Koblitz Curves and its practical uses in Bitcoin security

Kristian Bjoernsen
krbjorn@umail.ucsb.edu

***Abstract*—Abstract-Koblitz curves are a type of elliptic curves characterized by its non-random construction which allows for especially efficient computation. This is different from the most commonly used elliptic curves that have a pseudo-random structure where the parameters are chosen by a specified algorithm. With the rise of online cryptocurrency we are seeing practical uses and implementations of Koblitz curves in the exchange and ownership of cryptocurrency.**

Bitcoin uses a specific Koblitz curve *secp256k1* defined by the Standards for Efficient Cryptography Group (SECG). The curve is defined over the finite field F$_p$:

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

With a = 0, b = 7

In my paper I will introduce Koblitz curves and look at its advantages or disadvantages in comparison to normal pseudo-random curves. I want to explore the different defined Koblitz curves from SECG and see why the specific curve *secp256k1* was chosen by the creator of Bitcoin. I also want to give an overview of how the Bitcoin protocol uses Koblitz curves to ensure security in signing and transferring funds.

I. Introduction

Neil Koblitz introduced elliptic curves with a special cryptographically usefulness in his paper *CM curves with good cryptographic properties* [1]. Parameters associated with Koblitz curves admit especially efficient implementation and complex multiplication. The name Koblitz curve used to describe binary anomalous curves over $GF(2^k)$ which have $a, b \in 0, 1$. The Koblitz curve discussed later in this paper will have $a, b \in 0, 7$. The process of which the recommended parameters associated with a Koblitz curve are chosen, is by repeatedly selecting parameters admitting an efficiently computable endomorphism until a prime order curve is found. There are several computational advantages of Koblitz curves in complex multiplication, which will be presented.

The online cryptocurrency Bitcoin has implemented the use of the Koblitz curve *secp256k1* defined by the Standards for Efficient Cryptography Group (SECG) [2]. It is believed that because of security reasons the creator of Bitcoin preferred the non-random *secp256k1* over the pseudo-randomly structured *secp256r1*. Bitcoin uses Koblitz curves in the exhange and verification of ownership of cryptocurrency, and we will explore what algorithms are used in practice in the security of Bitcoin.

II. Koblitz Curves

The name Koblitz curve used to describe binary anomalous curves over $GF(2^k)$ which have $a, b \in 0, 1$. The curves are on the form:

$$y^2 + xy = x^3 + ax^2 + 1$$

Or for curves defined over the finite field F_p:

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

The Koblitz curves defined by the Standards for Efficient Cryptography Group (SECG) are defined over the finite field F_p. This is a generalization of the Koblitz curve, but the same principles for efficiency in computation are present in both the forms of the curves.

III. Strenghts of Koblitz Curves

The implementation of Koblitz curves with characteristic 2 in Diffie-Hellman type cryptosystems have the following benefits when performing complex multiplication:

1) They are nonsupersingular
2) The order of the group has a large prime factor
3) Doubling of points on the curve is very efficient
4) The curves are easy to find

Since the curves are nonsupersingular, this prevents the use the Menezes-Okamoto-Vanstone reduction of discrete log from elliptic curves to finite fields as an attack on the elliptic curve cryptosystem. In order for it to work the curves have to be supersingular.

Since the order of the group has a large prime factor, Koblitz curves prevents the computation of descrete logs by the baby-step/giant-step algorithm, as well as the Pollard-Rho algorithm. Typically the curve defined over $GF(2^k)$ has $k \geq 160$ and with k as a prime.

The doubling of Koblitz curves can be done almost as effective as Vanstone does with supersingular curves. One avoids the problem that the descrete logarithm problem can be solved using index calculus algorithms on supersingular curves since $k \leq 6$.

Another advantage with Koblitz curves is that they are easy to find, and easily accessible for implementation. Standards for Efficient Cryptography Group (SECG) defines many Koblitz curves for the use in crypotosystems which are easily accessible online [2].

A. Orders of Koblitz Curves

One of the nice characteristics of Koblitz curves is that the orders of the curves are easy to compute.

$$order(\varepsilon(GF(2^k))) = 2^k - \left(\frac{-1+\sqrt{-7}}{2}\right)^k - \left(\frac{-1-\sqrt{-7}}{2}\right)^k + 1$$

Such that $i = \sqrt{-1}$, $i^2 = -1$.

