INFOF-405 — Homework On the use of the Elliptic Curve *secp256k1* in an Elliptic Curve Digital Signature Algorithm in Bitcoin blocks

Robin Petit[1]
[1]Université Libre de Bruxelles, MA1-CS – BA3-MATH

I. Introduction to Cryptocurrencies

Cryptocurrencies are currencies built on the idea of *decentralization*. Their goal is to avoid being dependant of an organization that handles authentication, wealth verification and ledger. In order to achieve such decentralization, these currencies use a peer-to-peer system called *blockchain*. Blockchains copies are kept by users, and for every transaction, a new block is added to the blockchain. The intrinsic structure of these blocks depends on the currency.

Also, as the blockchain system is distributed, it needs to be synchronized. This is why users need to provide some sort of proof before validating blocks. These proofs can take several forms [1], [2]:

- A proof of work (PoW) requires from the users that they solve mathematically complicated tasks to solve (but easy to check). One can cite the HashCash [3] PoW used in Bitcoin [4].
- A proof of Stake (PoS) chooses a user to validate a transaction according to a random distribution weighted by the wealth of each user [5].
- A proof of Burn (PoB) requires from users that they *burn* (i.e. *pay*) a given amount of their currency in order to validate a transaction.

Each of these methods have pros and cons and different goals and flaws. Yet, this is not what is of interest in this report.

Although the users of such currencies don't want to have a single entity verifying and holding the data, they still want their transactions to be secure and their possessions to be protected. In order to prove the users' identity, Bitcoin blocks are signed using a Elliptic Curve Digital Signature Algorithm (ECDSA), in particular the curve *secp256k1* [6], a standard elliptic curve recommended by the Standards for Efficient Cryptography [7].

II. Use of secp256k1 ECDSA in Bitcoin block signature

A. Elliptic Curve Digital Signature

To have verified transactions, in order that only the owner of some wealth is able to use it in a payment, transactions are signed. The signature algorithm used in Bitcoin is an ECDSA.

An ECDSA works on elliptic curves on finite fields. *secp256k1* can be separated in 5 parts: *sec* meaning that this curve is a standard according to SEC [7], *p* meaning that the finite field is a prime field $\mathbb{F}_p$ (more precisely $\mathbb{Z}_p = \mathbb{Z}/p\mathbb{Z}$ for a prime p), *256* meaning that the elliptic curve parameters are 256-bits long, *k* meaning that this curve is a *Koblitz* elliptic curve as opposed to *random* elliptic curves, and *1* meaning that this curve is the first secp256kx standard curve (and is actually the only one).

An elliptic curve is defined by:

$$y^2 = x^3 + ax + b \mod p, \tag{1}$$

where a, b are the parameters of the curve, and p is the size of the field $\mathbb{F}_p$. [1] We denote by E the set of all points (x, y) satisfying (1), i.e.:

$$E = \left\{(x, y) \text{ s.t. } y^2 = x^3 + ax + b\right\} \subset \mathbb{F}_p \times \mathbb{F}_p.$$

Elliptic curves have as a trivial property that for every (x, y) being in the curve, $(x, -y)$ is also in the curve by horizontal symmetry induced by the only appearance of y being in a squared form. The other interesting property of such elliptic curves is that every for every non-vertical line L:

- If L is tangent to a point P of the curve, then $\exists! Q \in E \setminus E$ such that $Q \in L$.
- If $\exists P, Q \in E$ with $P \neq Q$ such that $\{P, Q\} \subset L$, then $\exists! R \in E \setminus \{P, Q\}$ such that $R \in L$.

These properties allow a definition of product and addition on E.

B. Operators and structure of group on E

Let's define the product:

$$\cdot_E : \mathbb{Z}_p \times E \to E : (k, P) \mapsto k \cdot_E P, \tag{2}$$

as follows:

$$\forall (k, P) \in \mathbb{Z}_p \times E : \begin{cases} (2k) \cdot_E P & := 2\left(k \cdot_E P\right) \\ (2k+1) \cdot_E P & := (2k) \cdot_E P + P \end{cases}$$

This definition requires then only the addition to be defined since $2 \cdot_E P = P + P$.

