Elliptic Curve Cryptography based Certification Authority

Mr. Kunal Abhishek
Scientist
Society for Electronic Transactions and Security (SETS), Chennai

Abstract

Elliptic Curve Cryptography (ECC) was proposed by Neal Koblitz and Victor Miller in 1985. The benefit of using ECC is that unlike RSA it gives a competitive security with a much shorter key size. ECC-based applications are supposed to be more efficient and faster than other asymmetric techniques like Elgamal and RSA. This article presents a glimpse on how a Certification Authority (CA) is designed and implemented using ECC.

1. Introduction

A Certification Authority (CA) is a trusted third party whose primary job is to issue digital certificates which is used for signing, authenticating, encrypting and non-repudiation purposes. A certification authority also generates a certificate revocation list (CRL) which contains a list of revoked certificates. A CA is a part of Public Key Infrastructure, popularly called PKI to enable secure, convenient, and efficient acquisition of public keys while ensuring sensitive data security, authenticating users and controlling the system by enforcing various policies to create, manage, store, distribute and revoke digital certificates [RFC 2822] [5].

Issues in ECC-based CA design

Design and realization of ECC-based applications for PKI services is an intricate task. Survey shows a large number of RSA-based CAs practicing in the market as compared to a very few ECC-based CAs. Companies like Thawte, Verisign, Global Sign, Comodo etc. are few names which provides ECC-based CA services. In case of India, no full-fledged ECC-based CA except for eSign services has been commercially implemented till date [1]. The root cause of scarce implementation of ECC-based CA is the interoperability issues. These issues are normally there due to complex elliptic curve mathematics as the basic operation involved in the product design as well as the issues due to ECC standards and schemes associated with the applications.

About Standards on Elliptic Curve Cryptography

Some of the ECC-based applications could be ECC supported dongle for user authentication, E-mail clients for E-mail facility with ECC support, ECC based Certification Authority for PKI services and a lot more. These applications support some standards and protocols due to elliptic curve. But mismatch of these standards in two applications taking place in communication leads to non-interoperability within the system. These standards given by different groups or bodies (refer Table 1) may not be compatible to work with each other. For say, the Public Key Cryptography Standards [9], PKCS#13, defined by RSA laboratory for ECC is not complete and is still under development. Applications usually do not support this standard at present and it often creates non-interoperability problem. In case of IEEE P1363 standard, it is fundamentally different from ANSI and FIPS 186-2 standards in the view that it does not mandate minimum security requirements. The IEEE P1363 standard also gives plenty of options to implement the public key cryptography schemes which definitely leads to interoperability issues. This is the reason to choose standards for particular scheme so that the system as a whole can be communicating seamlessly and reliably. Table 1 shows some of well-known standards recommended by standard bodies for elliptic curve cryptography.

Table 1. Standards Bodies and Working Groups [6]

Standard Body or Working Group	Standards	Abbreviated Title
ANSI	ANSI X9:62 ANSI X9:63	ECDSA Key Agreement and Key Transport. Covers ECDH, ECMQV and ECIES.
IEEE	P1363	In particular, it covers ECDSA, ECDH, ECIES and ECMQV
ISO	ISO/IEC 15946-1 ISO/IEC 15946-2 ISO/IEC 15946-3 ISO/IEC 15946-4 (draft) ISO/IEC 18033-2 (draft)	Techniques based on elliptic curves – Part 1 General Part 2: Digital Signatures Part 3: Key Establishment Part 4: Digital Signature giving message recovery Encryption Algorithm – Part 2: Asymmetric Ciphers

NIST	FIPS 186-2 FIPS 186-3	DSA, ECDSA Allows the generation of alternative curves using methods specified in ANSI X9:62
SECG	SEC1 SEC2	ECDSA, ECDH, ECIES and ECMQV Elliptic Curve listed
NESSIE	-	ECDSA, PSEC-KEM, ACE-KEM
IPA	-	ECDSA, ECDH, ECIES and PSEC-KEM
RSA Lab	PKCS#12	Elliptic Curve Cryptography

2. Overview of Elliptic Curve Cryptography

Neal Koblitz and Victor Miller in the year 1985 introduced elliptic curve mathematics to be used in cryptography. Elliptic Curve Cryptography (ECC) is considered one of the most secure mathematical crypto methods till date. One of the interesting aspects of Elliptic Curve Cryptography is the inability of mathematicians to solve the Elliptic Curve based Discrete Logarithm Problem [7].

