Defining a BIP-322 Signature Suite

The bitcoin blockchain, cryptocurrency and associated scripting language has been a catalyst for much of the innovation around decentralised technologies that has taken place over the last 10 plus years. This influence can be seen throughout many past RWOT papers. This advance reading proposes a natural extension to two earlier RWOT papers: Smart Signatures [1] and Smarter Signatures: Experiments in Verification [2]. These papers seeded the idea of using bitcoin scripts outside of their native blockchain context, e.g. for CA-style certificates (think Veriable Credentials) and complex delegation use cases.

A lot has changed since these ideas were initially proposed. Within the bitcoin space, Segwit, Taproot and Schnorr signatures have all been proposed and adopted within bitcoin core, extending the possibilities for writing and evaluating complex script logic. Additionally, Bitcoin Inprovement Propsal (BIP) 322 [3] proposes a generic signed message format that enables the use of the bitcoin scripting engine to sign and verify signatures on arbitrary messages. BIP 322 defines a concrete well specified construct to realise the smart signature scheme conceptualised in early RWOT papers [1,2].

BIP 322 has been implemented, or is in the process of being implemented, in a number of bitcoin libraries including a pending P.R. to bitoin core. Through work with Digital Contract Design at Legendary Requirements, I personally have created a python implementation of this scheme and used this to experiment with BIP 322 signatures in a series of Jupyter notebooks available on Github [4]. These include examples of n-of-m multisig BIP322 signatures, that begin to demonstrate the power of these smarter signature schemes. As a relative novice of the bitcoin scripting engine, I believe this is only just scratching the surface of the logic it is possible to encode within a script and therefore a BIP 322 signature.

Alongside these developments in the bitcoin space, both the Verifiable Credential Data Model and Decentralised Identifier specifications have been developed and transitioned into formal standards recommended by the W3C. This advance reading is a call for collaboration to explore how BIP 322 signatures could be integrated into these standardised data models and the protocols that use these. The aim would be to define standard approach for including a BIP 322 signature as a verification method within a DID document and it's use for DID Auth and credential issuance and verification. A BIP 322 Signature Suite.

A Brief Overview of BIP 322

A signer creates a BIP 322 signature over a message, `m`, by creating two virtual bitcoin transactions: referred to as `to_spend` and `to_sign` in the BIP. At a high level, a BIP 322 signature is produced by creating a valid spend of the `to_spend`'s Unspent Transaction Output (UTXO) which is consumed as an input to the `to_sign` transaction. The signer defines the `to_spend`, according

the BIP 322 specification, which includes specifying a hash of the message to be signed and the locking script for its UTXO. Subsequently they create the `to_sign` transaction (again following the BIP), identifying the `to_spend` UTXO as its input using its transaction id (a hash of the `to_spend` transaction) and the index of the output being spent (generally 0 as there is only one output). Finally, the signer signs the `to_sign` transaction proving they are authorized to spend the UTXO identified in the `to_spend` transaction. A BIP 322 is a base64 encoding of this signed `to_spend` transaction (or just the signature part, the witness, where the transaction uses Segwit).

For verification, a verifier is provided with a bitcoin invoice address, message and BIP 322 signature. From this information alone, they can reconstruct the `to_spend` as defined in the BIP and decode the signature to retrieve the signed `to_sign` transaction. Following this they simply verify the `to_sign` transaction is a valid spend of the `to_spend` transaction according the the bitcoin validation rules.

Conclusion

BIP 322 enables the encoding of complex, smart authorization logic into the lowest level of the signature scheme. Any logic that can be defined within a bitcoin script, can be used to encode the rules around the production signatures. This includes multi-sig, timelocked and many other configurations. These rules are guaranteed by the same process that provides assurance over the Bitcoin currency.

By defining a signature suite specification for BIP322 signatures, these constructs could be used as a verification method defined within a DID document and applied to all of the usecases we know and love around DIDs and VCs. To quote Smarter Signatures (RWOT2): > today's simplistic signatures are just the start; they can be improved, to create more powerful and more complex signatures that can truly be better and smarter."[2]

- [1] Smart Signatures, 2015, https://github.com/WebOfTrustInfo/rwot1-sf/blob/master/final-documents/smart-signatures.pdf
- [2] Smarter Signatures, RWOT2, 2016, https://github.com/WebOfTrustInfo/rwot2-id2020/blob/master/final-documents/smarter-signatures.pdf
- [3] BIP 322: Generic Signed Message Format, 2018, https://github.com/bitcoin/bips/blob/master/bip-0322.mediawiki
- [4] BIP 322 Signatures (Jupyter Notebook Examples) https://github.com/LegReq/bip0322-signatures