Literature Survey: Elliptic Curve Cryptography

Thomas Butterfield

27-10-20

Abstract

The aim of this literature survey is to determine the state-of-the-art in Elliptic Curve Cryptography (ECC), and to acquire a set of useful techniques, methods, and algorithms. This document is divided into five sections: Definitions, History, Implementation, Attacks, and Summary. I will firstly define some relevant terms, and discuss the history of ECC, then I will detail how it is implemented, and describe attacks on it both in theory and in practice.

1 Definitions

1.1 Elliptic Curve

Let p be a prime number, and let F_p denote the field of integers modulo p. An elliptic curve E over F_p is defined by an equation of the form

$$y^2 = x^3 + ax + b,$$

where a, $b \in F_p$ satisfy $4a^3 + 27b^2 \neq 0 \pmod{p}$ [1, 2].

1.2 ECDLP

The *Elliptic Curve Discrete Logarithm Problem* is defined as follows: given an elliptic curve E defined over a finite field F_q, a point $P \in E(F_q)$ of order n, and a point $Q \in \langle P \rangle$, find the integer $l \in [0, n-1]$ such that $Q = lP$. The integer l is called the discrete logarithm of Q to the base P, denoted $l = log_P Q$ [1].

1.3 ECDHP

The *(computational) Elliptic Curve Diffie-Hellman Problem* is defined as follows: given an elliptic curve E defined over a finite field F_q, a point

$P \in E(F_q)$ of order n, and points $A = aP, B = bP \in \langle P \rangle$, find the point $C = abP$ [1].

1.4 ECDDHP

The *Elliptic Curve Decision Diffie-Hellman Problem* is defined as follows: given an elliptic curve E defined over a finite field F_q, a point $P \in E(F_q)$ of order n, and points $A = aP, B = bP$, and $C = cP \in \langle P \rangle$, determine whether $C = abP$ or, equivalently, whether $c \equiv ab \pmod{n}$ [1].

1.5 Signature scheme

A *signature scheme* consists of four algorithms:
1. A *domain parameter generation algorithm* that generates a set D of domain parameters.
2. A *key generation algorithm* that takes as input a set D of domain parameters and generates key pairs (Q, d).
3. A *signature generation algorithm* that takes as input a set of domain parameters D, a private key d, and a message m, and produces a signature Σ.
4. A *signature verification algorithm* that takes as input the domain parameters D, a public key Q, a message m, and a purported signature Σ, and accepts or rejects the signature.
The signature verification algorithm always accepts input (D, Q, m, Σ) if Σ was indeed generated by the signature generation algorithm with input (D, d, m) [1].

1.6 Public-key encryption scheme

A *public-key encryption scheme* consists of four algorithms:
1. A *domain parameter generation algorithm* that generates a set D of domain parameters.
2. A *key generation algorithm* that takes as input a set D of domain parameters and generates key pairs (Q, d).
3. An *encryption algorithm* that takes as input a set of domain parameters D, a public key Q, a plaintext message m, and produces a ciphertext c.
4. A *decryption algorithm* that takes as input the domain parameters D, a private key d, a ciphertext c, and either rejects c as invalid or produces a plaintext m.
The decryption algorithm always accepts (D, d, c) and outputs m if c was

indeed generated by the encryption algorithm on input (D, Q, m) [1].

2 History

Cryptography is the study of mathematical systems for solving two kinds of security problems: privacy and authentication [3]. A privacy system prevents the extraction of information by unauthorised parties from messages transmitted over a public channel, assuring it is read only by the intended recipient. An authentication system prevents the unauthorised injection of messages into a public channel, assuring the legitimacy of the sender [3]. In 1976, Whitfield Diffie and Martin Hellman proposed that it was possible to develop systems in which two parties communicating solely over a public channel and using only publicly known techniques could create a secure connection [3]. This was the first public discovery of public-key cryptography.