2.1 What is an Elliptic curve?

Definition [3]. An Elliptic Curve over a field K which is either the field R of real numbers, the field Q of rational numbers, the field C of complex numbers or the finite field $\mathbf{F}_q$ of $q = p^r$ elements, is the set of points (x, y) with x, y$\in$K which satisfy the equation,

$$y^2 = x^3 + ax + b, \quad (1)$$

together with a single element denoted O and called the Point at Infinity. Here we assume K as a field of characteristic $\neq$2, 3. Let $x^3 + ax + b$ (where a, b$\in$K) be a cubic polynomial with no multiple roots. This is the short form of Weierstrass Equation with field characteristic $\neq 2, 3$.

2.2 Point at Infinity [3]

An elliptic curve E is given by (1).
We have its homogeneous form as,

$$y^2*Z = x^3 + ax*Z^2 + bZ^3, \quad (2)$$

The point (x, y) in (1) corresponds to (x, y, 1) in the projective coordinates form. To see the points which lie on curve E at the infinity, we set Z as zero.

Now, (2) becomes
$0 = x^3$ implies that $x = 0$

But $y \neq 0$ as (0, 0, 0) is not allowed.

Rescaling by y, we find that (0, y, 0) = (0, 1, 0) is the only *Point at Infinity* on E. The homogeneous formulation gives a clear interpretation of the Point at Infinity along with the finite points in a uniform manner. Since (0, 1, 0) lies on every vertical line, therefore, they intersect E at the Point at Infinity. Point at Infinity is the identity of the group law.

2.3 Determining Order of the Group on an Elliptic Curve [3]

Hasse's Theorem. Let N be the number of $_q$-points on an Elliptic Curve defined over $\mathbf{F}_q$. Then

$$| N - (q+1) | < 2\sqrt{q} \quad (3)$$

Example: Counting number of points on an Elliptic Curve.

Let E be the elliptic curve $y^2 = x^3 + 7x + 1$ over $\mathbf{F}_{101}$.
It can be shown that the point (0, 1) has an order 116, so $N_{101} = \#E(\mathbf{F}_{101})$ is a multiple of 116.

Hasses theorem says that

$$101 + 1 - 2\sqrt{101} < N101 < 101 + 1 + 2\sqrt{101},$$

which means that 82 < N101 < 122.

The only multiple of 116 in this range is 116, so $N_{101} = 106$.

Theorem: Let E be an elliptic curve defined by $y^2 = x^3 + ax + b$ over $\mathbf{F}_q$. Then

$$\#E(F_q) = q + 1 + \Sigma_{x \in \mathbf{F}q} (x^3+ax+b/\mathbf{F}_q) \quad (4)$$

Proof: For a given, there are two points (x, y) with x-coordinate x_0 if $x_0^3 + ax_0 + b$ is a nonzero square in $\mathbf{F}_q$, one such point if it is zero, and no points if it is not a square. Therefore, the number of points with x – coordinate x_0 equals 1 + (1+ $x^3+ax_0+b/\mathbf{F}_q$) Summing over all $x_0 \in \mathbf{F}_q$, and including 1 for the point ∞, yields

$$\#E(\mathbf{F}_q) = q + 1 + \Sigma_{x \in \mathbf{F}q} (1+(x^3+ax+b/\mathbf{F}_q))$$

Collecting the term 1 from each of the q summands yields the desired formula.

