A Model-Driven Analysis of Mimblewimble Security Properties and its Protocol Implementations

Un Análisis Basado en Modelos de las Propiedades de Seguridad de Mimblewimble y las Implementaciones del Protocolo

Uma Análise Orientada a Modelo das Propriedades de Segurança do Mimblewimble e suas Implementações de Protocolo

Adrián Silveira [1], Gustavo Betarte [2], Maximiliano Cristiá [3], Carlos Luna [4]

Recibido: 29/05/2023 Aceptado: 29/05/2023

Summary. - Mimblewimble is a privacy-oriented cryptocurrency technology that provides security and scalability properties that distinguish it from other protocols. Mimblewimble's cryptographic approach is based on Elliptic Curve Cryptography which allows verifying a transaction without revealing any information about the transactional amount or the parties involved. Mimblewimble combines Confidential transactions, CoinJoin, and cut-through to achieve a higher level of privacy, security, and scalability. In this work, we present and discuss these security properties and outline the basis of a model-driven verification approach to address the certification of the correctness of the protocol implementations. In particular, we propose an idealized model that is key in the described verification process. Then, we identify and precisely state the conditions for our model to ensure the verification of relevant security properties of Mimblewimble. In addition, we analyze the Grin and Beam implementations of Mimblewimble in their current state of development. We present detailed connections between our model and their implementations regarding the Mimblewimble structure and its security properties.

Keywords: security; formal verification; mimblewimble; idealized model; cryptocurrency.

Resumen. *- Mimblewimble es una criptomoneda orientada a la privacidad con propiedades de seguridad y escalabilidad que la distingue de otras criptomonedas. Mimblewimble está basado en Criptografía de Curvas Elípticas lo que permite verificar la validez de las transacciones sin revelar información alguna sobre el monto y las partes involucradas. Mimblewimble combina transacciones confidenciales y las técnicas de CoinJoin y cut-through para alcanzar mayor nivel de privacidad, seguridad y escalabilidad. En este trabajo, presentamos y discutimos estas*

[1] Profesor Asistente, Facultad de Ingeniería - Universidad de la República, Montevideo, Uruguay, adrians@fing.edu.uy ORCID iD: https://orcid.org/0000-0001-8742-046X

[2] Profesor Titular, Facultad de Ingeniería - Universidad de la República, Montevideo, Uruguay, gustun@fing.edu.uy, ORCID iD: https://orcid.org/0000-0002-6863-1082

[3] Profesor, CIFASIS y Universidad Nacional de Rosario, Rosario, Argentina, cristia@cifasis-conicet.gov.ar, ORCID iD: https://orcid.org/0000-0001-9163-2609

[4] Profesor Agregado, Facultad de Ingeniería - Universidad de la República, Montevideo, Uruguay, cluna@fing.edu.uy, ORCID iD: https://orcid.org/0000-0002-9985-5927

propiedades de seguridad y describimos un enfoque basado en la verificación de modelos para alcanzar la certificación de la corrección de las implementaciones del protocolo. En particular, proponemos un modelo idealizado que es clave en el proceso de verificación descrito. Luego, identificamos y describimos precisamente las condiciones que nuestro modelo debe cumplir para asegurar las verificaciones de propiedades de seguridad relevantes de Mimblewimble. Además, analizamos el estado actual de sus dos más importantes implementaciones, Grin y Beam. Finalmente, presentamos conexiones detalladas entre nuestro modelo y las implementaciones en relación con la estructura de Mimblewimble y sus propiedades de seguridad.

Palabras clave: *seguridad; verificación formal; mimblewimble; modelo idealizado; criptomoneda.*

***Resumo**. - Mimblewimble é uma tecnologia de criptomoeda orientada para a privacidade que fornece propriedades de segurança e escalabilidade que a distinguem de outros protocolos. A abordagem criptográfica do Mimblewimble é baseada na Elliptic Curve Cryptography, que permite verificar uma transação sem revelar nenhuma informação sobre o valor da transação ou as partes envolvidas. O Mimblewimble combina transações confidenciais, CoinJoin e cut-through para alcançar um nível mais alto de privacidade, segurança e escalabilidade. Neste trabalho, apresentamos e discutimos essas propriedades de segurança e delineamos a base de uma abordagem de verificação orientada por modelo para abordar a certificação da correção das implementações de protocolo. Em particular, propomos um modelo idealizado que é chave no processo de verificação descrito. Em seguida, identificamos e declaramos com precisão as condições de nosso modelo para garantir a verificação das propriedades de segurança relevantes do Mimblewimble. Além disso, analisamos as implementações Grin e Beam do Mimblewimble em seu estado atual de desenvolvimento. Apresentamos conexões detalhadas entre nosso modelo e suas implementações em relação à estrutura Mimblewimble e suas propriedades de segurança.*

