Unchained Identities: Putting a Price on Sybil Nodes in Mobile Ad hoc Networks

Arne Bochem, Benjamin Leiding, and Dieter Hogrefe

University of Goettingen, Institute of Computer Science, Goettingen, Germany
{arne.bochem,benjamin.leiding,hogrefe}@cs.uni-goettingen.de

Abstract. As mobile ad hoc networks (MANETs) and similar decentralized, self-organizing networks grow in number and popularity, they become worthwhile targets for attackers. Sybil attacks are a widespread issue for such networks and can be leveraged to increase the impact of other attacks, allowing attackers to threaten the integrity of the whole network. Authentication or identity management systems that prevent users from setting up arbitrary numbers of nodes are often missing in MANETs. As a result, attackers are able to introduce nodes with a multitude of identities into the network, thereby controlling a substantial fraction of the system and undermining its functionality and security. Additionally, MANETs are often partitioned and lack Internet access. As a result, implementing conventional measures based on central authorities is difficult. This paper fills the gap by introducing a decentralized blockchain-based identity system called *Unchained*. *Unchained* binds identities of nodes to addresses on a blockchain and economically disincentivizes the production of spurious identities by raising the costs of placing large numbers of Sybil identities in a network. Care is taken to ensure that circumventing *Unchained* results in costs similar or higher than following the protocol. We describe an offline verification scheme, detail the functionalities of the concept, discuss upper- and lower-bounds of security guarantees and evaluate *Unchained* based on case-studies.

Keywords: MANET, Security, Sybil Attack, Blockchain, Identity, Authentication

1 Introduction

Stimulated by the persistent growth and expansion of the Internet of Things (IoT) [21,25], as well as progressing digitalization of our daily life, e.g., [15] and [29], wireless ad hoc networks such as mobile ad hoc networks (MANETs) or vehicular ad hoc networks (VANETs) become more common and popular. MANETs and their sub-types are often heavily partitioned, with transient connections occurring between nodes due to their mobility, resulting in a constantly changing network topology. Furthermore, communication in MANETs is usually organized in a decentralized manner without a connection to any central authority or the Internet [19,28]. As a result, these networks are worthwhile and

easy targets for attackers. This raises the issue of providing proper security and privacy protection mechanisms in order defend them against attacks. Without such, the distributed nature of MANETs and their lack of a central authentication authority leaves them easy targets for Sybil attacks. This type of attack is a common issue in large-scale peer-to-peer (P2P) systems, where hostile or faulty computing elements threaten the security of the whole network. Single faulty entities may be able to present multiple identities, thereby controlling a substantial fraction of the system, consequently undermining its functionality and security [12].

Several techniques focus on preventing Sybil nodes from joining a network at all [11,16]. Other approaches attempt to detect them when they are already part of the network [3,30]. One of the key enablers of Sybil attacks is the absence of a mechanism that prevents attackers from setting up arbitrary numbers of (virtual) nodes. In MANETs, there usually is no central authority that controls or administers the network. Since detecting Sybil nodes after joining a network is a cumbersome and inaccurate task, we propose the *Unchained* protocol which introduces economic disincentives of introducing Sybil nodes to a network by leveraging blockchain technology and combining it with an offline verification approach.

Unchained uses blockchain technology to bind ad hoc network node identities to blockchain-based wallet addresses, i.e. public/private key pairs, and requires a certain deposit to be made on the blockchain in order to join the network. Circumventing the protocol and introducing a Sybil node means investing even more financial assets than it would cost to create an *Unchained* identity the regular way. Due to its offline verification approach, *Unchained* operates in environments without internet access and without direct access to the underlying blockchain, thereby "unchaining" its security mechanism. This allows its use in MANETs with no or merely intermittent Internet connectivity.

The remainder of this paper is structured as follows: Section 2 introduces related works and supplementary literature. Section 3 focuses on the operational details of the *Unchained* approach. Afterwards, Section 4 details security properties of the protocol and explains how to customize the protocol for various use cases, while Section 5 elaborates on different options to handle difficulty changes in the underlying cryptocurrency. Section 6 provides a discussion and evaluation based on case studies. Finally, Section 7 concludes this work and provides an outlook on future work.

2 Supplementary Literature and Related Works

This section provides background information and describes related works regarding previous approaches to solve the issue of Sybil attacks. Section 2.1 provides general information on the concepts of blockchain technology, terms and frameworks. Section 2.2 focuses on related works.