Example: Let E be the curve $y^2 = x^3 + x + 1$ over F_5. The nonzero squares mod 5 are 1 and 4. Therefore

$$\#E(\mathbf{F}_5) = 5 + 1 + \sum_{x=0}^{4} (x^3+ax+1/\mathbf{5}) = 6+1/5+2/5+1/5+1/5 +4/5 = 9$$

2.4 Order of a Point [3]

Let point $P \in E(\mathbf{F}_q)$. The order of P is the smallest positive integer k, such that

$$k * P = O \quad (5)$$

where O is the identity element of the group, called Point at Infinity. The order of a point is determined by point doubling and point addition till the point at infinity (O) is achieved.

2.5 Points Addition and Point Doubling [3]

Let (x_1, y_1), (x_2, y_2) and (x_3, y_3) denote the coordinates of P, Q and P+Q respectively, then x_3 and y_3 is given by,

$$x_3 = [(y_2 - y_1)/(x_2 - x_1)]^2 - x_1 - x_2$$
$$y_3 = - y_1 + [(y_2 - y_1)/(x_2 - x_1)] (x_1-x_3)$$

If P = Q then P + Q = P + P = 2P and therefore x_3 and y_3 is given by,

$$x_3 = [(3x^2_1 + a)/2y_1]^2 - 2x_1$$
$$y_3 = - y_1 + [(3x^2_1 + a)/2y_1] (x_1 - x_3)$$

2.6 Elliptic Curve Discrete Logarithm Problem (ECDLP) [3]

The security of an elliptic curve cryptography lies in the ECDLP. It is stated as below:

Given an elliptic curve E defined over a finite field $\mathbf{F}_q$, a point $P \in E(\mathbf{F}_q)$ of order n, and a point Q $\in$ (P), find the integer k $\in$ [0, n - 1] such that

$$Q = k * P, \quad (6)$$

for a certain k in k $\in$ 0, 1, 2, …., n – 1:

The integer k is called Discrete Logarithm of Q to the base P, denoted by

$$k = log_P Q \quad (7)$$

The elliptic curve parameters should be carefully chosen in order to resist index calculus attacks on the ECDLP. When E and P are properly chosen, the ECDLP is thought to be mathematically infeasible to solve.

2.7 Security of Elliptic Curve Cryptography

The security of any elliptic curve cryptosystem lies in selection of those elliptic curves whose discrete logarithm problem (ECDLP) is thought to be mathematically infeasible to solve [4]. For suitably chosen curves, key advantage of EC Cryptosystems is that, only exponential attacks are known that breaks the system. NIST, Certicom, Brainpool etc. are some agencies who have published a set of standard elliptic curves with cryptographic key sizes in the public domain. ECC however, provides a competitive security with smaller key sizes and therefore, it is considered as a cheaper, faster, low bandwidth consuming, less memory and low power consuming technique than traditional Elgamal or RSA.

The running time [7] to crack RSA is estimated as

$$Time_{RSA} \sim \exp(\sqrt[3]{\log(N)}) \quad (8)$$

where N is the product of two large primes p and q. This is a sub-exponential time that RSA executes to crack the scheme.

However, the running time [7] of an elliptic curve is roughly estimated as

$$Time_{EllipticCurve} \sim \exp(c\sqrt{N}) \quad (9)$$

where N is the cardinality of the elliptic curve which being a large prime number.

Elliptic curve creates discrete logarithm problem (DLP) known as ECDLP that allows competitively strong security with smaller key and certificate sizes. Due to smaller key used in ECC, less EEPROM is required to store the keys and certificates. Also, less data needs to be passed allowing shorter transmission time.
Note that ECDLP is solvable in exponential time by (9) and therefore it provides much more security with a given key size than that of RSA as they have sub-exponential time solution by (8).