Palavras-chave: *segurança; verificação formal; mimblewimble; modelo idealizado; criptomoeda.*

1. Introduction. - A cryptocurrency is a digital currency that can be exchanged online for goods and services. It can be converted into cash through a cryptocurrency exchange and vice versa. Many cryptocurrencies work using a technology called blockchain, a distributed ledger of transactions that is duplicated and distributed across the nodes of a computer network. A defining feature of cryptocurrencies is that there is no central trusted authority. The ledger is maintained using a consensus-based validation protocol where transactions are constructed in a peer-to-peer fashion and broadcast to the entire set of participants who work to validate them and build blocks. Therefore, the consensus algorithm is what decides which is the following block to be appended to the blockchain. This decentralized mechanism is devised to achieve reliability in a network consisting of unreliable nodes. In what follows, we present relevant security aspects of cryptocurrencies and discuss the importance of applying formal methods to verify their implementations.

1.1 Cryptocurrency Security. - Cryptocurrency protocols are a valuable target for several attacks since they deal with virtual money. Irreparable losses of money and credibility have been caused because of several attacks against cryptocurrency systems [3]. Security and confidentiality are now much more important in this situation. For this reason, the cryptocurrency community is looking for techniques and approaches that can reduce the likelihood of successful attacks. One such approach is the application of formal methods to software implementation. In particular, interest in formally certified implementations and formal proofs has increased [14].
Mimblewimble (MW) is a privacy-oriented cryptocurrency technology with scalability and security that set it apart from similar technologies. In MW, unlike Bitcoin, there is no such concept as an address, and all the transactions are confidential. The method used in this work is based on formal software verification, and it aims to formally verify the fundamental mechanisms of MW and its implementations [7, 6]. Security issues are explored on an idealized model of the MW protocol. Such model delivers a realistic environment while abstracting away the specifics of any particular implementation. Then, verification should be performed on more detailed models, where low-level mechanisms are specified. Finally, it must be proved that the low-level model implements the idealized model.

1.2 Related Work and Contributions. - The cryptocurrency community is interested in applying formal methods to guarantee correctness and properties over their protocol constructions. For instance, Idelberger et al. [8] proposes the use of defeasible logic frameworks such as Formal Contract Logic for the description of smart contracts. However, the authors did not analyze the necessary conditions the cryptocurrency protocols should satisfy to guarantee security properties. Boyd et al. presented a blockchain model in Tamarin [2], which is useful for analyzing certain blockchain based protocols.

We believe that cryptocurrency protocols should undergo a formal security assessment in order to prove and guarantee security properties from a formal point of view. A complete formal model of the cryptocurrency is crucial. For instance, Ruffing et al. [15] show an attack against Zerocoin [12] exposing the lack of an important missing property in the formal security analysis of the cryptocurrency.
The goal of the present work was to identify and analyze the main components of the MW protocol in order to build an idealized model and verify security properties. The specific objectives were to i) identify and specify the based schemes and protocols of MW, ii) identify and define the main components of our model, iii) define and verify relevant security properties and iv) compare our model with the most popular implementations of MW: Grin and Beam.

The proposed idealized model constitutes the main contribution together with the analysis of the essential properties it is shown to verify. This idealized model constitutes the basis of a model-driven verification approach to address the certification of the correctness of the protocol's implementation. Furthermore, this model will enable the analysis of several attacks to prove their existence or absence over the model.

This work is organized as follows. Section 2 provides a brief description of MW. Section 3 defines the schemes and protocols our model is based on. Section 4 describes the building blocks of a formal idealized model of MW. Section 5 analyzes protocol and security properties over the model. Then, Section 6 analyzes the Grin and Beam implementations of MW in their current state of development. Final remarks and directions for future work are presented in Section 7.

2. The Mimblewimble protocol. - In August 2016, someone called "Tom Elvis Jedusor" (the french name for Voldemort in Harry Potter) posted a text file on the IRC Channel describing a cryptocurrency protocol with a different approach from BitCoin. This article titled 'Mimblewimble' [9] addressed some privacy concerns and the ability to compress the transaction history of the chain without losing validity verification. In October 2016, since this document left some questions open, Andrew Poelstra published a paper [13] where he described, in more detail, the design of a blockchain based on MW.

Next, we describe how money transfer is carried out on the MW protocol. Suppose Alice and Bob agree on a money transfer. Alice wants to send v coins to Bob. They communicate off-chain and create the MW transaction, including the transaction amount v.