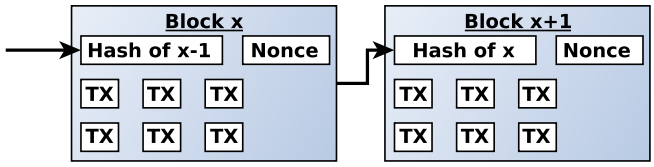


Fig. 1: Blockchain structure, adapted from [22].

2.1 Blockchain Technology

A blockchain consists of a (theoretically) unlimited number of blocks which are chained together in a chronological order. Each block consists of transactions that successfully passed a validation procedure. As illustrated in Figure 1, the collected valid transactions result in a new block that is added to the existing blockchain. The blockchain concept, also called distributed ledger system, is most notably known for providing the foundation of the peer-to-peer (P2P) cryptocurrency and payment system Bitcoin (₿) [22].

A key enabler of blockchains is the so called mining process, allowing to achieve a global consensus on which transactions to include in the next block in a decentralized way. Currently, the most common blockchain consensus algorithm is based on proof-of-work, which is used by Bitcoin, Ethereum [32], and others. When collecting transactions to form a new block, participants have to solve a computationally hard puzzle that is referred to as proof-of-work. A proof of work is a piece of data that is difficult to produce but easy to verify and satisfies certain requirements. Bitcoin's proof-of-work is based on searching for a nonce (value) that when hashed together with a block header, begins with a number of zero bits. "The average work required is exponential in the number of zero bits required and can be verified by executing a single hash" [22]. The varying number of zero bits is used to adjust the difficulty of finding a valid block. Hardware speed ups and growing user participation in building blocks result in more computing power being available for the mining process. In order to publish new blocks in, on avergae, a given time intervals, the difficulty of the proof-of-work is adjusted depending on the available computing power. In the case of Bitcoin, the target time per block is ten minutes. In the case that new blocks are generated too fast, the difficulty increases, if new blocks are generated too slowly, it is decreases.

As soon as a block with a valid nonce is found, the block is published and attached to the chain. All participants verify the submitted proof-of-work for correctness, the included transactions for validity and accept it as the new latest block. Since each block depends on its predecessor, changing the content of a block requires an infeasible recalculation of all successor blocks. The first user to find a new block also receives a block reward. In the case of Bitcoin, this reward is, as of December 2017, 12.5 ₿ plus additional transaction fees. These block rewards have both the purpose of disseminating the currency among users, as well as incentivizing miners to spend energy on securing the blockchain. If

multiple blocks are found at the same chain height, mining may proceed on either block and the longer chain is considered valid.

2.2 Related Works

Several other projects focus on Sybil attack prevention and Sybil attack detection in different network environments, therefore we only highlight some further publications. SybilGuard [35] is one of the well-known protocols that aims to limit the corruptive influence of Sybil attacks in peer-to-peer networks. The SybilLimit protocol is an advanced version of SybilGuard and aims to defend online social networks from Sybil nodes [34]. SybilGuard as well as SybilLimit rely on human-established trust relationships, hence they cannot be applied to mobile ad hoc networks.

[3] and [30] focus on Sybil attack detection in MANETs, whereas [33] targets the detection and localization of Sybil nodes in VANETs. In contrast to these approaches, Unchained focuses on preventing sybil nodes from joining a network instead of detecting them when they are already part of the network.

Furthermore, [11] and [23] try to prevent and detect Sybil attacks in sensor networks, whereas Unchained focuses on mobile ad hoc networks in general.

Blockchains matured and grew in popularity, resulting in various blockchain architectures, e.g., Ethereum [32], Qtum [10], or IOTA [27], as well blockchain-based applications and use cases, e.g., as a platform for IoT applications [9,26], applications in the automotive sector [18], in the finance sector [8,24] or as a part of security and authentication protocols [17,20,26].

3 Unchained Identities in MANETs

When setting up a new network, e.g., a MANET, each node is equipped with an identity that uniquely identifies the specific device within the network. When deployed, communicating devices have to validate each others identities for security and privacy reasons before exchanging information. The following section describes the general process of creating new identities when flashing the firmware to a device as well as validating identities. Both of these processes are based on blockchain technology and do not require any trusted third parties apart from a decentralized cryptocurrency's P2P network. For illustration purposes, we use the Bitcoin network in the following sections. However, *Unchained* can be implemented on all proof-of-work based Blockchains.