3. Performance Analysis of RSA and ECC

Asymmetric cryptosystems based on RSA have usually got sub-exponential time complexities by (8) for cryptanalysis. RSA is quite expensive in case we use large key sizes (refer Table 2). Since ECC gives same level of security with a much smaller key size, it may be a viable candidate for asymmetric cryptography. Table 3 gives a performance analysis of execution time for RSA and ECC. Clearly ECC-based CA would be much efficient in terms of security with much shorter key sizes, computational time & cost and less prone to cryptanalyst or hackers when high key sizes (256 bit) is taken into considerations.

Table 2. Security Comparison for ECC & RSA [8]

Symmetric Security Level	ECC	RSA	Protects to Year
80	160	1024	2010
112	224	2048	2030
128	256	3072	2040
192	384	7680	2080
256	512	15360	2120

Table 3. Performance Comparison ECC-256 and RSA 3072 [From Certicom]

Operations	ECC-256	RSA-3072
Key Generation	166ms	Too Long
Encrypt/Verify	150ms	52ms
Decrypt/Sign	168ms	8s

Table 4. Operational Speed-up Comparison by Certicom and RIM

Operations	Operation Time (in Seconds)	Speedup (ECC: RSA)
RSA 1024	10.99	1
ECC 160	0.81	13.6
RSA 2048	83.26	1
ECC 224	2.19	38

Most of the Certification Authorities uses RSA with 2048 bits key size which requires a lot of computational time (refer Table 4) in various cryptographic operations which can still be achieved by ECC using 224 bits key size.

ECC 521 bits key size gives security that is equivalent to the security given by 15360 bit of RSA key size (refer Table 2). To execute 15360 bits long keys in computation requires a dedicated hardware called a crypto co-processor which is very expensive to use. This is the reason why RSA-3072 bits key is used normally. However, ECC can be implemented in available ROM and there is no need of additional co-processor to perform strong and fast authentication [Certicom, 1998].

4. Interoperability Issues due to Elliptic Curve Standards

Implementation of elliptic curve cryptography (ECC) is never been a simple task due to ECC interoperability and compatibility issues associated with different applications. These issues are generally caused due to diverse standards available in ECC for the same purpose/schemes (refer table 1). Applications are usually developed keeping adherence to different set of standards and very often they cause non-interoperability while communicating with other applications. Some of the ECC standards do not mandate minimum security requirements and other is incomplete. These ECC standards are suggested by NIST, IEEE P1363, FIPS, ANSI etc. like well accredited institutions (refer Table 1). We discuss some of the interoperability issues of ECC-based CA especially with the mail client and browsers like Thunderbird and Firefox from Mozilla company here.

ECC Interoperability Issues. Following are the issues that encountered during CA implementation especially in the CA interaction with the mail client and during generation of the X.509v3 digital certificates with implementation of suitable ECC schemes:

1. **Incomplete Standard PKCS#13**. RSA laboratory has designed Public Key Cryptography Standards i.e. PKCS#13 for Elliptic Curve Cryptography. The PKCS#13 standard is still under development [9] that means it cannot be widely used and may lead to the interoperability problems.
2. **IEEE P1363**. The IEEE P1363 standard supports ECC but only issue is that it does not mandate minimum security requirements like ANSI and FIPS 186-4 [10] standards do.
3. **Selection of elliptic curve**. Most of the web browsers and mail clients don't support all the curves recommended by different agencies like NIST, Certicom, SECG, Brainpool etc.
4. **Key agreement and key storage**. PKCS#13 is not a good choice for storing the EC-keys as it is not supported by mail clients and web browsers in general. The mail client also does not support all the ECC-based key agreement schemes.
5. **Selection of hashing algorithm**. The X.509v3 digital certificate doesn't support all hashing algorithm.
6. **Selection of signing algorithm**. The X.509v3 digital certificate doesn't support all ECC-based signing algorithm.
7. **Selection of encryption scheme**. The X.509v3 digital certificate doesn't support all Encryption algorithm.
8. **Selection of key sizes**. Most of the web browsers and mail clients don't support all the curves over different prime sizes.
9. **Selection of certificate format**. The mail client does not support all the certificate formats.