Then, the transaction should be added to a block which is distributed across the nodes of a computer network to be added to the blockchain.

In MW, transactions are Confidential transactions [11]. A transaction allows a sender (Alice) to encrypt the amount v of bitcoins by using blinding factors. Only the two parties involved know the amount of bitcoins being exchanged in a confidential transaction. However, for anyone observing the transaction, it is possible to verify its validity by comparing the number of inputs and outputs; if both are the same, the transaction will be considered valid. Such procedure ensures that no money has been created from nothing and is key in preserving the system's integrity. In MW transactions, the recipient (Bob) randomly selects a range of blinding factors provided by the sender, which are then used as proof of ownership by the receiver.

3. Schemes and Protocols. - A commitment scheme is a cryptographic primitive that allows a player in a protocol to choose a value and commit to his choice such that he can no longer change his mind. The value is kept hidden from others with the ability to reveal the committed value later. In MW transactions, the transaction amounts and blinding factors are hidden in Pedersen commitments. We say that it is hard (in terms of complexity) for someone observing the transaction to know the transaction amount or the blinding factor. In addition, since the transaction amounts are hidden, it should be possible to verify that the values are positive without revealing any information about them. Range proofs should be provided to guarantee the transactional amount lies in some range. Moreover, a transaction contains a signature to guarantee it was honestly constructed. Next, we define the schemes and protocols our model is based on.

3.1 Commitment Scheme. - A commitment scheme [5] is a two-phase cryptographic protocol between two parties: a sender and a receiver. At the end of the commit phase, the sender is committed to a specific value that he cannot change later, and the receiver should have no information about the committed value. A non-interactive commitment scheme [4] can be defined as follows:

Definition 1 (Non-interactive Commitment Scheme) A non-interactive commitment scheme $\zeta(Setup, Com)$ consists of two probabilistic polynomial time algorithms, $Setup$ and Com, such that:

- Setup generates public parameters for the scheme depending on the security parameter λ.
- Com is the commitment algorithm: $Com : M \times R \rightarrow C$, where M is the message space, R the randomness space and C the commitment space. For a message $m \in M$, the algorithm draws uniformly at random $r \leftarrow R$ and computes the commitment $com \leftarrow Com(m, r)$.

We have simplified the notation, but it is essential to remember that Com, M, R, and C depend on the parameters generated by $Setup$.

We say the commitment algorithm is a linear function if:

$$\forall\, m_1, m_2 \in M, r_1, r_2 \in R{:}Com(m_1, r_1) + Com(m_2, r_2) = Com(m_1 + m_2, r_1 + r_2)$$

In other words, Com is additive in both parameters.

Transactions in MW are derived from Confidential transactions, which are enabled by Pedersen commitments with homomorphic properties over elliptic curves. We define the non-interactive Pedersen commitment scheme we will use in our model, based on *Definition 1*, as follows:

Definition 2 (Pedersen Commitment Scheme with Elliptic Curves) Let M and R be the finite field F_n and let C be the set of points determined by an elliptic curve C of prime order n. As in *Definition 1*, the probabilistic polynomial time algorithms are defined as:

- Setup generates the order n (dependent on the security parameter λ) and two generator points G and H on the elliptic curve C of prime order n whose discrete logarithms relative to each other are unknown.
- $Com(v, r) = r.G + v.H$, with v the transactional value and r the blinding factor chosen randomly in F_n.

Each MW transaction contains a list of range proofs of the transactional values. Next, we define the range proof scheme we will analyze in our model.

3.2 Range Proof Scheme. - Range proofs aim at proving that a secret value is in a particular range without revealing the value. Transactions in MW contain a list of range proofs proving that the transactional values are positive and less than a specific upper bound to avoid overflow errors. We define a non-interactive zero-knowledge (NIZK) range proof scheme from a commitment scheme as follows:

Definition 3 (NIZK Range Proof Scheme) Let $\zeta(Setup, Com)$ be a Pedersen commitment scheme as in *Definition 2*. Let P be the range proof space for the values $v \in M$ such that v lies in some range $[a, b]$. A non-interactive zero-knowledge range proof scheme $\eta\ (Prove, Verify)$ consists of two probabilistic polynomial time algorithms, $Prove$ and $Verify$, such that:

- $Prove : M \times R \times C \rightarrow P$, which receives a value v, the random value r and the commitment $c = Com(v, r)$ and computes the range proof for the value v.
- $Verify : C \times P \rightarrow bool$, that given a commitment value and a range proof, decides if the value is in the range.