3.1 Creating a New Identity

The process of creating a new unchained identity is illustrated in Figure 2 and assumes that a key pair, i.e. public and private key, presenting a Bitcoin wallet address already exists and that it holds a certain amount of Bitcoin. The amount contained within this address needs to be sufficient to make the deposit necessary in the first step of the identity creation process. First, the coins in the Bitcoin

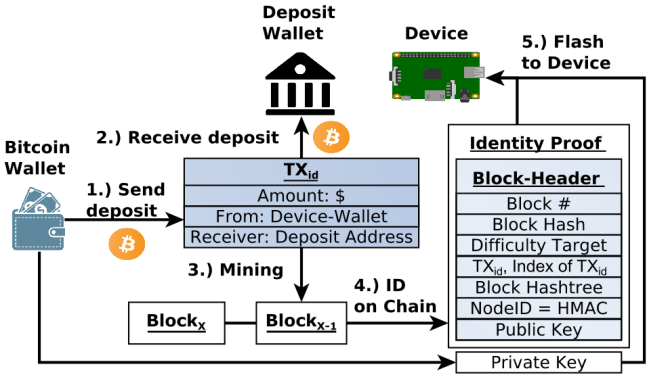


Fig. 2: Creating a new identity.

wallet are transferred to a pre-defined deposit address (step 2) and the resulting transaction is mined into a block by the Bitcoin network as part of Block$_X$ (step 3). In the subsequent step 4, an *Identity proof* is created based on the information from the minded block. The proof contains the block header (block number, block hash, difficulty target of the block), the deposit transaction, hashes for the merkle tree allowing to prove that the transaction is part of the block, the index number of the deposit address in the block as well as the public key.

Furthermore, a unique *NodeID* is calculated based on the Hash Message Authentication Code (HMAC) as illustrated in Equation 1 - 3. First, the block's proof-of-work hash is used as a key for the HMAC calculation in combination with the index number of the deposit transaction in the block. The purpose of this approach is to prevent attackers from attempting to create node IDs matching arbitrary attacker defined criteria. Since the ID depends on the deposit transaction, but also on the block's proof-of-work hash, guessing the node ID is equivalent to predicting the correct hash of the next block of the Bitcoin blockchain and therefore not feasible.

$$k_{\text{HMAC}} := \text{Block}_{\text{PoWHash}} \tag{1}$$

$$\text{TXindex} := \text{index of deposit TX in Block} \tag{2}$$

$$\text{nodeID} := \text{HMAC}(k_{\text{HMAC}}, \text{TXindex}) \tag{3}$$

Finally, the constructed *identity proof* and the node's private key are flashed onto the node, which is afterwards deployed in the network.

3.2 Identity Validation

Communication between nodes of a network is an essential functionality of ad hoc network. Before transmitting application data, nodes verify each others' identity in a bidirectional manner in order to secure and protect sensible network data.

The validation of node identities is performed upon first contact of each two nodes, such as a two-way handshake or the broadcasting of identity information

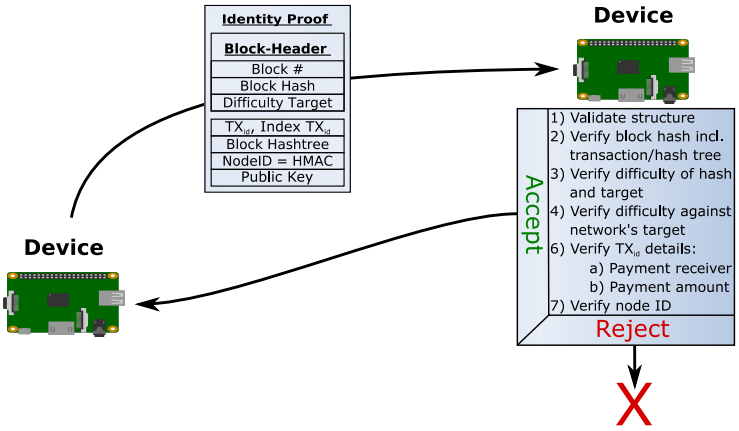


Fig. 3: Overview of the validation process.

to let nodes learn about their neighbors. In the following, we describe how participating nodes verify an identity upon receipt of its *identity proof*. A graphical overview of this procedure is given in Figure 3. In the case of a two-way handshake, the procedure is simply repeated on each side after receiving the *identity proof*. Another potential scenario is a number of nodes broadcasting their *identity proofs* and verifying received *identity proofs*, allowing them to connect to surrounding peers when necessary.

