) NIST P-384 does not complete single or multi-scalar formulas. SafeCurves requires curves to support simple, fast, complete, constant-time single-coordinate single-scalar multiplication. This includes the SafeCurves ladder requirement but goes further by requiring completeness. SafeCurves also requires curves to support simple, fast, complete, constant-time multi-scalar multiplication.

(4) NIST P-384 does not support indistinguishability. SafeCurves note: "Elligator 2" works for any odd prime and any curve of the form y^2=x^3+Ax^2+Bx with nonzero AB(A^2-4B). This includes all Montgomery curves y^2=x^3+Ax^2+x except for one curve y^2=x^3+x. It also includes, after conversion, all Edwards curves x^2+y^2=1+dx^2y^2 except for one curve x^2+y^2=1-x^2y^2. More generally, it includes all curves with points of order 2, except for j-invariant 1728. Standard representations of elliptic-curve points are easily distinguishable from uniform random strings. This poses a problem for many cryptographic protocols using elliptic curves: censorship-circumvention protocols, for example, and password-authenticated key-exchange protocols.

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