Diffie and Hellman presented the concept of a public-key cryptosystem but not any practical implementation of such a system. So in 1978, Rivest, Shamir, and Adleman presented a new encryption method which did just that, it was the first public implementation of public-key cryptography. This method had the novel property that publicly revealing an encryption key did not thereby reveal the corresponding decryption key [4]. This is known as RSA public-key cryptography.

Victor Miller [5] and Neal Koblitz [6] discussed an analogue of the Diffie-Hellman key exchange protocol based on elliptic curves over finite fields which used the multiplicative group of a finite field. The advantage of this system was that it appeared to be immune from attacks of the style of Western, Miller, and Adleman [5, 7]. This was the invention of Elliptic Curve Cryptography.

3 Implementation

3.1 Point Arithmetic

The addition operation in $E(F_p)$ is specified as follows:

3.1.1 Point Addition

Let $P = (x_1, y_1) \in E(F_p)$ and $Q = (x_2, y_2) \in E(F_p)$, where $P \neq Q$. Then $P + Q = (x_3, y_3)$, where

$$x_3 = \lambda^2 - x_1 - x_2, \quad y_3 = \lambda(x_1 - x_3) - y_1, \quad and \quad \lambda = \frac{y_2 - y_1}{x_2 - x_1}$$

[2].

3.1.2 Point Doubling

Let $P = (x_1, y_1) \in E(F_p)$. Then $P + P = 2P = (x_3, y_3)$, where

$$x_3 = \lambda^2 - 2x_1, \quad y_3 = \lambda(x_1 - x_3) - y_1, \quad and \quad \lambda = \frac{3x_1^2 + a}{2y_1}$$

[2].

3.2 Protocols

In cryptography it is customary to personalise the participants. Typically, Alice and Bob want to communicate, while Eve, the eavesdropper, intercepts and tries to read their messages [8].
The EC algorithms are specified in *SEC 1: Elliptic Curve Cryptography* [9]. An overview of EC cryptographic algorithms for key agreement and digital signature are explained below.

3.2.1 ECDH

The *Elliptic Curve Diffie-Hellman* key agreement protocol allows Alice and Bob to securely exchange the value of a point on an elliptic curve, although neither of them initially knows the value of the point [8,10]: Alice and Bob must agree on a finite field F_q, and elliptic curve E/F_q, and a generator point $G \in E(F_q)$ of (prime) order n. Both Alice and Bob have a key pair (d, Q) consisting of a private key d (a randomly selected integer less than n) and a public key $Q = d * G$.
Let (d_A, Q_A) be the key pair of Alice and (d_B, Q_B) be the key pair of Bob.
1. Alice computes $K = (x_K, y_K) = d_A * Q_B$
2. Bob computes $L = (x_L, y_L) = d_B * Q_A$
3. Since $d_A Q_B = d_A d_B G = d_B d_A G = d_B Q_A$. Therefore $K = L$ and hence $x_K = x_L$
4. Hence the shared secret is x_K
Since it is practically impossible to find the private key d_A or d_B from the public key K or L, it is not possible for Eve to obtain the shared secret [8,10,11].

3.2.2 ECDSA

The *Elliptic Curve Digital Signature Algorithm* allows Alice to sign a digital document and Bob to verify that the signature is valid [8]: To authenticate a message sent by Alice, she signs the message using her private key. Alice sends the message and the signature to Bob. This signature can be verified only by using the public key of Alice. Since Bob know's Alice's public key, he can verify whether the message was indeed sent by Alice or not [10]. ECDSA is a variant of the Digital Signature Algorithm (DSA) that operates on elliptic curve groups.
Alice and Bob must first agree on a finite field F_p, an elliptic curve E/F_p, and a generator point $G \in E(F_p)$ of (prime) order n. Sender Alice has a key pair (d_A, Q_A) consisting of a private key d_A (a randomly selected integer less than n) and a public key $Q_A = d_A * G$ (G is the generator point). An overview of the ECDSA process is defined below.