First, the structure of the *identity proof* is validated. A valid *identity proof* contains a block header, a deposit transaction, the hashes of a merkle tree proving that the transaction is part of the specific block, a node ID and the public key of the node, which was also used to sign the deposit transaction. At this point, it is also verified that the block's height is above $\text{networkParameter}_{\text{height}}$ Afterwards, the block corresponding to the block header is verified based on the hashes from the merkle tree that is used to verify the transaction's presence in the block. Next, the node checks that the difficulty of the block hash matches the difficulty target in the block header and that the difficulty target is at least $\text{networkParameter}_{\text{minDifficulty}}$ for the given block height.

Once these properties are confirmed, the deposit transactions is verified. It contains a payment greater or equal to $\text{networkParameter}_{\text{amount}}$, which is sent to the mandatory receiver address $\text{networkParameter}_{\text{receiver}}$. The transaction has to be correctly signed with exactly one key pair that is also used for any future cryptographically secure (encryption, signatures, key exchange) communications with other nodes and the public key that is included in the *identity proof*. Finally, it is verified that the node ID matches the formula given in Equation 3.

If all steps described above are successful, the validation process finsihes and the participating nodes may start communicating. Otherwise the received *identity proof* is discarded and no further communication is initiated.

4 Parameter Choice

In Section 4.1 we discuss various network parameters that allow adjusting the behavior of *Unchained* to suit different use cases. In addition, security properties and considerations are detailed. Section 4.2 focuses on pricing an identity.

4.1 Network Parameters

First, we describe the network parameters that allow the customization of *Unchained* to suit different use cases.

Starting Block Height The parameter $\text{networkParameter}_{\text{height}}$ defines the minimum block height that is accepted for *identity proofs*. The block height is defined by the number of blocks preceding a block on the blockchain. The genesis block's block height is zero [4]. The block height corresponds to the block height at the start of the network's lifetime. By rejecting *identity proofs* at lower block heights, there is no need to consider allowing blocks with significantly lower difficulties.

Deposit Address $\text{networkParameter}_{\text{receiver}}$ is the deposit address to which a transaction, used when creating an *identity proof*, sends a certain amount of Bitcoin. The main property of this address is that funds sent there should not be recoverable by an attacker that is trying to create a large number of identities. Hence, using transaction fees instead of a deposit is not a viable option since an attacker may mine a valid block on their own and directly recover all funds. Currently, *Unchained* provides three different options that define what happens to the deposit.

The first option is proof-of-burn [7], where an invalid receiving address with no (known) existing private key is used. As a result, the sent deposit cannot be recovered. This method is secure, but not elegant since it destroys a certain amount of Bitcoin and the Bitcoin supply is strictly limited by the underlying Bitcoin protocol.

The second option is that the software of the network secured with *Unchained* is developed by a certain entity, or the network is maintained or controlled by a certain entity. The entity may choose to use an address under its control as the receiving address. This way, the developers or maintainers of the network could raise funds for further development by receiving Bitcoin through the creation of identities used by users of the network. As the developers have an interest in keeping the network secure, this approach is a viable choice that prevents attackers from recovering funds.

A third approach is to use the donation address of a charity. Unless the charity itself has an interest in attacking the network or is otherwise compromised, an attacker is unlikely to be able to recover the funds. If desired by a network operator, they may choose to allow multiple deposit addresses to be used. Hence, users may choose between different charities when making a donation to create an *Unchained* identity.

Deposit Amount The $\text{networkParameter}_{\text{amount}}$ parameter determines the minimum deposit size required to set up a new identity. The amount is chosen in such a way that it is affordable for those who would like to participate in the network, while still being high enough to disincentivize the creation of large numbers of Sybil nodes. For larger networks, the deposit size may be lower since the network may tolerate higher numbers of spurious identities before an attacker gains a tangible benefit from their use. Section 4.2 details further considerations, limits and implications that depend on the deposit size.

A potential alternative is to use a small value for $\text{networkParameter}_{\text{amount}}$ and introducing a bigger $\text{networkParameter}_{\text{lockedAmount}}$ value. The first amounts gets sent to $\text{networkParameter}_{\text{receiver}}$, while the second amount is sent back to the identity's owner, but locked up using the `CheckLockTimeVerify` output [31] of a transaction or another type of smart contract. The locktime is equal to the lifetime of the identity. This way users may recover their funds after leaving the network, while ensuring that the creation of high numbers of concurrent identities still lock up significant amounts of capital.