B. Group Homomorphism of Koblitz Curves

Koblitz curves are defined over $GF(2^k)$, and this gives them the following advantagious property:

$$P = (x, y) \in \varepsilon \rightarrow Q = (x^2, y^2) \in \varepsilon$$

The advantages in computation of Koblitz curves lies within the existence of the group homomorphism:

$$\tau : \varepsilon(GF(q)) \rightarrow \varepsilon(GF(q))$$

Based on the Frobenius map:

$$\tau(x, y) = (x^2, y^2)$$

C. Point multiplication of Koblitz curves

To show how this identity is benefitial to the computation of point multiplication for Koblitz curves we write it in terms of the squaring map $\tau(x, y) = (x^2, y^2)$ and the point on the elliptic curve $P = (x, y)$:

$$\tau(\tau(P)) \oplus [2]P = [\mu]\tau P$$

This can be rewritten symbolically as:

$$[\tau^2 + 2]P = [\mu\tau]P$$

We can think of the Frobenius (squaring) map $\tau(x, y) = (x^2, y^2)$ like multiplication by the complex number τ satisfying the following equation:

$$\tau^2 + 2 = \mu\tau$$

With the solution of the equation:

$$\tau = \frac{\mu \pm \sqrt{-7}}{2}$$

By combining the squaring map with ordinary scalar multiplication, we can multiply points on E_a by any element of the ring $Z[\tau]$. [3]

The property presented above of complex multiplication is useful for elliptic scalar multiplication. This is because the squaring implementation of multiplication by τ is free when F_{2^k} is represented as a normal basis. That means it is advantageous to regard n when computing $[n]P$ as an element of $Z[\tau]$, not just an integer. In effect we replace the binary expansion of the coefficient with a *τ-adic expansion*.

Example:

$$9 = \tau^5 - \tau^3 + 1$$

$$[9]P = (x^{32}, y^{32}) - (x^8, y^8) + (x, y)$$

The use of τ-adic NAF's gives a significant reduction in the number of terms, just as NAF's give a significant improvement over binary expansions for integers.

IV. Selection of Koblitz Curve parameters

The process of which the recommended parameters associated with a Koblitz curve are chosen, is by repeatedly selecting parameters admitting an efficiently computable endomorphism until a prime order curve is found. Following SECG standard for elliptic curve cryptography, elliptic curve parameters over F_p must have:

$$\lceil log_2 p \rceil \in [192, 224, 256, 384, 521].$$

Elliptic curve domain parameters over F_p with $\lceil log_2 p \rceil = 2t$ supply approximately t bits of security. This means that solving the logarithm problem on the elliptic curves of the bits mentioned above will take approximately 2^t operations.

All the recommended elliptic curve parameters defined over F_p presented in SECG features a special form of primes for their field order p. These primes make for especially efficient implementations in elliptic curve cryptosystems.

V. The Koblitz curve $secp256k1$

The elliptic curve domain parameters over F_p associated with a Koblitz curve secp256k1 are specified by the sextuple T = (p, a, b, G, n, h) where the finite field F_p is defined by[2]:

p = FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFE FFFFFC2F

$p = 2^{256} - 2^{32} - 2^9 - 2^8 - 2^7 - 2^6 - 2^4 - 1$

The curve E: $y^2 = x^3 + ax + b$ over F_p is defined by:

a = 00000000 00000000 00000000 00000000 00000000 00000000 0000000000000000

b = 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000007

The base point G in compressed form:

G = 02 79BE667E F9DCBBAC 55A06295 CE870B07 029BFCDB 2DCE28D9 59F2815B 16F81798

The order n of the curve:

n = FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFE BAAEDCE6 AF48A03B BFD25E8C D0364141

The cofactor h of the curve:

h = 01

VI. Bitcoin

Bitcoin is a digital cryptocurrency platform developed by Satoshi Nakamoto in 2008, and released as open source in 2009. The system is based on peer-to-peer, enabling users to transfer directly without the use of a intermediary. The transactions on the network is verified by network nodes and recorded on a public distributed ledger, the so-called *block chain*, available to everyone.

To create Bitcoins, users of the network offer computing power to the network to record and verify payments into the public ledger (block chain). This activity is called $mining$ and the transaction fees on the network, as well as newly created bitcoins are distributed to the users who mine.

Bitcoin uses elliptic curves in its cryptosystem, and has implemented many ECDSA as a way of verifying ownership and facilitating transactions on the network.

VII. ECDSA in Bitcoin

ECDSA is an important algorithm used in the authorization of ownership and transfer of the Bitcoin cryptocurrency.

A. ECDSA Key Generation

Input: G and n
Output: Public key Q and private key d

1) Compute a random integer in the range $[1, n-1]$
2) Compute $Q = dG$
3) Public key: Q, Private key: d

In Bitcoin, someone with the private key that corresponds to funds on the public ledger is the authorized owner of the funds and have the option to spend them. A private key in Bitcoin is a single unsigned 256 bit integer.

Public key can either be compressed or uncompressed. Compressed public keys are 33 bytes, consisting of a prefix either 0x02 or 0x03, and a 256-bit integer called x.

B. ECDSA Signatures

Bitcoin also implements the use of ECDSA Signatures to manage ownership of funds on the public ledger. In essence, with the public key, one can determine through a mathematical operation on the signature if the signature was originally produced from the hash and the private key, without the need of knowing the private key. In Bitcoin, the signatures are 71, 72, or 73 bytes long.

VIII. Bitcoin and $secp256k1$

Before Bitcoins implementation of $secp256k1$ in its ECDSA algorithm, this specific curve was not widely used. Many of the early adopters of Bitcoin questioned the use of a seemingly simple elliptic curve, but it is now gaining popularity [4]. Users argue that the closest alternative to $secp256k1$, namely $secp256r1$ is a better alternative. Other security software companies like OpenSSL now supports the use of $secp256k1$ in its security protocols.

