`OP_SCHNORRCHECKSIG`: Exploring Schnorr Signatures as an Alternative to ECDSA for Bitcoin

Olaoluwa Osuntokun
`osuntokun@.cs.ucsb.edu`

April 2015

Abstract

Bitcoin[1] currently utilizes the Elliptic Curve Digital Signature Algorithm (ECDSA)[8] as a zero-knowledge proof of ownership[10] in order to authorize the transfer of Satoshis[14] from one output to another. ECDSA applied to cryptocurrency has its share of short comings, namely: signature malleability can invalidate unconfirmed transaction chains[16, 13], techniques for batch verification have been shown to be insecure[6], and support for threshold signatures require Secure Multi-Party Computation[7].

For this project, I will explore the possibility of integrating Schnorr Signatures[11] into Bitcoin in the form of a new `OP_*CHECKSIG` operator as an alternative to ECDSA. Until recently (2008), Schnorr Signatures were encumbered by US Patent 4,995,082[11]. The primary advantages of Schnorr Signatures over Elliptic Curves, as defined in [9, 4], include the support of efficient batch signature verification[6], immunity to malleability[9], resistance to hash-function collisions, and support for efficient usable threshold signatures due to the simplicity of the signature[15, 2].

Additionally, I will contribute contribute an implementation of batch signature verification with fraud detection as described in [6] to an open source library[5] that implements ed25519[9] in Go[3]. Furthermore, I will run a series of benchmarks aiming to demonstrate the potential speed optimizations that block and transaction verification[12] can gain by moving to Schnorr. Finally, in order to demonstrate the flexibility of Schnorr with respect to threshold transactions, I will implement a scheme supporting arbitrary N-of-M threshold signatures with $log_2\binom{N}{M}$ space efficiency[2].

Deliverables include: a written report, presentation, and demo code.

References

[1] Satoshi Nakamoto, *Bitcoin: A Peer-to-Peer Electronic Cash System*, `https://bitcoin.org/bitcoin.pdf` 2009.

[2] Gregory Maxwell, *SF Bitcoin-Dev Meetup: Requirements for future multi-signature* , `https://people.xiph.org/~greg/gmaxwell_sfbitcoin_2015_04_20.pdf`, April 20th 2015,

[3] Daniel J. Bernstein, *The Go Programming Language*, `https://golang.org/`

[4] Daniel J. Bernstein, *Curve25519: new Diffie-Hellman speed records*, `http://cr.yp.to/ecdh/curve25519-20060209.pdf`

[5] Adam Langley, *ed25519 for Go*, `https://github.com/agl/ed25519`

[6] Daniel J. Bernstein, Jeroen Doumen, Tanja Lange, Jan-Jaap Oosterwijk *Faster batch forgery identification*, `http://cr.yp.to/badbatch/badbatch-20120919.pdf`

[7] Steven Goldfeder, Joseph Bonneau, Edward W. Felten, Joshua A. Kroll, Arvind Narayanan *Securing Bitcoin wallets via threshold signatures*, `http://www.cs.princeton.edu/~stevenag/bitcoin_threshold_signatures.pdf`

[8] Bitcoin WIki: Elliptic Curve Digital Signature Algorithm, `https://en.bitcoin.it/wiki/Elliptic_Curve_Digital_Signature_Algorithm`

[9] Daniel J. Bernstein, Niels Duif, Tanja Lange, Peter Schwabe, Bo-Yin Yang, *High-speed high-security signatures*, `http://ed25519.cr.yp.to/ed25519-20110926.pdf`,

[10] Bitcoin WIki: `OP_CHECKSIG`, `https://en.bitcoin.it/wiki/OP_CHECKSIG`,

[11] Schnorr C.P, *Method for identifying subscribers and for generating and verifying electronic signatures in a data exchange system*, US Patent 4,995,082, http://www.google.com/patents/US4995082, 1991

[12] Bitcoin WIki: Transaction, `https://en.bitcoin.it/wiki/Transaction#Verification`

[13] Pieter Wuille, *Dealing with malleability* `https://github.com/bitcoin/bips/blob/master/bip-0062.mediawikit`

[14] Bitcoin WIki: Transaction, `https://en.bitcoin.it/wiki/Transaction#Output`

[15] Stinson, Douglas R and Strobl, Reto, *Provably secure distributed Schnorr signatures and a (t, n) threshold scheme for implicit certificates*, Information Security and Privacy, pgs 417-434, 2001,

[16] Bitcoin WIki: Transaction Malleability, `https://en.bitcoin.it/wiki/Transaction_Malleability`