Signature Generation For signing a message m sent by Alice, using Alice's private key d_A.
1. Calculate $e = HASH(m)$, where $HASH$ is a cryptographic hash function, such as SHA-2
2. Select a random integer k from $[1, n-1]$
3. Calculate $r = x1 \pmod n$, where $(x1, y1) = k * G$. If $r = 0$, go to step 2
4. Calculate $s = k^{-1}(e + d_A r) \pmod n$. If $s = 0$, go to step 2
5. The signature is the pair (r, s) [1, 8, 10–12].

Signature Verification For Bob to authenticate Alice's signature, Bob must have Alice's public key Q_A.
1. Verify that r and s are integers in $[1, n-1]$. If not, the signature is invalid
2. Calculate $e = HASH(m)$ where $HASH$ is the same function used in the signature generation (Section 3.2.2).
3. Calculate $w = s^{-1} \pmod n$
4. Calculate $u_1 = ew \pmod n$ and $u_2 = rw \pmod n$ 5. Calculate $(x1, y1) = u_1 G + u_2 Q_A$ 6. The signature is valid is $x1 = r \pmod n$, invalid otherwise [1, 8, 10–12].

3.3 Safe Curves

There are several different standards covering selection of curves for use in elliptic-curve cryptography:

- ANSI X9.62 (1999)

- IEEE P1363 (2000)
- SEC 2 (2000)
- NIST FIPS 186-2 (2000)
- ANSI X9.63 (2001)
- Brainpool (2005)
- NSA Suite B (2005)
- ANSSI FRP256V1 (2011)

Each of these standards tries to ensure that the ECDLP (finding a user's private key, given their public key) is difficult [13]. Unfortunately, there is a gap between ECDLP difficulty and ECC security. None of these standards do a good job of ensuring ECC security, secure implementations of the standard curves are theoretically possible but very hard. There are many attacks that break real-world ECC without solving ECDLP [13].
In 2006, Daniel Bernstein designed a new high-security ECDH function which achieved record setting speeds (twice as fast as any other function), with several side benefits including state-of-the-art timing-attack protection [14]. This function, known as Curve25519, is suitable for a wide variety of cryptographic applications. The Curve25519 function is F_p-restricted x-coordinate scalar multiplication on $E(F_{p^2})$, where p is the prime number $2^{255}-19$ and E is the elliptic curve $y^2 = x^3+486662x^2+x$ [14]. Curve25519 is regarded as a safe curve, since it satisfies the basic parameter requirements, the ECDLP security requirements, and the ECC security requirements beyond ECDLP security [13].

4 Attacks

4.1 Exhaustive Search

The elliptic curve parameters for cryptograhic schemes should be carefully chosen in order to resist all known attacks on the ECDLP. The most naïve algorithm for solving the ECDLP is exhaustive search whereby one computes the sequence of points $P, 2P, 3P, 4P, ...$ until Q is encountered. The running time is approximately n steps in the worst case and $n/2$ steps on average. Therefore, exhaustive search can be circumvented by selecting elliptic curve parameters with n sufficiently large to represent an infeasible amount of computation (e.g., $n \geq 2^{80}$) [1].

4.2 Pohlig-Hellman and Pollard's rho algorithm

The best general-purpose attack known on the ECDLP is the combination of the Pohlig-Hellman algorithm and Pollard's rho algorithm, which has a fully-exponential running time of $O(\sqrt{p})$ where p is the largest prime divisor of n. To resist this attack, the elliptic curve parameters should be chosen so that n is divisible by a prime number p sufficiently large so that $\sqrt{p}$ steps is an infeasible amount of computation (e.g., $p > 2^{160}$) [1]. If, in addition, the elliptic curve parameters are carefully chosen to defeat all other known attacks (e.g. Isomorphism attacks), then the ECDLP is believed to be infeasible given the state of today's computer technology [1].