Minimum Difficulty This parameter defines the minimum amount of work that is required to generate a new block that may be used to build an *identity proof*. While Section 5 details more sophisticated approaches to control the allowable difficulty of blocks for *identity proofs*, the most basic way is to set a simple minimum difficulty parameter $\text{networkParameter}_{\text{minDifficulty}}$ that matches the underlying blockchain's difficulty at the time of setting up the network using *Unchained*. Alternatively, a value slightly below this value may be chosen to allow for drops in network difficulty.

If the parameter is hardcoded, it should be selected sufficiently low. If Bitcoin's target difficulty drops below the hardcoded value, it becomes impossible to create further identities. To avoid this issue, implementations should set $\text{networkParameter}_{\text{minDifficulty}}$ dynamically as described in Section 5.

Updates Changes in the valuation of Bitcoin, difficulty or simply the general operating environment of the secured network may change over time. Therefore, it may become necessary to update the parameters described above to ensure that the network is operating as desired.

Assuming an entity that is maintaining and developing the network, it is possible to periodically distributed signed bundles of updated network parameters, including the block height at which this bundle should take effect. After the bundle is published, users who generate fresh identities should attach the update to their *identity proof* before flashing it onto their node. Similar considerations are also described in Section 5 with a special focus on difficulty updates.

4.2 Pricing an Identity

An attacker trying to create a large number of identities aims to minimize costs. One option is to mine a block conforming to the $\text{networkParameter}_{\text{minDifficulty}}$

parameter, filling it solely with deposit transactions, but never publishing it to the Bitcoin network. Since *Unchained* does not verify the full blockchain, these transactions do not even require valid inputs. However, mining a block with valid difficulty and not publishing it incurs a high opportunity cost, as well as energy cost. Hence, instead of paying for the identities, the attacker pays for the hashing power used to create the block. While the energy costs may vary, depending on location, the opportunity cost is easy to quantify and equal to the block reward plus additional transaction fees.

Given the current block size (1 MB), minimum transaction size (224 B) and block reward (12.5 ₿ plus fees) of Bitcoin, this also leads to an upper limit on the price for one identity, as given in Equation 4, with the current maximum amount given in Equation 5 [1,2,7].

$$\text{amount}_{\max} = \text{block reward} \cdot \frac{\text{min TX size}}{\text{max block size}} \tag{4}$$

$$= 12.5\,\text{₿} \cdot \frac{224\,\text{B}}{1\,\text{MB}} = 2.8\,\text{m₿} \tag{5}$$

Given the current price of Bitcoin as of 2017-10-29 at approximately $10 399 [5,6], the resulting maximum price per identity is roughly equivalent to $29. Going above this limit makes it cheaper for an attacker to generate a fake block than simply paying for the identities, as long as fees and energy costs are disregarded. Most networks will likely set a lower value than 2.8 m₿ for $\text{networkParameter}_{\text{amount}}$, in order to make identities more affordable for users and anticipate volatility with regards to Bitcoin valuation.

5 Handling Difficulty Adjustments

Our system has to adapt to changes in the target difficulty of the underlying cryptocurrency. In the case of Bitcoin, the difficulty is adjusted every 2016 blocks. This is equivalent to roughly two weeks. These adjustments are made to keep the time interval between each block at, in the case of Bitcoin, on average 10 minutes. To handle these adjustments, we propose multiple approaches with different trade-offs.

Each node keeps a list of accepted target difficulties for each 2016 block interval. If the accepted target difficulty of an interval is adjusted upwards due to new information, identities confirmed using blocks with lower difficulty are retroactively invalidated. When the accepted target difficulty is lowered, it may be prudent to retroactively accept discarded peers into the network. However, since invalid identities are unlikely to be stored, the second case is unlikely to be implemented. The different approaches of handling difficulty changes concern the way this list of accepted target difficulties is updated.

5.1 Maximum Seen Difficulty

The first approach is both simple to implement as well as fully decentralized. The list of accepted target difficulties is initialized to zero or a known history

at the point the node is initialized. Whenever an *identity proof* is received by a node, it looks up the target difficulty for that block in the list of difficulties. If both difficulty values match, the identity is accepted. In case the difficulty of the received *identity proof* is lower, the identity is discarded. Alternatively, when the difficulty of the received identity is higher, it is accepted and the target difficulty in the list is updated. If the list of accepted target difficulties was initialized with a known history however, these known-good values should not be overwritten even if an identity with a higher difficulty is encountered.