Notice that the Prove algorithm computes a zero-knowledge proof to the commitment to verify that the committed value is in a particular range. In other words, the guarantee enables a prover to convince a verifier that the statement holds without revealing any information about the secret value. In addition, since the transactional values should be positive, the amount should lie in $[0, b]$ such that b is large enough to guarantee privacy concerns.

3.3 Schnorr Signature Protocol. –The construction of the MW transaction is made off-chain by the parties. For simplicity, we shall work with a signature protocol between two parties, but this can be generalized to multi-parties.

During the transaction construction Alice needs to verify Bob's Schnorr signature. Schnorr signature protocols can be applied over any group where discrete logarithm is hard, in our case, over an elliptic curve C. Next, we define the Schnorr signature protocol used by them during the transaction construction. The message m can be the empty string.

Definition 4 (Schnorr Signature Protocol) Let C be an elliptic curve of prime order n with generator G. Let $hash : \{0,1\} * \rightarrow F_n$ be a cryptographic hash function over the finite field F_n. Alice secretly knows $k_A \in F_n$ whose public key is $K_A = k_A . G$

Signing

The following steps are followed to create a signature on a message $m \in \{0,1\}^*$:

1. Alice chooses nonce $n_A \leftarrow \$ F_n$ where $\leftarrow \$$ denotes that n_A is drawn uniformly at random from F_n.
2. She computes public key $N_A = n_A . G$
3. She computes $e = hash\ (N_A \mid K_A \mid m)$ and $s_A = n_A + e . k_A$ where $\mid$ denotes concatenation and N_A, K_A are represented as a bit string.
4. The signature σ is defined as follows:

$$\sigma = (s_A, N_A), \textit{with public key } K_A$$

Validation

A signature $\sigma = (s_A, N_A)$ is valid if the following holds:

$$s_A.G = N_A + e.K_A$$

Each MW transaction contains a signature σ made by the parties during the transaction construction, which can be seen as a Schnorr multi-signature.
Next, a Schnorr signature protocol aggregation is defined according to our model.

Definition 5 (Schnorr Signature Protocol Aggregation) Let C be an elliptic curve of prime order n with generator G. Let $hash$: $\{0,1\}^* \rightarrow F_n$ be a cryptographic hash function over the finite field F_n. Alice and Bob secretly know $k_A, k_B \in F_n$ whose public keys are $K_A = k_A . G$ and $K_B = k_B . G$ respectively.

Signing

The following steps are followed to create a multisignature on a message $m \in \{0,1\}^*$:

1. Alice and Bob choose nonces $n_A, n_B \leftarrow \$ F_n$ respectively
2. They compute public keys $N_A = n_A . G$ and $N_B = n_B . G$

3. They compute $e = hash\ (N_A + N_B \mid K_A + K_B \mid m)$ and respectively compute:

$$s_A = n_A + e.k_A \qquad\qquad s_B = n_B + e.k_B$$

4. The aggregate signature σ is defined as follows:

$$\sigma = (s_a + s_B, N_A + N_B)$$

with the aggregate public key $K_A + K_B$

Validation
A signature $\sigma = (s_a + s_B, N_A + N_B)$ is valid if the following holds:

$$(s_A + s_B).G = N_A + N_B + e.(K_A + K_B) \tag{1}$$

Next, we show that a signature σ honestly constructed will be valid.
If we consider the signing process, we know that:

$$s_A = n_A + e.k_A \text{ and } s_B = n_B + e.k_B$$

By applying algebraic properties on elliptic curves, the left term on the equality 1 can be written as:

$$(s_A + s_B).G = (n_A + e.k_A).G + (n_B + e.k_B).G = n_A.G + e.k_A.G + n_B.G + e.k_B.G =$$

$$n_A.G + n_B.G + e.(k_A.G + k_B.G)$$

So, if we substitute the left term on the equality 1, we have:

$$n_A.G + n_B.G + e.(k_A.G + k_B.G) = N_A + N_B + e.(K_A + K_B)$$

The above equality holds because:

$$N_A = n_A.G,\ K_A = k_A.G \text{ and } N_B = n_B.G,\ K_B = k_B.G$$

(N_A, K_A) are Alice's public keys, and (N_B, K_B) are Bob's public keys. Since we are working over the elliptic curve C where the discrete logarithm is hard, the only ones who know the private keys (n_A, k_A) and (n_B, k_B) are Alice and Bob respectively.

4. Idealized Model. - The essential elements of our model are transactions, blocks, and chains. Each node in the blockchain maintains a local state. The main components are the local copy of the chain and the set of transactions waiting to be validated and added to a new block. Moreover, each node keeps track of unspent transaction outputs (UTXOs). Next, we define all the elements which compose our idealized model.