Solution to the above issues. To address the issues given above, following recommendations are suggested to be considered for a hassle-free fully interoperable certification authority implementation as given in the following sub-sections.

4.1 Avoid PKCS#13 Standard.

Avoid PKCS#13 standard due to the fact that PKCS#13 is a new and incomplete standard and most of the applications do not adhere to this. Use PKCS#12 to store keys. PKCS#12 binds keys into the X.509v3 certificate and is able to communicate with mail client without any compatibility problem.

4.2 Taking FIPS Security Recommendations along with IEEE P1363

The minimum security requirement in CA design as specified under table 2 and use 256-bit ECC-key size as a minimum recommended key size for use. This 256-bit ECC-key size is supported by almost all the web browsers and mail clients with ECC-enabled feature.

4.3 Selection of Elliptic Curve

The CA should have support for the set of elliptic curves of different cryptographic sizes to achieve interoperability for seamless functioning of the system. These curves are thought to be having very hard ECDLP and recommended for cryptographic uses. The minimum prime size of the curve given by NIST and SECG is 256 which is safe to use by 2040 (refer Table 5). The choice of the ECC curve for the CA certificate public-key should be based on the following criteria, in order of preference: interoperability, performance, and strength. The curve that achieves maximum interoperability with browsers/mail clients is of 256 bits key sizes. A 256 bits curve may offer a better balance of features. For best interoperability, elliptic curves with prime size 256 bit is suggested for use with browsers and mail clients. Table 5 shows the available curves, their prime sizes and validity for safe use as given below:

Table 5. Selection of Elliptic Curve for CA Implementation

Elliptic Curve	Recommended by	Prime/Key Size	Validity
secp256r1	SECG	256	2040
secp384r1	SECG	384	2080
secp521r1	SECG	521	2180
P-256	NIST	256	2040

4.4 Suggested Standard for Key Agreement and Key Storage

Use PKCS#12 for packaging private key along with X.509v3 certificate. The protocols and standards given as per Table 6.

Table 6. Suggested Key Agreement and Private Key Storage

Scheme	Protocol/Format	Standard
Key Agreement	ECDH, ECMQV	ANSI X9.63
Key Generation & Storage	PKCS#12 file	PKCS#12

4.5 Suggested Hash Function

Use SHA-256 or SHA-512 with Elliptic Curve Digital Signature Algorithm (ECDSA) for signing certificates with different key sizes.

4.6 Suggested Signing Algorithm

Use Elliptic Curve Digital Signature Algorithm (ECDSA) as the only signing algorithm for interoperability with the mail client and OpenSSL.

4.7 Suggested Encryption Algorithm

Elliptic Curve Integrated Encryption Scheme (ECIES) is the only well supported encryption scheme to achieve interoperability.

4.8 Suggested Key Sizes

As per FIPS recommendation [8], key sizes including 256-bit, 384-bit and 521-bit are only used. ECC-enabled mail clients and web browsers usually support 256-bit key size nicely in general. With key sizes 256-bit is recommended for safe use (refer Table 5).

4.9 Suggested Standards for CSR, Digital Certificate

Table 7 gives the Suggested standards for CSR and digital certificate.

Table 7. Suggested Standards for CSR, Digital Certificate

Scheme	Protocol/Format	Standard/Reference
CSR	-	PKCS#10
Digital Certificate	X.509v3[PKIX]	PKCS#12, RFC 3280
Digital Certificate Format	PEM, CER, DER, CRT, PFX/P12	-
Private Key	-	PKCS#12

5. ECC-based Certification Authority Design

5.1 Design Requirements

Cryptosystem. The CA uses elliptic curve cryptography (ECC) as its public-key cryptosystem. The only signature scheme to be supported is the Elliptic Curve Digital Signing Algorithm (ECDSA).

