Mimblewimble and Other Spells

Hector Escobedo

September 20, 2018

Initium

Tom Elvis Jedusor

On August 2nd, 2016, an unknown person going by the name "Tom Elvis Jedusor"[1] published a document detailing a new blockchain structure called Mimblewimble with radical differences from that of Bitcoin. A number of Bitcoin developers were intrigued by this mysterious innovation, and some began to elaborate on it further. Only the bare minimum of details were included in the document, but these proved to be workable.

[1] In the French translation of *Harry Potter*, this is Voldemort's real name.

Neverending

The problem with the blockchain

As the blockchain grows, the amount of data and time required to verify it grows linearly with the number of blocks. This makes it harder to start up and synchronize new mining nodes.

Additionally, transactions and their amounts can be traced between addresses. This makes preserving privacy and anonymity harder.

Each of these problems have attracted a number of proposed solutions, such as checkpointing or ring signatures (found in Monero). Mimblewimble offers a solution to both in one fell swoop, using advanced cryptography.

Simplicity

Ultimate sophistication

Why the desire for a new system? Simple. There is more data stored in the Bitcoin blockchain than is actually needed for it to work as a payment system.

If you think about it, all a payment system needs to do is make sure that there is no money being created out of thin air, and that whatever person A transfers to person B can subsequently be spent by person B. Mimblewimble satisfies these properties quite elegantly, making the amount of data that we need to verify transactions much smaller.

Novelty

A high order incantation

Mimblewimble differs from the conventional blockchain in a few important ways.

1. No addresses: Transactions are created interactively between sender and recipient.
2. Aggregation: Every block is one giant transaction and individual user transactions cannot be distinguished.
3. Private amounts: All inputs and outputs are hidden.
4. Cut-through: The entire chain can be summarized and verified with relatively little data.

Cryptographic wizardry

Quod erat demonstrandum

Get ready for some math.

Elliptic curves over finite fields

A finite field F_q of order (size) q is a special mathematical structure that behaves “nicely” with respect to addition and multiplication and only has a finite number of elements.

An elliptic curve over a finite field is defined by an equation of the form

$$y^2 = x^3 + ax + b,$$

where $a, b, x, y \in F_q$ and a and b are fixed, so a pair (x, y) is an element of the elliptic curve group E if it satisfies this equation. Very importantly, the sum of any two points in E always remains in E.

Discrete logarithms

Plus some operations

From now on we will be working entirely within E. Remember that we can only add points in E, not multiply them together. However, we can define the nth multiple of a point p as simply p added to itself n times. Denote this as np. This is also known as a "scalar multiplication" operation, and is easy to compute. Also, we may refer to doubling a point, which is just $2p = p + p$.

Given two points p and q, the discrete logarithm problem is finding the integer k such that $p = kq$. If p really is a multiple of q, then k exists, but there is no efficient way to find it after the fact without a quantum computer.

Elliptic curve cryptography

Elliptic curve cryptography (ECC) is based on the discrete logarithm problem. You can replace both RSA and regular Diffie-Hellman with systems based on a suitable elliptic curve group, and the keys will be much smaller for an equivalent level of security to boot!

This is why ECC is used for Bitcoin signatures. However, if we can determine an elliptic curve group that contains a special kind of subgroup, we can do even more than that.

Special subgroups

Every point p generates a cyclic subgroup P, which is the set of its multiples until you return to $p = np$,[2] where n divides the order of E. In fact, for a given subgroup, we may have more than one generator. Let g and h be generators of the subgroups $G, H \subset E$, where both have prime order and we do not know the discrete logarithm $g = kh$ or $h = jg$. This is to rule out any funny business later.

If E already has prime order, the only subgroup is itself. So this won't work. Make sure its order is a composite number!

[2] See what I did there?

Pedersen commitment

It turns out that using only plain ECC, we can construct what is known as a homomorphic, binding commitment scheme, specifically a Pedersen commitment scheme.[3]

How we do this is to take v as our value to commit to, a random number r as the blinding factor, with g, h generators of G and H, respectively, and we get the unforgeable commitment $c = vg + rh$, which is a point in E.

Using only c, we cannot find v or r if the discrete logarithm problem is hard. But wait a second, why is it homomorphic?