4.1 Transactions. - Given two fixed generator points G and H on the elliptic curve C of prime order n (whose discrete logarithms relative to each other are unknown), we define a single transaction as follows:

Definition 6 (Transaction) A single transaction t is a tuple of type:

$$Transaction \overset{\text{def}}{=} \{i : I^*,\ o : O^*, tk : TxKernel, tko : KOffset\}$$

with X* representing the lists of elements of type X and where:

- $i = [c_1, \ldots, c_n]$ and $o = [o_1, \ldots, o_m]$ are the lists of inputs and outputs. Each input c_i and output o_i are points over the curve C and they are the result of computing the Pedersen commitment $r.G + v.H$ with r the blinding factor and v the transactional value in the finite field F_n.

- $tk = \{rp, ke, \sigma\}$ is the transaction kernel where:
  – $rp = [rp_1, \ldots, rp_m]$ is a list of range proofs of the outputs. The j th item rp_j in rp corresponds to the j th item o_j in o
  – ke is the transaction excess represented by $(\sum_1^m r' - \sum_1^n r - tko).G$
  – σ is the kernel Schnorr signature (for simplicity, fees are left aside)
- $tko \in F_n$ is the transaction kernel offset.

Inputs are previous transaction outputs. The transaction kernel offset will be used to construct a block to satisfy security properties.
The ownership of a coin is given by the following definition:

Definition 7 (Ownership) Given a transaction t, we say S owns the output o if S knows the opening (r, v) for the Pedersen commitment $o = r.G + v.H$.
The strength of this security definition is directly related to the difficulty of solving the logarithm problem. If the elliptic curve discrete logarithm problem in C is hard, then given a multiple Q of G, it is computationally infeasible to find an integer r such that $Q = r.G$.
It is essential to notice that the sender and the receiver do not learn their respective blinding factors during the construction of the transaction. Instead, they build a Schnorr signature that is used to guarantee the authenticity of the transaction's excess value.
We say that a transaction is valid if the following property holds:

Property 1 (Valid Transaction) A transaction t is valid (valid transaction(t)) if t satisfies:

i. The range proofs of all the outputs are valid.
ii. The transaction is balanced.
iii. The kernel signature σ is valid for the excess.

These three properties have a straightforward formalization in our model.
The first property we should guarantee is that all the range proofs of all the outputs are valid.

Definition 8 (Valid Range Proof Outputs Transaction) Let $t = \{i, o, tk, tko\}$ be a transaction as in *Definition 6*, with transaction kernel $tk = \{rp, ke, \sigma\}$ where $o = [o_1, \ldots, o_m]$ is the list of outputs and $rp = [rp_1, \ldots, rp_m]$ is the list of the range proof outputs. Let $\eta(Prove, Verify)$ be a NIZK scheme as in *Definition 3* with P the range proof space where $rp_j \in P$ proves that o_j lies in the range $[0, 2^n]$ where n is small enough to not cause overflow errors. We say all the range proof output transactions are valid if: for all $rp_j \in rp, Verify(o_j, rp_j) = true$.
The list of range proof outputs provides proof that each transactional output is positive without revealing further information.
The second property is defined as follows:

Definition 9 (Balanced Transaction) A transaction $t = \{i, o, tk, tko\}$, with transaction kernel $tk = \{rp, ke, \sigma\}$, is balanced if the following holds:

$$\sum_{o_j \in o} o_j - \sum_{c_j \in i} c_j = ke + tko.G$$

A balanced transaction guarantees no money is created from thin air.
The kernel signature σ is a Schnorr signature aggregation with the kernel excess ke as the public key.
Note that, for simplicity during the transaction construction, in *Definition 5* we consider a Schnorr signature aggregation between two parties; however, once the transaction is constructed, it is not

necessary to know the parties involved.

Definition 10 (Valid Signature for the kernel excess) Let $t = \{i, o, tk, tko\}$ be a transaction as in *Definition 6* with transaction kernel $tk = \{rp, ke, \sigma\}$ where:

- rp is a list of range proofs of the outputs.
- ke is the transaction excess.
- $\sigma = (s, N)$ is the kernel Schnorr signature aggregation as in *Definition 5* on the empty string m.

We say the kernel signature σ is valid with public key ke if the following holds:
$s.G = N + e.ke$ such that $e = hash(N \mid ke)$

4.2 Aggregate Transactions. - A single transaction can be seen as the sending of money between multiple parties. An aggregate transaction represents many transactions.