Public Key Infrastructure. The CA generates X.509v3 ECC-based digital certificates with conformance to PKIX [2] [5]. The certificates are usable in PKI-enabled Internet protocols such as SSL/TLS, S/MIME etc.

CA Certificate. The elliptic curves defined over 256 bits prime field are most suitable for the CA certificate to avoid interoperability issues with mail clients and browsers. The certificate should be in X.509v3 format. The CA certificates should be downloadable and should also be available in CER, PEM, binary (DER), CRT and P12 or PFX format.

User Registration. User registers himself/herself with the CA server in order to request for a digital certificate.

User Information. CA includes a request page which will act as enrolment front-end and which will gather some basic information (and allow a legal click-through page) prior to taking the CSR upload or collecting certificate data. The data collected should include the requesters name and contact information.

Generation of Key Pair. CA gives a software to the user to generate their key pair. User keeps private key safely at his machine and sends public key binded with the CSR to the CA by uploading it to the CA server.

Requesting Certificate. A user requests CA for issuing him a certificate by uploading a PKCS#10 Certificate Signing Request (CSR) to the CA server.

PKCS#10 Certificate Signing Request (CSR). User sends his public key along with the request through the Certificate Signing Request (CSR) by uploading it to the CA Server. A user generates his CSR using OpenSSL in PKCS#10 format with public-keys and signatures using the NIST and Certicom specified elliptic curves. The CA is interoperable with CSR generated by the user. The CSR must contain the requesters name to be used as the Common Name (CN) in the Subject Distinguished Name (DN) of the certificate and must also contain the public key to be placed in the certificate. The CA supports CSRs that contain the Email Address attribute.

End-User Certificates. X.509v3 digital certificate format is best for interoperability. CA generates its key pair and a certificate to sign user's digital certificates from the user supplied information. CA provides a link to the Certificate Directory where it publishes users' certificates. Users can download their certificates as PKCS#12 file by clicking the link and downloading it to their machine.

CA Certificate Signing Type. It can be root CA signed, self-signed or another CA signed under root CA.

End User Certificate Signing Type. Only CA-signed certificates should be permitted to be used by the user to ensure trusted third-party assurance.

Certificate Validation. The CA checks if the CSR, keys and Certificates are perfect for use.

View Certificate. User can view its X.509v3 certificate in text format too.

Certificate Revocation List (CRL). The CA generates a full CRL daily and makes it available at the location pointed to by the primary CRL Distribution Point extension that the CA populates in all end-user certificates.

5.2 CA Design Policies

Open Source based Certification Authority. Open source crypto library like OpenSSL can be used in CA design for generating ECC-based X.509v3 digital certificates.

Root CA. The CA to be designed here would be a Root CA. A copy of the CA Root certificate should be stored in the Trusted Root Certificate Store for users to trust it.

Certificate Class. Class 1, Class 2 or Class 3 certificates can be used based on the user requirements.

Support for NIST and SECG Prime Elliptic Curves. CA supports both the NIST as well as SECG Prime Curves to avoid interoperability problems with Mozilla Thunderbird Mail Client and OpenSSL.

No Point Compression. There is no any point compression technique be used in CA implementation so as to avoid patent issues.

Keys Generation and Storage. Keys should be generated on the user's machine and should be kept inside a Hardware Secured Module (HSM) like secured ash or smart card or can be kept in .p12 certificate file and nowhere else.

Minimum Key Sizes. The minimum ECC-key sizes are 256 bits. It can be scalable up to 521 bits depending upon the compatibility with the mail client/web browsers and environment in which the digital certificate would has to be used.

Web-based User Interface. There should be a secure CA server and users can interact with CA through web-based interface.