[3] This is also used in Monero.

Additive homomorphism

$$(v_1g + r_1h) + (v_2g + r_2h) = (v_1 + v_2)g + (r_1 + r_2)h$$

We can add two commitments together to get a commitment to the sum of their values!

Okay, how does this help? Well, every transaction must commit to zero. This means there has been zero net theft or inflation, which is what makes it work as a payment system.

Example transaction

Lets say that Alice wants to send the amount of 25 to Bob. She reveals the amount and her secret blinding factor A to Bob. He then generates his own blinding factor B and creates a new commitment that balances out.

$$(25g + Ah) + (-25g + Bh) = 0g + (A + B)h$$

The new blinding factor $A + B$ is called the excess value, and serves as Bob's private key to sign the transaction. Assuming there are no elements in common between G and H (practically speaking, nearly guaranteed), this ensures that the sum value is zero, or it would not be a valid key. (Check this?)

Bilinear pairing

This is an advanced topic, but the basis of it is actually quite simple. Take two cyclic groups, G_1 and G_2, both with prime order. A bilinear pairing is a function

$$f : G_1 \times G_2 \to G_T$$

with the special property that

$$f(aP, bQ) = f(P, Q)^{ab}$$

for all points P and Q. The domain G_T is a multiplicative group rather than an additive group, which is why the multiples become exponents.

Sinking signatures

A sinking signature is specially constructed so that a signature for the latest block can intentionally be used to forge a valid signature for all previous blocks.

This is what we use the bilinear pairing for, by the way. I think.

Range proofs

Do you know what happens when you add two very large numbers on a computer? You cause an overflow error. (Also, transactions of zero value are bad.) To prevent this from occurring in our system without revealing the actual amount of each transaction, we use a cryptographic scheme called a range proof. Like the name implies, this is a mathematical assurance that the value lies in a certain range.

If $2^{64} - 1$ is the maximum value that we do not want to overflow, then a range of $0 < v \leq 2^{44}$ is sufficient, because you would have to add up at least $2^{20} = 1048576$ values to overflow, making the transaction size astronomically large, and thereby it is infeasible that anyone would accept it.

Imagining the current future

So let's imagine that you want to receive donations or payments with Mimblewimble. Instead of posting a public address, you tell people to directly send you half-complete transactions. This could be over any secure data connection. You then complete the transaction and send it to miners.

Note that this is interactive yet not necessarily immediate. It may make using a cold wallet somewhat more difficult, because you can't transfer directly to it without access to the secret blinding factor.

Verification

One source estimates that a 50,000 block chain could be cut-through to produce a compact chain of only about 300 blocks. In other words, instead of $O(n)$ time and data necessary for verifying the chain, where n is the full number of blocks, we only need $O(\log n)$.

Each block is reduced to only a “kernel” of only a few kilobytes. No more need for light wallets: nearly any device can trustlessly verify the expected required work!

The one drawback

Can't have your cake and eat it too

The drawback of Mimblewimble? It does not support scripts. Therefore a range of functionality we are used to in Bitcoin, such as time locking outputs, conditional payments, and cross-chain atomic swaps are not possible without further work. It is a payment system only.

However, the Grin developers are apparently working on this issue, with some success.

The community

Definitely not muggles

You may not be surprised to learn that the developer community for Mimblewimble projects primarily consists of the apparently non-negligible intersection between cryptographers and *Harry Potter* fans.

Keep in mind that this whole idea is now barely two years old.

Grin

The Bitcoin Cash of Mimblewimble. Believes the chain will scale as needed.

Uses a novel Cuckoo Cycle POW. Has a testnet. Unbounded emission curve. Open source community owned. Camelcases MimbleWimble.

BEAM

The Bitcoin Core of Mimblewimble. Does not believe the chain will scale as needed.

Uses a variant of Equihash POW. Testnet launching soon. Bounded emission curve. Private company owned for the time being. Spells Mimblewimble properly.

Some closing thoughts

Words of wisdom

“Anyway, my mum always said the things we lose have a way of coming back to us in the end, if not always in the way we expect.” (Luna Lovegood)

“For where your treasure is, there your heart will be also.” (Matthew 6:21)

Finis

I open at the close