Elliptic Curve Digital Signature Algorithm and its Applications in Bitcoin

Arnt Gunnar Malvik
Bendik Witzoee

Desember 2015

1 Abstract

Elliptic Curve Cryptography is an approach to cryptography based on the usage of elliptic curves over finite fields. This approach allows for smaller key sizes when compared to other schemes in cryptography such as the RSA, while keeping the same level of security. The Elliptic Curve Digital Signature Algorithm (ECDSA) is the most widely used standardized elliptic curve-based signature scheme [5], with applications in diverse fields. One modern application of the ECDSA is found in the Bitcoin protocol, which has seen a surge in popularity as an open source, digital currency. The total value of existing Bitcoins is estimated at over 4.5 billion USD by November, 2015 [13], creating the need for a secure means of transaction and handling.

In this paper we will present the ECDSA, covering signature generation and verification. We will then discuss the consequences the choice of elliptic curves has on the performance and security of the ECDSA. As a real world application, we will discuss the Bitcoin protocol and its elliptic curve Secp256k1 [1]. Specifically, we will be looking at the Bitcoin's choice of a Koblitz curve instead of an elliptic curve over a prime field. The implications this choice has on ECDSA will then be discussed.

2 The ECDSA Algorithm

The Elliptic Curve Digital Signature Algorithm was first proposed by Scott Vanstone in 1992, and is the elliptic curve analogue of the Digital Signature Algorithm (DSA). The main advantage with ECDSA is that you achieve the same level of security as with DSA, but with smaller keys. By having smaller keys you also achieve quicker calculations and smaller public keys to pass around. The ECDSA has been accepted as ISO, ANSI, IEEE, and FIPS standards in 1998, 1999, 2000 and 2000 respectively. The algorithm is described by the following steps [8]:

(i) Setup

1. The elliptic curve group $E(a, b, p)$ with parameters a, b, p, and order either prime n or divisible by prime n
2. The primitive element $P \in E$, which is of order n
3. The prime p which is of 160 bits or more
4. The private key which is a random integer $d \in [2, n2]$
5. The public key which is a point on the curve $Q = [d]P$

(ii) Signing

1. Generate a random integer $r \in [2, n2]$
2. Compute $[r]P = (x_1, y_1)$
3. Compute the integer $s_1 = x_1 \, (mod \, n)$
4. If $s_1 = 0$, stop and go to Step 1
5. Compute $r^{-1} \, (mod \, n)$
6. Compute $s_2 = r^{-1}(H(m) + d * s_1) \, (mod \, n)$
7. If $s_2 = 0$, stop and go to Step 1
8. The signature on the message m is the pair of integers (s_1, s_2)

(iii) Verification

1.The verifier receives the message and the signature: $[m, s_1, s_2]$
2. The verifier knows the system parameters and the public key Q
3. The integers s_1, s_2 are in the range $[1, n1]$
4. Compute $w = {s_2}^{-1} (mod \, n)$
5. Compute $u_1 = H(m) * w \, (mod \, n)$
6. Compute $u_2 = s_1 * w (mod \, n)$
7. Compute $[u_1]P \oplus [u_2]Q = (x_2, y_2)$
8. Compute the integer $v = x_2 \, (mod \, n)$
9. The signature is valid if $v = s_1$

3 Bitcoin's Koblitz Curve

Bitcoin uses the elliptic curve Secp256k1 [2], which is defined by the sextuple $T = (p, a, b, G, n, h)$.
The prime $p = 2^{256} - 2^{32} - 2^9 - 2^8 - 2^7 - 2^6 - 2^4 - 1$
The curve $E : y^2 = x^3 + ax + b$ over $F(p)$ is defined by: $a = 0, \ b = 7$
The base point G in compressed form is: $G = 02 \ 79BE667E \ \ F9DCBBAC \ \ 55A06295$
$CE870B07 \ \ 029BFCDB \ \ 2DCE28D9 \ \ 59F2815B \ \ 16F81798$
The order $n = FFFFFFFF \ FFFFFFFF \ FFFFFFFF \ FFFFFFFE \ BAAEDCE6$
$AF48A03B \ BFD25E8C \ D0364141$
The cofactor $h = 01$

