# Evolutions of BIP340

## Schnorr Signature for secp256k1

Antoine Riard, Bitcoinology, August 2024

# About: ariard

- bitcoin protocol hacker ~2018 (base-layer and lightning)

- managing partner @ thelab31.xyz (R&D / security consulting “boutique”)

- areas of research interest: protocol security and bitcoin scalability
    - cross-layer mempool issues (e.g mempoolfullrbf)
    - lightning: time-dilation, dust-inflation and pinning attacks
    - coinpool and payment pools research

- privacy note: no photo thanks

# Talk Index

- 1) Background on ECC Schemes in Bitcoin
- 2) Multi-Sig and Threshold Signatures Schemes
- 3) Adaptor Signatures Schemes
- 4) Blind Signatures Schemes

# A Small Dive in Public Key Cryptography

- cryptographic signature:
    - digital bits "*that must be easy for anyone to recognize the signature as authentic, but impossible for anyone other than the legitimate signer to produce it*"
    - **New Directions in Cryptography** (1976)

- RSA signature scheme: "*i know d the modular multiplicative inverse of e modulo the product of p and q prime numbers*"
    - **A Method for Obtaining Digital Signatures and Public-Key Cryptosystems** (1978)

- ECC signature scheme: "*i know the discrete logarithm over finite fields*"
    - **A Public Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms** (1985)

# ECC Signature Schemes on Bitcoin

- originally, coins can be locked under 2 script operations:
  - CHECKSIG: stack input *signature pubkey*
  - CHECKMULTISIG: stack input *dummy sig1 sig2 <num_of_sigs> pub1 pub2 <num_of_pubs>*

- ops are using ECDSA, i.e Elliptic Curve Digital Signature Algorithm:
  - a Elliptic Curve Cryptography scheme standardized in 2009
  - less compact and malleable compared to Schnorr signature

- why genesis bitcoin didn’t have Schnorr signature
  - patent encumbering the Schnorr signature scheme expired only in 2010
  - schnorr scheme originally designed for limited devices e.g smart cards

# Schnorr Signature Scheme (BIP340)

- **key generation**: P = d . G, with *d* the *scalar* and G the *curve generator*

- **signature**:
    - R = k . G, with *k* the *scalar* and G the *curve generator*
    - e = hash(m), with *m* the *message*
    - sig = k + ed

- **verification**:
    - R = sig . G - e . P
    - k . G = k + ed . G - e . (d. G)
    - k . G = k + ed - ed . G
    - k. G = k. G

# Multi-Signature Schnorr #1

- problem: a group of signers wish to produce a joint signature on a joint message

- main advantage: *compactness*
    - a single public key
    - a single signature

- solution: group signers rounds of exchange to build common pubkey and signature:
    - **Simple Schnorr Multi-Signatures with Applications to Bitcoin** (2018)
    - **MuSig2: Simple Two-Round Schnorr Multi-Signatures** (2021)

# Multi-Signature Schnorr #2

- **key aggregation:** non-interactive round where each signer exchanges a *public key*

- **nonce exchange**: interactive round  where each signer exchanges a *public nonce* on the *message*

- **partial signature exchange**: interactive round where each signer exchanges a *partial signature* on the *message*

- An *honest* coordinator aggregates the nonces and finalizes the signatures

# Multi-Signature Schnorr #3

- schnorr-based multi-signature can be used for many custody use-cases

- a single (self-)custodian can secure coins under a single pubkey
    - contrary to CHECKMULTISIG, the signing policy is *hidden*
    - keys can be *distributed* on separate devices / hardware wallets

- a N number of custodians can secure coins under a single pubkey
    - signing policy to coordinate the coin unlock can be kept *hidden*
    - each custodian can independently verify *overlay rules* on coins  spend
    - e.g federation deployment

# Threshold-Signature Schnorr #1

- problem: a group of signers wish to produce a joint signature on a joint message
  - they do not know ahead who will be available at signing time
- main advantage: *high availability*
  - a M number of signer offliness does not prevent signatures production
- solution: group signers rounds of exchanges to build common pubkey and signature:
  - **FROST: Flexible Round-Optimized Schnorr Threshold Signatures** (2020)

# Threshold-Signature Schnorr #2

- **distributed key generation:** each signer generates a *share* and a verifiable *shamir secret commitment* ; then broadcast, aggregate and verifies them.

- **nonce exchange**: interactive round where each signer generates a *nonce* and a *nonce commitment* on the *message*.

- **signature exchange**: interactive round where each signer generates a *partial signature* on the *message.*

- An honest coordinator aggregates the nonce and partial signatures.

# Threshold-Signature Schnorr #3

- schnorr-based multi-signature can be used for many custody use-cases

- a lightning node can have fallback signers for high-availability:
    - e.g high-stake channels are locked under a n-of-m threshold scheme
    - if a signer server is down, funds can be still be spent

- a N number of custodians can offer high-availability for withdrawals:
    - e.g large-scale federation deployment with 10 or more participants
    - if one participant is offline, the funds can be spends according to the signing policy

# Adaptor Signature Schnorr #1

- problem: a group of signers wish to communicate among themselves secrets by producing a joint signature on a joint message

- main advantages: *privacy and authenticity*
    - signature finalizer reveals a secret by acquiring a coin
    - a posteriori plausible deniability

- solution: one participant among the group of signer replace its nonce by the public and the nonce
    - **Scriptless Scripts** (2017)

# Adaptor Signature Schnorr #2

- **key setup**: lock a coin under a *public key* owned by a first participant.

- **adaptor signature generation**: the first participant reveals an adaptor signature for a second public key to a second participant.

- **adaptor signature finalization**: the first participant spends the coin, revealing the secret to the second participant.

# Adaptor Signature Schnorr #3

- schnorr-based adaptor signature can be used for many Bitcoin contracts use-cases

- a N number of participants can orchestrate compact coin swaps:
  - secret exchange among participants is *obfuscated*

- a provider can sell solutions to cryptographic puzzles:
  - e.g sudoku contest where a valid olution is bought from a random participant
  - probably, the adaptor public key should be a ZKP of its own

# Blind Signature Schnorr

- problem: a signer wish to produce a sign a message on behalf of another party without reveal in cleartext of the message

- main advantage: *privacy*
    - signer can attest attributes without of a ciphered message

- solution: the party tweak the message to be blinded before to submit to signature of a signer:
    - **Blind Signatures for Untraceable Payments** (1983)

# Thanks to Bitconology!