Definition 11 (Aggregate Transaction) An aggregate transaction tx is a tuple of type:

$$TransacAgg \stackrel{\text{def}}{=} \{i : I^*,\ o : O^*, tks : TxKernel^*, tko : KOffset\}$$

Transactions can be merged non-interactively to construct an aggregate transaction. This process can be applied recursively to add more transactions into one aggregate transaction. The CoinJoin mechanism [10] makes it possible. It combines all inputs and outputs from separate transactions to form a single transaction, and the signatures can be composed by the parties. A Transaction Join can be understood as a simple way to perform CoinJoin with no composite signatures.

Definition 12 (Transaction Join) Given a valid transaction $t0$ and an aggregate transaction tx:

$$t_0 = \{i_0, o_0, tk_0, tko_0\}\ and\ tx = \{i, o, tks, tko\}$$

a new aggregate transaction can be constructed as:

$$tx = \{i_0 \mid\mid i, o_0 \mid\mid o, tk_0 \mid\mid tks, tko_0 + tko\}$$

The validity of the transactional parties guarantees the validity of an aggregate transaction during the construction process.

Lemma 1 (Invariant: CoinJoin Validity) Let $t0$ be a valid transaction and tx be a valid aggregate transaction. Let tx' be the result of aggregating $t0$ into tx as in *Definition 12*. Then, tx' is valid.

4.3 Unconfirmed Transaction Pool. - The unconfirmed transaction pool (*mempool*) contains the transactions which have not been confirmed in a block yet.

Definition 13 (Mempool) A mempool mp is a list of type:

$$Mempool \stackrel{\text{def}}{=} AggregateTransaction^*$$

4.4 Blocks and chain. - The genesis block Gen is a particular block since it is the first block ever recorded in the chain. Transactions can be merged into a block. A block is a significant transaction with aggregated inputs, outputs, and transaction kernels.

Definition 14 (Block) A Block b is either the genesis block Gen, or a tuple of type:

$$Block \overset{\text{def}}{=} \{i : I^*,\ o : O^*,\ tks : TxKernel^*, ko : KOffset\}$$

where:
- $i = [c_1, \dots, c_n]$ and $o = [o_1, \dots, o_m]$ are the lists of inputs and outputs of the transactions.
- $tks = [tk_1, \dots, tk_t]$ is the list of t transaction kernels.
- $ko \in F_n$ is the block kernel offset which covers all the transactions of the block.

In our model, a chain is defined as a list of blocks.

Definition 15 (Chain) A chain is a non-empty list of blocks:

$$Chain \overset{\text{def}}{=} Block^*$$

For a chain c and a valid block b, we can define a predicate $validate(c, b)$ representing the fact that is correct to add b to c. This relation must verify, for example, that all the inputs in b are present as outputs in c; in other words, they are UTXOs.

5. Properties. - Since we deal with virtual money, we should guarantee privacy and security properties on our idealized model. Next, we detail some relevant properties that can be verified in our model.

5.1 Protocol Properties. - The property of no coin inflation or zero-sum guarantees that no new funds are produced from thin air in a valid transaction. The property can be stated as follows.

Lemma 2 (No Coin Inflation) Let $t = \{i, o, tk, tko\}$ be a valid transaction with transaction kernel $tk = \{rp, ke, \sigma\}$. Then, the transaction excess only contains the blinding factor and the kernel offset.
The proof of this lemma ensures that, since a valid transaction is balanced, the difference between the output values and the input values is zero. It means that no coins are being created or destroyed in the transaction.
The cut-through process is an essential feature of MW. This process aims to erase redundant outputs that are used as inputs within the same block. Let C be a list of coins that appear as an output in the block b. If the same coins appear as an input within the block, then C can be removed from the list of inputs and outputs after applying the cut-through process. The only remaining data are the block headers, transaction kernels and UTXOs. After applying cut-through to a valid block b, ensuring that the resulting block b' is still valid is essential. We can say that the validity of a block should be invariant concerning the cut-through process.

Lemma 3 (Invariant: Cut-through Block Validity) Let $b = \{i, o, tks, ko\}$ be a block with i and o the list of inputs and outputs, $tks = [tk_1, \dots, tk_t]$ the list of transaction kernels and ko the block kernel offset. Let $b' = \{i', o', tks, ko\}$ be the resulting block after applying the cut-through process to b where: $i' = i \setminus (i \cap o)$ and $o' = o \setminus (i \cap o)$. Hence, if b is a valid block, then b' is valid too.

5.2 Privacy and Security Properties. - In blockchain systems, the notion of privacy is crucial: sensitive data should not be revealed over the network. In particular, it is desirable to ensure properties such as confidentiality, anonymity, and unlinkability of transactions. Confidentiality refers to the property of preventing other participants from knowing certain information about the transaction, such as the amounts and addresses of the owners. Anonymity refers to hiding the real identity of the parties involved in a transaction. In contrast, unlinkability refers to the inability to link different transactions of the same user within the blockchain.