[1]This expression is well defined since $x, y, a, p \in \mathbb{F}_p$ with $\mathbb{F}_p$ a field. Therefore, stable addition and product are provided to $\mathbb{F}_p$.

Let's define the sum:

$$+_E : E \times E \to E : (P, Q) \mapsto P +_E Q \tag{3}$$

according to the following properties:

- If $P = Q$, then there exists a unique $R = (x_R, y_R) \in E$ such that the tangent line to P crosses E at R. Therefore define $P +_E Q = P +_E P := R' = (x_R, -y_R)$.
- Else, there exists a unique $R = (x_R, y_R) \in E$ such that the line $\langle P, Q \rangle$ crosses E at R. Therefore define $P +_E Q := R' = (x_R, -y_R)$.

Note that these definitions are based on the elliptic curves defined on the field $\mathbb{R}$, but these properties still hold when discretized on a finite field (such as $\mathbb{Z}_p$).

This structures provides a structure of Abelian group [8] to E with the point at infinity as 0_E.[2]

C. *Definition of secp256k1 as ECDSA*

secp256k1 is standardized in SEC2 with the following parameters (on 256 bits, i.e. 32 bytes):

- p = `FFFF FFFF FFFF FFFF FFFF FFFF FFFF FFFF FFFF FFFF FFFF FFFF FFFF FFFE FFFF FC2F` = $2^{256} - 2^{32} - 977$.
- $a = 0$.
- $b = 7$.

In addition to these parameters, the ECDSA also needs a point $G \in E$ called *base point* and n its order (i.e. the value $n \in \mathbb{N}^*$ such that $n \cdot_E G = 0_E$ or also the cardinality of the Abelian subgroup $\{n \cdot_E G\}_{n \in \mathbb{N}^*} \subset E$) with as constraint that n must be prime. secp256k1 uses:

- $G = (G_1, G_2)$ for:
  - G_1 = `79BE667E F9DCBBAC 55A06295 CE870B07 029BFCDB 2DCE28D9 59F2815B 16F81798`,
  - G_2 = `483ADA77 26A3C465 5DA4FBFC 0E1108A8 FD17B448 A6855419 9C47D08F FB10D4B8`.
- $n = \left|\{n \cdot_E G\}_{n \in \mathbb{N}^*}\right|$ = `FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFE BAAEDCE6 AF48A03B BFD25E8C D0364141`.
- $h = 1$.

Where h is defined as the cardinality of the quotient group $E/\langle G \rangle$.[3] Note that $h = 1$ is equivalent to say that $E = \langle G \rangle$

The sextuple $T = (p, a, b, G, n, h)$ defines completely the elliptic curve [7].

Flaws of such curves and signatures and proposition for newer ones are described in [9], [10].

D. *Signature*

The signature is based on asymmetric encryption: each user of the Bitcoin system has a unique couple:

$$(K_{PR}, K_{PU}) \in [\![1, n[\![\times E$$

containing the private key K_{PR} and the public key K_{PU}, with the following constraint:

$$K_{PU} = K_{PR} \cdot_E G.$$

This means that there exists a bijection $\pi_G : [\![1, n[\![\to E : k \mapsto k \cdot_E G$ that is a mapping between private and public keys. So *theoretically* knowing the public key is sufficient to find the private key associated to it. Yet, despite allowing to find $K_{PU} = \pi_G(K_{PR})$ easily, the computational power doesn't allow yet to find easily $K_{PR} = \pi_G^{-1}(K_{PU})$.

One can also deduce that the maximum amount of private keys is given by n because:

$$\forall m \in \mathbb{N}^* : (n + m) \cdot_E = (n \cdot_E G) +_E (m \cdot_E G) = m \cdot_E G.$$

The *final signature* is a couple (r, s) defined on the data D to be signed as follows. Take $k \in [\![1, n[\![$ at *random* (see Subsection II-F) such that for $(x, y) := k \cdot_E G$, there is $r := x \mod n \neq 0$ and $s := k^{-1}(D + rK_{PR}) \mod n \neq 0$, and then define the signature as the couple (r, s). [11]

Note that the operations are taken $\mod n$, meaning that all of these lie in $\mathbb{Z}_n$, which is a field (since n is required to be prime). Therefore $k^{-1} \in \mathbb{Z}_n$ is well defined.

