Rabin Signatures in Bitcoin Cash

Owen Vaughan – Senior Researcher, nChain

In this article we show how arbitrary messages can be signed and verified directly in Bitcoin Cash script without introducing new opcodes (such as the proposed `OP_CHECKDATASIG` (also known as `OP_DATASIGVERIFY`)). By utilising the simple algebraic structure of Rabin signatures, any piece of data placed in a transaction can be signed - even if it originates from outside the Bitcoin Cash blockchain.

Rabin Signatures

The Rabin Digital Signature (RDS) algorithm is a relatively unknown early example of a digital signature scheme developed by Michael Rabin in 1979 [1]. Its security relies on the key observation that calculating a modular square-root is as hard as integer factorisation. For this reason, it shares many features with RSA but also has some interesting differences that are worth exploring.

The Rabin cryptosystem setup is similar to RSA. Key generation is initiated by identifying two primes p and q of similar magnitude. But, unlike RSA there is an additional requirement that $p \equiv 3 \bmod 4$ and $q \equiv 3 \bmod 4$. The combination (p, q) together form the private key with the corresponding public key $n = p \cdot q$.

To sign a message m a padding U is identified such that the hash $H(m||U)$ is a quadratic residue modulo n. In other words, there exists an S such that

$$S^2 \equiv H(m||U) \bmod n \ . \tag{1}$$

The combination (S, U) is the Rabin signature.

Forging the signature by finding the square-root of the right-hand-side is as hard as the integer factorisation of n itself. On the other hand, if p and q are known then by the Chinese remainder theorem

$$S^2 \equiv H(m||U) \bmod p \qquad \text{and} \qquad S^2 \equiv H(m||U) \bmod q \ . \tag{2}$$

Since we have chosen $p \equiv 3 \bmod 4$ and $q \equiv 3 \bmod 4$ the simplified version of the Tonelli-Shanks algorithm can be used to calculate

$$S \equiv \left[\left(p^{q-2} \cdot H(m||U)^{\frac{q+1}{4}} \bmod q \right) \cdot p + \left(q^{p-2} \cdot H(m||U)^{\frac{p+1}{4}} \bmod p \right) \cdot q \right] \bmod n \ . \tag{3}$$

To verify the signature is valid one checks that (1) holds. This is true if and only if there exists an integer λ in the range $0, \dots, n-1$ such that

$$S^2 = H(m||U) + \lambda \cdot n \ . \tag{4}$$

The factor λ may be safely included with the signature, so the information that can be made public is the combination (S, λ, U). In this case, to verify the signature one checks that (4) holds, which can be easier to implement than (1).

There are three interesting features of the Rabin signature scheme:

a) Signature generation is computationally expensive, whereas verification of the signature is computationally cheap.
b) The security of the signature relies only on the hardness of integer factorisation. As a result, Rabin signatures are existentially unforgeable (unlike RSA).
c) The hash function H must be of a similar magnitude to the public key n.

To ensure a 128-bit security level a Rabin (or RSA) public key n must be of the order 3072-bits. This is notably larger than the 256-bits needed for a similar key in elliptic curve cryptography. The digest of the hash function H and the S part of the signature must therefore also be of the order 3072-bits.

Implementation in Bitcoin Cash

The simple algebraic structure of the verification equation (4) lends itself to implementation in Bitcoin Cash script, which only allows for certain basic arithmetic operations. The computationally expensive generation of the signature itself is calculated off-block.

One important application is the ability to verify the signature of an arbitrary message m. This message could comprise of data originating outside the Bitcoin Cash blockchain. Compare this to the existing opcode `OP_CHECKSIG` which verifies only that a specific type of message, namely the transaction hash, has been signed by an ECDSA signature.

Consider a redeem script of the form

OP_DUP OP_HASH160 $<H_{160}(n)>$ OP_EQUALVERIFY OP_MUL OP_SWAP OP_2 OP_ROLL OP_CAT FUNC_HASH3072 OP_ADD OP_SWAP OP_DUP OP_MUL OP_EQUAL

where n is the public key of the signer. This will evaluate to `TRUE` if and only if it is provided with the input

$$<S>\ <U>\ <m>\ <\lambda>\ <n>$$

where m is an arbitrary message and (S, λ, U) is a valid Rabin signature of that message.

Alternatively, one may include $<m>$ in the redeem script if the signature of a predetermined message is required.

There are two barriers to the direct implementation of the above redeem script, however these can both be overcome without the need to introduce new opcodes.

i) Bitcoin Cash script does not have a hash function with 3072-bit digest, which we have denoted by `FUNC_HASH3072` as a placeholder.
ii) The arithmetic operations `OP_MUL` and `OP_ADD` are restricted to 32-bit inputs. This conflicts with the 3072-bit integers in the Rabin signature scheme.

To overcome (i) we can project a 256-bit hash digest onto 3072-bits. For example, we may first apply `OP_HASH256`, split the result into 12 chunks, hash each chunk again with `OP_HASH256,` and finally concatenate the result.

A solution to (ii) is to split each integer into 32-bit chunks and then apply Karatsuba's algorithm. Improvements in efficiency may be made by verifying each chunk on both sides of the equation on the fly. Once the data has been massaged into a suitable form, such a process will result in 96 loops for the main algorithm, which must be unpacked in Bitcoin Cash script. Note that this solution would require the removal of the limit of 201 opcodes.

An alternative solution to (ii) is to raise the Bitcoin Cash arithmetic bit-length to account for larger integers. In this case due care must be taken to engineer a safe solution.

Conclusion

By utilising the properties of Rabin signatures we have seen how an arbitrary message can be signed, and how the signature can be verified directly in Bitcoin Cash script without introducing new opcodes. All computationally expensive operations (key generation, signature construction) are performed off-block. Only the simple step of verifying that (4) holds is performed within script.

The existentially unforgeable property of the solution allows extra functionality to be added to the Bitcoin Cash platform without compromising the security of the network, nor changing the core protocol itself.

We will continue to develop this solution using Rabin signatures, and will seek to collaborate with others on this work. nChain does not intend to seek patent protection for its work on this solution; instead, nChain will publish its work in this area for public review and usage.

References

[1] Michael O. Rabin, “Digitalized signatures and public-key functions as intractable as factorization”, Technical Report 212, MIT Laboratory of Computer Science, Cambridge, 1979.