5.2.1 Security properties of Pedersen commitments. - In MW transactions, input and output amounts are hidden in Pedersen commitments. We have stated that the Pedersen commitment is expected to satisfy hiding and binding properties. The former implies that the transaction amount of coins remains private for the rest of the network over time. The latter means that senders cannot change their commitments to a different transaction amount. If that were possible, it would mean that an adversary could spend coins that have already been committed to a UTXO, which would allow the creation of coins out of thin air.
We have shown that Pedersen commitments are perfectly hiding and computational binding [16]. In particular, we have proved the latter one.

5.2.2 Security properties of range proofs. - The goal of zero-knowledge proofs is to prove that a statement is true without revealing any information beyond the verification of the statement. In MW, we need to ensure that the amount is positive in every transaction so that users cannot create coins. Here, the hard part is to prove that without revealing the amount. In our model, the output amounts are hidden in the form of a Pedersen commitment and the transaction contains a list of range proofs of the outputs to prove that the amount is positive. This verification is performed as the first step of the validation of the transaction (Property 1).
We have stated that the range proof scheme (*Definition 3*) is expected to satisfy the properties of completeness, soundness, and zero-knowledge. Completeness states that if the statement holds for a witness v, the argument provided by the prover can convince the verifier. Soundness says that if the statement does not hold for a witness v, the prover cannot convince the verifier about the statement.
Zero-knowledge states that the argument does not leak any information about the witness, except whether the statement is true or false.
In our model, we have shown that a range proof scheme is perfect completeness, computational soundness and perfect zero-knowledge [16].

5.3 Unlinkability and Untraceability. - In our model, each node has a pool of unconfirmed transactions in the mempool. These transactions are waiting for the miners to be included in a block. We can distinguish two security properties of the transactions. Untraceability refers to the transactions in the mempool and unlinkability to the transactions in the block.
Untraceability states that for every transaction in the mempool, it is impossible to relate the transaction to the IP address of the node that originated it.
Unlinkability states that given a valid block b, it is computationally infeasible to know which input cancels which output.
In particular, for the unlinkability property, we have proved that for any valid block b and for any polynomial probabilistic time adversary A, the probability of A in finding a balanced transaction within b is negligible [16].

6. Implementations. - Because of its robust security, privacy and scalability, there are several implementations of Mimblewimble. In 2019, the first two practical implementations were launched: Grin and Beam. Although, there are some design and technical differences in both projects, they implement and extend the core of the MW protocol. Both Grin and Beam implementations address the main features of the MW protocol, namely the properties of confidentiality, anonymity and unlinkability comprised in our work. They transactions are based on confidential transactions.

Each input and output is in the form of a Pedersen commitment. All this data has a straightforward relation to our definition of transaction (*Definition 6*).
CoinJoin, as we have mentioned, combines inputs and outputs from multiple transactions into a single transaction in order to obfuscate them. In Grin, every block is a CoinJoin of all other transactions in the block.
Beam supports cut-through as we described above. In addition, Beam adds a scalable feature to eliminate all intermediate transaction kernels [1] in order to keep the blockchain as compact as possible.

7. Conclusions and Future Work. - We have introduced a formal analysis of the MW protocol that is the basis of a model-driven verification approach to address the certification of the correctness of a protocol's implementation. First, we have defined the main components of our idealized model: transactions, blocks and chain. Then, we have provided validity conditions to guarantee the correctness of the blockchain. We have stated precise conditions for a valid transaction and a valid block. Furthermore, we have defined and proved that the validity of a block is invariant with respect to the cut-through process and CoinJoin. We have also identified and precisely stated the conditions for our model to ensure the verification of relevant security properties of MW which is an important contribution of this work. Finally, we have analyzed and compared the Grin and Beam implementations in their current state of development, considering our model and its properties as a reference base.
This work contributes to analyzing the MW protocol's correctness and security properties over an idealized model beyond any particular implementation. Since cryptographic proofs are becoming increasingly error-prone and difficult to check, we plan to carry out a specification of our MW model using an interactive prover to provide automated verification of the model. Security goals and hardness assumptions shall be modeled to verify our stated security properties.