4.3 Real-World Example

Whilst the *Elliptic Curve Discrete Logarithm Problem* is difficult to solve on classical computers [1], there is a gap between ECDLP difficulty and ECC security [13]. In 2010, it was discovered that Sony had made a basic and critical mistake in the implementation of their Elliptic Curve Cryptosystem. As discussed in section 3.2.2, as part of the Signature Generation stage of the ECDSA, a random integer k is generated and used, it is imperative that k is truly random and not predicatable in any way. However, Sony wrote their own software which **used a constant nunber for each signature**. This allowed the computation of their private key, from the public key, using simple algebra [15].

5 Summary

Much research has been done into Elliptic Curve Cryptography since its discovery in 1985 [5, 6]. The security of ECC is based on the hardness of the ECDLP, and currently the best algorithms known to solve the ECDLP have fully exponential running time. In contrast to the subexponential-time algorithms known for the integer factorisation problem, on which the security of RSA is based. This means that, for example, a 160-bit EC key provides the same level of security as a 1024-bit RSA key [1, 8]. ECC is a very strong encryption method, but only when implemented properly, using a safe curve [13, 14] and a good random number generator [15]. These are important considerations for this project.

References

[1] D. Hankerson, A. J. Menezes, and S. Vanstone, *Guide to elliptic curve cryptography*. Springer Science & Business Media, 2003.

[2] J. López and R. Dahab, "An overview of elliptic curve cryptography," 2000.

[3] W. Diffie and M. Hellman, "New directions in cryptography," *IEEE Transactions on Information Theory*, vol. 22, no. 6, pp. 644–654, 1976.

[4] R. L. Rivest, A. Shamir, and L. Adleman, "A method for obtaining digital signatures and public-key cryptosystems," *Commun. ACM*, vol. 21, p. 120–126, Feb. 1978.

[5] V. S. Miller, "Use of elliptic curves in cryptography," in *Advances in Cryptology — CRYPTO '85 Proceedings* (H. C. Williams, ed.), (Berlin, Heidelberg), pp. 417–426, Springer Berlin Heidelberg, 1986.

[6] N. Koblitz, "Elliptic curve cryptosystems," *Mathematics of computation*, vol. 48, no. 177, pp. 203–209, 1987.

[7] L. Adleman, "A subexponential algorithm for the discrete logarithm problem with applications to cryptography," in *20th Annual Symposium on Foundations of Computer Science (sfcs 1979)*, pp. 55–60, 1979.

[8] J. H. Silverman, *The arithmetic of elliptic curves*, vol. 106. Springer Science & Business Media, 2009.

[9] SECG, "1: Elliptic curve cryptography. version 2.0," *Standards for Efficient Cryptography*, 2009.

[10] M. Anoop, "Elliptic curve cryptography, an implementation guide," *online Implementation Tutorial, Tata Elxsi, India*, vol. 5, 2007.

[11] A. Jurišic and A. Menezes, "Elliptic curves and cryptography," *Dr. Dobb's Journal*, pp. 26–36, 1997.

[12] N. Koblitz, A. Menezes, and S. Vanstone, "The state of elliptic curve cryptography," *Designs, codes and cryptography*, vol. 19, no. 2-3, pp. 173–193, 2000.

[13] D. J. Bernstein and T. Lange, "Safecurves: choosing safe curves for elliptic-curve cryptography. https://safecurves.cr.yp.to, accessed 25 october 2020," 2013.

[14] D. J. Bernstein, "Curve25519: New diffie-hellman speed records," in *Public Key Cryptography - PKC 2006* (M. Yung, Y. Dodis, A. Kiayias, and T. Malkin, eds.), (Berlin, Heidelberg), pp. 207–228, Springer Berlin Heidelberg, 2006.

[15] G. Hotz, "Console hacking 2010-ps3 epic fail," in *27th Chaos Communications Congress*, 2010.