Bitcoin uses ECDSA in the way that every Bitcoin address is a cryptographic hash of a public key, and the owner of this specific Bitcoin is the person who has the corresponding private key. So to transfer a Bitcoin to another person you actually give that person the coin's corresponding private key. Breaking the ECDSA is equivalent to solving the elliptic curve discrete logarithm problem (ECDLP). This means that if someone manages to solve the ECDLP, or if there is a backdoor planted into the Secp256k1 curve (which as discussed in section 5 is not likely) they would gain access to 4.5 billion USD worth of Bitcoins. It is worth noting that Secp256k1 currently is not among the recommended curves by NIST.

4 ECDSA Performance

The performance of the ECDSA is mainly dependent on how efficiently we can implement multiplication of points on the chosen elliptic curve in step 2 of the signing and step 7 of the verification. These operations are relatively complicated compared to the modular inversion steps, and depending on the choice of elliptic curve, cost different amounts of field inversions, multiplications and squarings. In other words, the choice of elliptic curve will affect the performance of the ECDSA implementation.

One strategy to speed up point multiplication is to exploit group endomorphisms. An endomorphism ϕ of E over K is a map $\phi : E \mapsto E$ such that $\phi(\infty) = \infty$ and $\phi(P) = (g(P), h(P))$ for all $P \in E$, where g and h are rational functions whose coefficients lie in K. All endomorphisms of E over K form the endomorphism ring of E over K. The key to enhancing performance of point multiplication lies in the fact that certain endomorphisms allow for very fast computation of operations that are costly on the regular curve. If the mapping to the endomorphism is sufficiently cheap [6], considerable improvements in computation time can be achieved.

Bitcoin utilizes a Koblitz curve for ECDSA, which is not a bad choice in terms of efficiency. The Frobenius map is an endomorphism that maps the Koblitz curve such that $\tau(\infty) = \infty, \tau(x, y) = (x^2, y^2)$. Furthermore, $(\tau^2 + 2)P = \mu\tau(P)$ for all $P \in E_a(F_{2^m})$. If we regard the Frobenius map as a complex number, any integer k can be represented by an expansion into powers of τ such that

$$[m]P = [m_0]P + [m_1]\tau(P) + [m_2]\tau^2(P) + \cdots + [m_r]\tau^r(P) \text{ with } m_i \in \{0, 1, -1\} \text{ [6]}$$

Combining this property with normal basis representation for elements of $E(F_{2^m})$ is especially useful, since squaring in the normal basis is a virtually free left shift of the bits.

For endomorphisms that are relatively easy to compute, point multiplication can be sped up by approximately 33%, with the Frobenius map over Koblitz curves outperforming even this estimate [6]. This makes Koblitz curves such as the one used in Bitcoin especially attractive for ECDSA.

5 ECDSA Security

In the previous section it was shown that choosing Koblitz curves can have a noticeable impact on the performance of the ECDSA. However, in addition to being efficient, the ECDSA needs to be secure. An important question is: How does the choice of an elliptic curve affect the security of the signature?

Indeed, both Koblitz curves as well as curves of prime order are included in the NIST Digital Signature Standard [12], and Koblitz curves were recommended for government use as far back as in 2000 [11]. This suggests they both provide adequate security for use in the ECDSA. However there exists some speculation regarding the potential existence of backdoors in certain NIST-recommended algorithms for curve generation [3][7]. The basis of the skepticism lies in the seed chosen when constructing prime curves [4].

In contrast, the construction of Koblitz curves recommended for use in the ECDSA is straightforward and transparent. We will give a short overview of the process. A Koblitz curve is defined as $E_a = y^2 + xy = x^3 + ax^2 + 1$ over the field F_{2^m}, $a \in (0, 1)$. For cryptographic purposes, an elliptic curve's order should be divisible by a large prime [10], so we would like for the order of our curve to either be prime or divisible by a small integer and a prime [9]. If we choose m non-prime, we do not achieve this because large divisors appear from the subgroups $E_a(F_{2^d})$ where $d|m$. Thus, we choose m prime and observe that there is only one divisor for $d = 1$, from $E_a(F_2)$.