5.3 CA Architecture Model

CA incorporates a certificate directory where it publishes certificates for users to retrieve it for use. CA has also a Certificate Revocation List (CRL) server where it publishes revoked certificates and user retrieves CRL information from CRL server. The CA architecture model is shown in Figure 1.

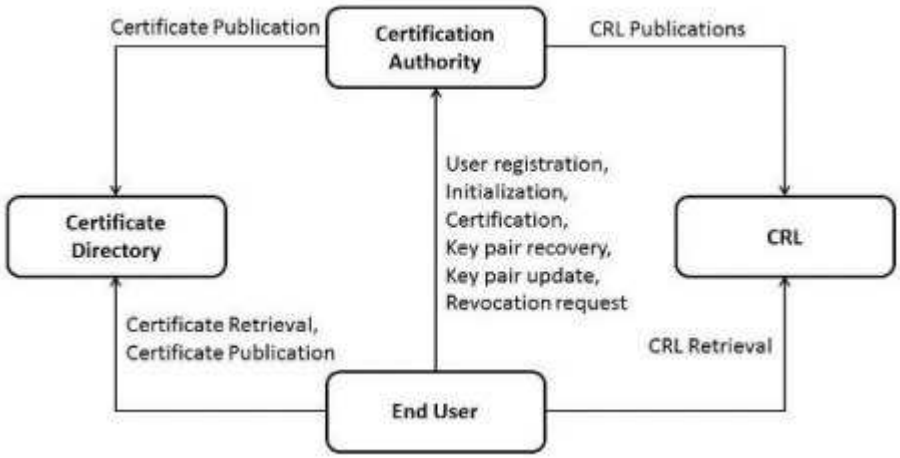


Figure 1: Certification Authority Architecture Model

6. ECC-Based X.509v3 Digital Certificate Generation

The X.509v3 digital certificate generation involves three steps: (i) key pair generation (ii) CSR generation and, (iii) certificate generation.

6.1 CA Certificate Generation

Keys Generation CA generates its key pair using OpenSSL with 521bit key size. The elliptic curve secp256r1 is recommended by SECG for certificate purposes.

CSR Generation CA generates its certificate signing request (CSR) (with self-sign certificate option) using its public key recently generated.

Certificate Generation Self signed root certificate is generated using CSR for the Root CA. This Root CA certificate is published in the certificate directory for the users to retrieve.

6.2 User Certificate Generation: User can have CA signed certificate for use. Following steps can be taken to get the user certificate.

Keys Generation User downloads a software from the CA to generate their key pair using OpenSSL crypto library with 521 bits key size.

CSR Generation User generates their certificate signing request (CSR) using his/her public key recently generated.

Certificate Generation CA takes CSR as input and generates certificate for the user as output and publishes it in the certificate directory.

Conclusion

This article covered the design and implementation aspects of a Certification Authority based on Elliptic Curve Cryptography. We discussed interoperability issues of ECC-based CA due to various ECC standards and schemes in general. As an implementation research, we believe that this work will motivate and inspire organizations to design and implement a fully interoperable ECC-based Certification Authority for trust based secure communication.

References

1. www.cca.gov.in
2. Yangtao, Yuan and Quan, Liu and Fen, Li: A Design of Certificate Authority Based on Elliptic Curve Cryptography. In: Proceedings of the 2010 Ninth International Symposium on Distributed Computing and Applications to Business, Engineering and Science (DCABES '10), pp. 454{457. IEEE Computer Society, Washington, DC, USA (2010)
3. Koblitz, Neal: A Course in Number Theory and Cryptography - 2nd Edition. Springer-Verlag, New York (1994)
4. Washington, Lawrence, C.: Elliptic Curves Number Theory and Cryptography - 2nd Edition. Chapman & Hall/CRC, Boca Raton (2008)