Like described earlier, the parameters of $secp256k1$ was chosen in a predictable way, in contrast to more popular NIST curves, which is believed to make it less likely that the creator of the curve inserted any form of backdoor into the curve.

IX. How secure is $secp256k1$

Unlike pseudo-randomly chosen parameters for elliptic curves, Koblitz curves are chosen in a predictable way. When using elliptic curve parameter standars such as SECG curves, there is some concerns regarding rigidity in the curves. SafeCurves.kr.yp.to is a internet resource that looks at the current elliptic curve cryptography standards and the security of many different standard curves. [5]

SafeCurves argues that attackers might have manipulated the choices of standard curves to be vulnerable to a secret attack that applies to a small fraction of curves. Rigidity is required to protect against corner cases in curve vulnerability. *[Rigidity is a feature of a curve-generation process, limiting the number of curves that can be generated by the process.]*[5]. Without rigidity, a curve creator could keep generating curves until a curve vulnerable to the secret attack is found.

The curve $secp256k1$ is defined by SafeCurves as a "somewhat rigid" curve, where the generation process of the curve is generally considered secure.

The current SECG chair, Dan Brown, addressed Bitcoin users on the online forum Bitcointalk.org regarding the use of $secp256k1$ in Bitcoin[6]. He explains that the Weistrass coefficients of $(a, b) = (0, 7)$ cannot be the result of malicious exaustive search of curve selection until the curve lands on a weak class. The rigidity of the curve stated above supports this claim.[6]

The SECG chair has no good explanation for the base point G, but the general understanding is that the base point G cannot contain a backdoor in ECDLP and ECDHP. The ECDSA algorithm signature check of (r, s) involves checking if r is zero. If this check is for some reason dropped there is a possibility of the G being chosen in such a way that for $(0, s)$ is valid for a particular signature. However this is very unlikely.[6]

X. $secp256k1$ vs $secp256r1$

The main difference between $secp256k1$ and $secp256r1$ is that $secp256k1$ is a Koblitz curve, while $secp256r1$ is a prime field curve. Koblitz curves are generally known to be a few bits weaker than prime field curves, but when talking about 256-bit curves, it has little impact.

$secp256k1$ is a pure SECG curve, while $secp256r1$ is a so-called NIST curve. NIST curves are more widely used and has received more scrutiny than other SECG curves. In particular, leaked documents by the NSA contractor and whistleblower Edward Showden suggested that the NSA had used its influence over NIST to insert a backdoor into a random number generator used in elliptic curve cryptography standards[7].

Satoshi, the creator of Bitcoin, would have wanted to reduce the risk of there being a backdoor in the curve he would implement, and since NIST and NSA are very close, a pure SECG curve might have been preferred.[8]

XI. Conclusion

Koblitz curves allows for fast computation and complex multiplication through the use of τ-adic expansion, and features many advantageous characteristics when used in elliptic curve cryptosystems.

Bitcoin is an cryptocurrency that has implemented the use of the Koblitz curve $secp256k1$ in its security algorithms. Other than the advantageous characteristics of Koblitz curves the choice of the specific curve $secp256k1$ was likely made with the expectation of being the most secure option to prevent backdoors in the curve. $secp256k1$ is a pure SECG curve that is not a part of the NIST standard, and this is probably one of the contributing reasons why this particular curve was implemented in Bitcoin.

In regards to security, it seems unlikely that there has been inserted a malicious backdoor in the $secp256k1$ curve, as explained by the SECG chair, Dan Brown. The combination of transparency, rigidity and the widespread use of the curve it seems unlikely that the curve is compromised.

References

[1] Koblitz: 10. N. Koblitz. CM curves with good cryptographic properties, Proc. Crypto '91, Springer-Verlag (1992)

[2] Standards for Efficient Cryptography *SEC 2: Recommended Elliptic Curve Domain Parameters* January 27, 2010 [http://www.secg.org/sec2-v2.pdf].

[3] Jerome A. Solinas *Efficient Arithmetic on Koblitz Curves* National Security Agency, Ft. Meade. March 2000.

[4] The Bitcoin Wiki, *Secp256k1*. [https://wiki.bitcoin.com/w/Secp256k1] October 31, 2015.

[5] SafeCurves, Index. [http://safecurves.cr.yp.to/index.html].

[6] Bitcoin Talk, forum. *Dan Brown e-mail reply*. [https://bitcointalk.org/index.php?topic=289795.msg3183975msg3183975] September 18, 2013.

[7] PCWorld. *Overreliance on the NSA led to weak crypto standard, NIST advisers find*. [http://www.pcworld.com/article/2454380/overreliance-on-the-nsa-led-to-weak-crypto-standard-nist-advisers-find.html] July 15, 2014

[8] Daniel J. Bernstein, Tanja Lange. *Security dangers of the NIST curves*. [http://www.hyperelliptic.org/tanja/vortraege/20130531.pdf]