Choosing $a = 0$ and $a = 1$ for the two Koblitz curves specified over F_2 yields orders of 4 and 2, respectively. Because F_2 is a subgroup of F_{2^m}, the order of the latter must be divisible by either 4 or 2. The definition used when choosing suitable Koblitz curves for recommendation in ECDSA is that the curve order N must be 'nearly prime', that is

$$N = f * r,\ f = 2 \text{ or } f = 4 \text{ and } r > 2 \text{ is a prime.}$$

Koblitz curves turn out to never be of prime order for $m > 1$, so this condition ensures the order is divisible by a large prime. Now, iterating through values of m, one can check for $a = 0$ and $a = 1$ if the order of the curve is nearly prime and find suitable curves of interest for cryptographic purposes. Doing so will result in the curves that are recommended by NIST for the various security levels, and the selection process is transparent and easy to understand.

6 Conclusion

The elliptic curve digital signature algorithm is a widely used signature scheme, which offers security due to the difficulty of the elliptic curve discrete logarithm problem. Bitcoin has chosen a Koblitz curve for their implementation of the ECDSA, which is directly linked to the ownership of each single coin. The performance of the ECDSA algorithm is largely influenced by elliptic curve point multiplications, which can be performed significantly faster by exploiting group endomorphisms. Koblitz curves are especially suited for the ECDSA algorithm due to the Frobenius map. There seems to be little evidence that Koblitz curves such as the one used by the Bitcoin protocol offer more security than other curve types, but there exists some speculation on potential backdoors in certain elliptic curves. The transparent way in which Koblitz curves are chosen seems to render them free of this speculation.

References

[1] BitcoinWiki. *Secp256k1*. URL: https://en.bitcoin.it/wiki/Secp256k1.

[2] Certicom Corp. *Standards For Efficient Cryptography*. URL: http://www.secg.org/sec2-v2.pdf.

[3] Microsoft Dan Shumow Niels Ferguson. *On the Possibility of a Back Door in the NIST SP800-90 Dual Ec Prng*. URL: http://rump2007.cr.yp.to/15-shumow.pdf.

[4] Tanja Lange Daniel J. Bernstein. *Security Dangers of the NIST Curves*. URL: http://www.hyperelliptic.org/tanja/vortraege/20130531.pdf.

[5] Scott Vanstone Darrel Hankerson Alfred Menezes. "Guide to Elliptic Curve Cryptography". In: Springer, 2004. Chap. 4.4.1.

[6] Scott Vanstone Darrel Hankerson Alfred Menezes. "Guide to Elliptic Curve Cryptography". In: Springer, 2004. Chap. 3.

[7] Glenn Greenwald James Ball Julian Borger. *Revealed: how US and UK spy agencies defeat internet privacy and security*. URL: http://www.theguardian.com/world/2013/sep/05/nsa-gchq-encryption-codes-security.

[8] Çetin Kaya Koç. *Elliptic Curve Cryptography Algorithms*. URL: http://cs.ucsb.edu/~koc/ecc/docx/13eccalgs.pdf.

[9] Jerome A. Solinas. *Efficient Arithmetic on Koblitz Curves*. URL: http://computacion.cs.cinvestav.mx/~armfaz/res/soli2000.pdf.

[10] Horst G. Zimmer Susanne Schmidt. "Elliptic Curves: A Computational Approach". In: 2003. Chap. 3.3.

[11] National Institute of Technology. *FIPS 186-2 - Digital Signature Standard*. URL: http://csrc.nist.gov/publications/fips/archive/fips186-2/fips186-2.pdf.

[12] National Institute of Technology. *FIPS 186-4 - Digital Signature Standard*. URL: http://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.186-4.pdf.

[13] Bitcoin Watch. *Economy*. URL: http://www.bitcoinwatch.com/.