8. References

[1] Beam. Beam description. Comparison with classical MW, 2018.
Available online: https://docs.beam.mw/BEAM_Comparison_with_classical_MW.pdf (accessed on May 29, 2023).
[2] C. Boyd, K. Gjøsteen, and S. Wu. A Blockchain Model in Tamarin and Formal Analysis of Hash Time Lock Contract. In Bruno Bernardo and Diego Marmsoler, editors, 2nd Workshop on Formal Methods for Blockchains (FMBC 2020), volume 84 of OpenAccess Series in Informatics (OASIcs), pages 5:1–5:13, Dagstuhl, Germany, 2020. Schloss Dagstuhl–Leibniz-Zentrum für Informatik.
[3] Vitalik Buterin. Critical update re: Dao vulnerability, 2017.
Available online: https://blog.ethereum.org/2016/06/17/critical-update-re-dao-vulnerability (accessed on May 29, 2023).
[4] B. Bünz, J. Bootle, D. Boneh, A. Poelstra, P. Wuille, and G. Maxwell. Bulletproofs: Short proofs for confidential transactions and more. In 2018 IEEE Symposium on Security and Privacy (SP), pages 315–334, May 2018.
[5] Claude Crépeau. Commitment. In Henk C. A. van Tilborg and Sushil Jajodia, editors, Encyclopedia of Cryptography and Security, 2nd Ed, pages 224–227. Springer, 2011.
[6] Beam Foundation. Beam confidential cryptocurrency, 2020.
Available online: https://beam.mw/ (accessed on May 29, 2023).
[7] Grin. Introduction to MimbleWimble and Grin, 2016.
Available online: https://github.com/mimblewimble/grin/blob/master/doc/intro.md (accessed on May 29, 2023).
[8] F. Idelberger, G. Governatori, R. Riveret, and G. Sartor. Evaluation of logic-based smart contracts for blockchain systems. In J. Alferes, L. Bertossi, G. Governatori, P. Fodor, and D. Roman, editors, Rule Technologies. Research, Tools, and Applications - 10th International Symposium, RuleML 2016, Stony Brook, NY, USA, July 6-9, 2016. Proceedings, volume 9718 of LNCS, pages 167–183. Springer, 2016.
[9] T. Jedusor. Mimblewimble, 2016.
Available online: https://scalingbitcoin.org/papers/mimblewimble.txt (accessed on May 29, 2023).
[10] G. Maxwell. Coinjoin: Bitcoin privacy for the real world, 2013.
Available online: https://bitcointalk.org/index.php?topic=279249.0 (accessed on May 29, 2023).
[11] G. Maxwell. Confidential transactions write up, 2020.
Available online: https://web.archive.org/web/20200502151159/,
https://people.xiph.org/~greg/confidential_values.txt (accessed on May 29, 2023).
[12] Ian Miers, Christina Garman, Matthew Green, and Aviel D. Rubin. Zerocoin: Anonymous distributed e-cash from bitcoin. In 2013 IEEE Symposium on Security and Privacy, pages 397–411, 2013.
[13] A. Poelstra. Mimblewimble, 2016.
Available online: https://download.wpsoftware.net/bitcoin/wizardry/mimblewimble.pdf (accessed on May 29, 2023).
[14] Grigore Rosu. Formal Design, Implementation and Verification of Blockchain Languages Using K (Invited Talk). In Bruno Bernardo and Diego Marmsoler, editors, 2nd Workshop on Formal Methods for Blockchains (FMBC 2020), volume 84 of OpenAccess Series in Informatics (OASIcs), pages 1:1–1:1, Dagstuhl, Germany, 2020. Schloss Dagstuhl–Leibniz-Zentrum für Informatik.
[15] Tim Ruffing, Sri Aravinda Thyagarajan, Viktoria Ronge, and Dominique Schröder. Burning zerocoins for fun and for profit: A cryptographic denial-of-spending attack on the zerocoin protocol. Cryptology ePrint Archive, Paper 2018/612, 2018. https://eprint.iacr.org/2018/612.

[16] Adrián Silveira, Gustavo Betarte, Maximiliano Cristiá, and Carlos Luna. A formal analysis of the mimblewimble cryptocurrency protocol. Sensors, 21(17), 2021.

Nota contribución de los autores:

1. Concepción y diseño del estudio

2. Adquisición de datos

3. Análisis de datos

4. Discusión de los resultados

5. Redacción del manuscrito

6. Aprobación de la versión final del manuscrito

AS ha contribuido en: 1, 2, 3, 4, 5 y 6.

GB ha contribuido en: 1, 2, 3, 4, 5 y 6.

MC ha contribuido en: 1, 2, 3, 4, 5 y 6.

CL ha contribuido en: 1, 2, 3, 4, 5 y 6.

Nota de aceptación: Este artículo fue aprobado por los editores de la revista Dr. Rafael Sotelo y Mag. Ing. Fernando A. Hernández Gobertti.