Simple Schnorr Multi-Signatures with Applications on Bitcoin

Abstract. ECDSA (Elliptic Curve Digital Signature Algorithm) is the current digital signature scheme for Bitcoin. The Schnorr signature algorithm, although still lacking a standardization, is superior in every aspect to ECDSA. This work presents a new Schnorr-based multi-signature scheme called MuSig which makes use of key aggregation and is provably secure in the plain public-key model.

Introduction. Multi-signatures are a form of technology used to add multiple participants to cryptocurrency transactions. The Schnorr multi-signature technology provides a fairly simple and neat solution of combining digital signatures and keys where multiple signers with their own public and private keys sign one message. This means that instead of generating an individual signature for each new bitcoin transaction, signatories can only use one signature.

Digital Signature. A digital signature is used to validate the authenticity and integrity of a message or digital document. It's the digital equivalent of a handwritten signature but it offers far more inherent security. Basically the signer signs a hashed message with his private key and sends the signed message along with his public key and the plaintext message to the receiver so the receiver can verify if the message has not been modified or the signer is the real sender.

Schnorr Signature. Schnorr signature is a digital signature which is secure under discrete logarithm problem. Like ECDSA, Schnorr also uses the same private-public key pairs. The only difference is in the signing and verification algorithm, which happens to be a lot simpler than ECDSA as it is linear. Here, we have a cyclic group $\mathbb{G} = \langle g \rangle$ of prime order p and a random secret key $sk \in \mathbb{Z}p$. To sign a message m, we pick $r \in \mathbb{Z}p$ randomly and compute $R = g^r$, $c = H(X, R, m)$, $s = r + cx \bmod p$. Then the signature is $\sigma = (R,s)$.

Multi-signature. A multisignature scheme allows a group of signers to produce a joint signature on a common message, which is more compact than a collection of distinct signatures from all signers. Given this signature and the list of signers' public keys, a verifier is able to check if every signer in the group participated in signing.

Naive Schnorr Multi-signature. Based on key aggregation property of $X = \Pi^{n}_{i=1} X_i$ and $R = \Pi^{n}_{i=1} R_i$ But it is not secure under cancellation problem.

Rogue Key Attack. Multisignature schemes have to be secure against the rogue key attack where an adversary chooses his public key as a function of public keys of honest users.

Bellare and Neven Multi-signature. The Bellare-Neven scheme prevents rogue key attack by having a distinct challenge c_i for each signer in its partial signature $s_i = r_i + c_i x_i$,$c_i = H$ (⟨L⟩, X_i, R, m). However, key aggregation is no longer possible since the entire list of public keys is required for verification $g^s = R\Pi^n_{i=1} X_i^{ci}$.

Maxwell Multi-signature. It is a variant of the BN scheme based on schnorr multi-signature with key aggregation and distinct challenge property where $X = \Pi^n_{i=1} X_i$, $c_i = H_{agg}$ (⟨L⟩, X_i). H_{sig} (X, R, m) , $a_i = H_{agg}$ (⟨L⟩, X_i)

Applications to Bitcoin. Fraudulent actions such as hacking have become increasingly common as cryptocurrency has grown in popularity. Many businesses have implemented multi-signature agreements for their transactions in order to limit fraudulent activity because it provides a higher level of security. For example, Lightning-network-channel and Multisig-wallet are some examples of multi-signature applications in Bitcoin.

Implementation. We implemented the Schnorr multi-signature in Python 3.0 because of the convenience with large numbers and Due to limited resources, we considered fewer restrictions.

Results. Here, you can see the difference in runtime depending on different number of signers

Number of signers	Setup time [ms]	Key Generation time [ms]	Signature time [ms]	Verification time [ms]	Overall time [ms]
10	55,99	1,55	16,67	2,44	76,65
30	41,96	3,49	139,65	4,97	190,07
100	51,43	11,27	1950,04	18,18	2030,91
300	70,96	30,71	26757,31	93,43	26952,41

Conclusion. The purpose of this report is to provide an introduction to the Schnorr signature algorithm and describe some of its amazing applications to bitcoin and the benefits and improvements that would result from its implementation. Improved efficiency and privacy are the benefits that Schnorr would bring to Bitcoin.