5. Stallings, William: Cryptography and Network Security Principles and Practice - 4th Edition. Prentice Hall, New Jersey (2005)
6. Hankerson, Darrel, Menezes, Alfred J., Vanstone, Scott: Guide to Elliptic Curve Cryptography. Springer-Verlag New York (2003)
7. Rosing, M.: Implementing Elliptic Curve Cryptography. Manning Publications Company, (1999)
8. FIPS PUB 140-2, Security Requirements for Cryptographic Modules, http://csrc.nist.gov/publications/fips/ fips140-2/fips1402.pdf
9. PKCS#13: Elliptic Curve Cryptography Standard, http://www.emc.com/emc-plus/rsa-labs/ standards-initiatives/pkcs-13-elliptic-curve-cryptography-standard.htm
10. FIPS PUB 186-4, Digital Signature standard (DSS) http://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.186-4.pdf

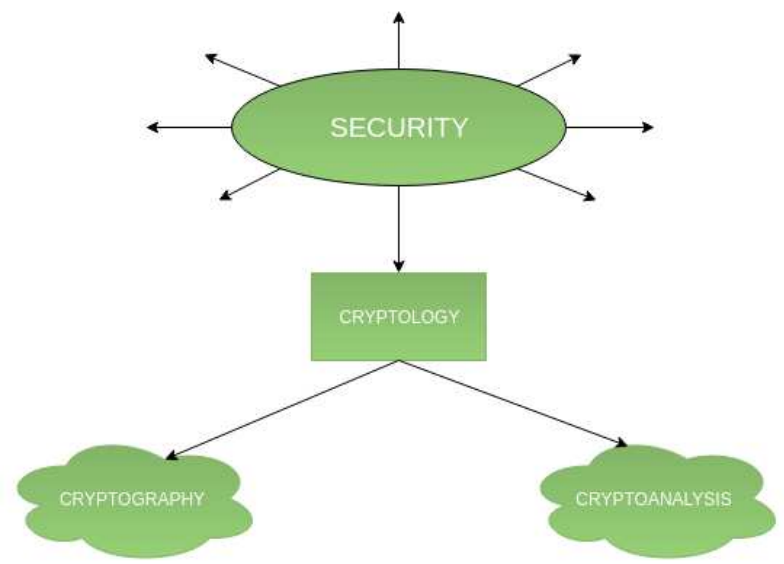
About the author

Mr. Kunal Abhishek is a Scientist at SETS, Chennai with 13 years of experience in design and development of cryptographic and Network Security products/solutions. He also served as Software Engineer in Weapons & Electronic Systems Engineering Establishment (WESEE), an R&D unit of Indian Navy for 7 years. He is Principal Investigator of SETS in-house developed PKI solution called e-Abhedya. He was instrumental in framing "Digital Signature End Entity Rules, 2015" with inclusion of ECC for PKI services under the IT Act of India. His research interest includes ECC based PKI and Secure Kernel Development. He holds an M.S. degree from BITS, Pilani and currently pursuing Ph.D. in Computer Science from Bharathidasan University, Trichy.

Cryptography | Introduction to Crypto-terminologies

Cryptography is an important aspect when we deal with network security. ‘Crypto’ means secret or hidden. Cryptography is the science of secret writing with the intention of keeping the data secret. Cryptanalysis, on the other hand, is the science or sometimes the art of breaking cryptosystems. These both terms are a subset of what is called as Cryptology.

Classification –
The flowchart depicts that cryptology is only one of the factors involved in securing networks. Cryptology refers to study of codes, which involves both writing (cryptography) and solving (cryptanalysis) them. Below is a classification of the crypto-terminologies and their various types.



Read more at https://www.geeksforgeeks.org/cryptography-introduction-to-crypto-terminologies/

Cryptography is typically bypassed, not penetrated.
Cryptography products may be declared illegal, but the information will never be.
Cryptography is the ultimate form of non-violent direct action.
Without strong encryption, you will be spied on systematically by lots of people.
Privacy and encryption work, but it's too easy to make a mistake that exposes you.