For Bitcoin, the data D to be signed is a SHA-256 [12] hash of the block to sign. D is thus 256 bits long, which is a condition for the signature [11].

E. *Verification of the signature*

The main objective of a signature is to be able to identify the user that has emitted a block. So anyone should be able to verify that the signature is correct, even without the knowledge of K_{PR} that has been used in the signature.

For a third party to check the validity of a signature, all that is needed is the signature couple (r, s) and the public key K_{PU} of the emitter. The procedure is the following: with $(r, s) \in \mathbb{Z}_n \times \mathbb{Z}_n$ and $K_{PU} \in E$, let's define:

$$(u, v) := \left(Ds^{-1}, rs^{-1}\right) \in \mathbb{Z}_n \times \mathbb{Z}_n,$$

and:

$$(x, y) := u \cdot_E G + v \cdot_E K_{PU} \in E.$$

Then the signature is valid if and only if $r = x \in \mathbb{Z}_n$.

F. *Security issues*

This method of signature relies on a hash function that must be cryptographically secured. This is why SHA-256 is used and not SHA-1 which is now considered insecure.

The most important security issue is the generation of the random number $k \in [\![1, n[\![$. As the only parameters that are unknown to the network are the private key K_{PR} of the signer and the random number k, if there is a flaw on this generation, the private key can be found out, and then reused to falsify transactions. [13]

[2] The additive opposite of a point $(x, y) \in E$ being $(x, -y) \in E$.

[3] With $\langle G \rangle$ denoting the subgroup generated by G.

III. Conclusion

To be able to identify a user performing a transaction in order to prove that a transaction has accepted by them, Bitcoin uses a digital signature algorithm using an elliptic curve (secp256k1). Such ECDSA's use algebra on finite groups and fields to sign each transaction by two numbers r s whose validity is easy to check to avoid malicious attacks on the Bitcoin network.

Yet, despite being simple to perform and to verify, this signature process must be highly secure because if it were to be hacked, members of the Bitcoin network would not be assured of the safety of their *wallet* because anyone could *mimic* their signature.

References

[1] "Bitcoin.org - proof of work," https://en.bitcoin.it/wiki/Proof_of_work.

[2] I. Bentov, A. Gabizon, and A. Mizrahi, "Cryptocurrencies without proof of work," *CoRR*, vol. abs/1406.5694, 2014. [Online]. Available: http://arxiv.org/abs/1406.5694

[3] A. Back, "Hashcash - a denial of service counter-measure," http://www.hashcash.org/papers/hashcash.pdf, 2002.

[4] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," https://bitcoin.org/bitcoin.pdf, 2008.

[5] BitFury Group, "Proof of stake versus proof of work — white paper," http://bitfury.com/content/5-white-papers-research/pos-vs-pow-1.0.2.pdf, 2015.

[6] https://bitcoin.org/en/developer-guide.

[7] Certicom Research, "Standards for efficient cryptography 2 (sec2) version 2," 2010.

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

[9] D. J. Bernstein, N. Duif, T. Lange, P. Schwabe, and B.-Y. Yang, "High-speed high-security signatures," http://ed25519.cr.yp.to/ed25519-20110926.pdf, 2011.

[10] J. Joseffson and I. Liusvaara, "Rfc8032," Internet Requests for Comments, RFC Editor, RFC 8032, January 2017. [Online]. Available: https://tools.ietf.org/html/rfc8032

[11] National Institute of Standards and Technology, "FIPS PUB 186-4: Digital signature standard (dss)," July 2013.

[12] ——, "FIPS PUB 186-4: Secure hash standard (shs)," August 2015.

[13] T. Pornin, "Rfc6979," Internet Requests for Comments, RFC Editor, RFC 6979, August 2013. [Online]. Available: https://tools.ietf.org/html/rfc6979