This solution allows the eventual detection and invalidation of forged identities that were validated using blocks of insufficiently high difficulty, as long as a connection to an honest node from the same two week period is made at some point. No infrastructure in addition to the previously described system is necessary.

As a caveat, this method is vulnerable to a denial of service attack. Assuming an attacker is able to mine a block targeting a difficulty that is higher than the difficulty of the underlying cryptocurrency and uses the block in an *identity proof*, the targeted nodes will update their lists of target difficulties accordingly, invalidating all regular identities that were generated during the timeframe corresponding to the malicious block. However, mining a block targeting a higher difficulty is even more expensive than mining a regular block. This issue can be mitigated by combining this approach with one of the two following methods. At the same time, this method can be used as a fallback solution for both of them.

5.2 Bundled Updates

Assuming network is run by a single operator, the operator may publish signed messages containing the target difficulty for each 2016 block range. The message is appended to each *identity proof*, setting the target difficulty in the list to the provided value. A drawback of this solution is that, in case the operator ceases to exist, no further difficulty updates can be broadcasted, leaving new nodes unable to join the network. However, when combining this approach with the mechanism from Section 5.1, only nodes worried about denial of service attacks need to attach update messages to their *identity proofs*. Nevertheless, joining the network without one of these messages also remains as an alternative. Nonetheless, when the operator ceases operations, the mitigation for the denial of service attack vector also ceases to be functional.

5.3 Majority Vote

Rather than relying on a single operator as in Section 5.2, nodes may choose to accept signed difficulty updates from multiple providers. One or more of these messages may then be attached to an *identity proof*. The values of each update provider are stored in the list of target difficulties. In the event that for any interval mismatching difficulty update messages are detected, the majority value is considered the true difficulty target. Whenever there is no majority, the highest

value is treated as the true difficulty target. In the case of a majority, nodes might mistrust future update messages provided by providers belonging to the minority.

This approach has multiple benefits over the previous approach. There is no single point of failure that prevents the network from growing. Additionally, supposing an attacker is able to trick a difficulty update provider to forge an update, the result is not necessarily a successful attack, as the attacker is still missing a majority that accepts the update, Hence, the attacker has to compromise at least 50 % of the update providers to perform a denial of service attack.

Moreover, this approach is compatible with the solution from Section 5.1, allowing nodes to join the network even without access to any signed difficulty update messages.

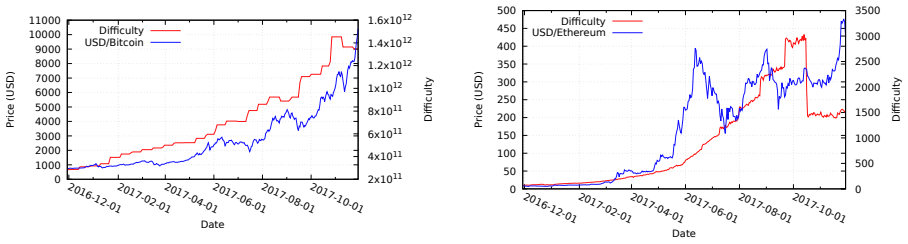
6 Evaluation

The following section focuses on evaluating the *Unchained* protocol and the provided security guarantees. Since *Unchained's* security guarantees mainly depend on the difficulty level as well as the token price of the underlying proof-of-work blockchain, we analyze how changing difficulty levels and token prices would have affected the Sybil attack prevention mechanism of a fictional MANET deployed in December 2016. Section 6.1 and Section 6.2 perform analysis based on the assumption that the Bitcoin blockchain is used, and Section 6.3 uses the same scenario based on the Ethereum blockchain. We choose these two chains for several reasons: First, they are the most popular and most utilized proof-of-work blockchains that currently exist. Second, the different changes in difficulty and price cover important corner cases such as increasing and decreasing difficulties over time with sudden drops and raises. We assume a scenario of a MANET that was initially deployed in December 2016 with nodes continually joining and leaving the network and operated until the time of writing this work in November 2017.

6.1 Bitcoin Difficulty Analysis

The difficulty level of the Bitcoin blockchain is adjusted every 2016 blocks, which is equivalent to 14 days given a blocktime of ten minutes per block. As illustrated in Figure 4a, the difficulty level is steadily rising with two minor exception in August and November 2017. At the same time, as shown in Figure 4a, the Bitcoin price itself also increased almost steadily by a factor of ten within the last twelve months.

As already discussed in Section 4 the lower bound of security guarantees provided by *Unchained* is always the lowest level of difficulty and the lowest price per block that occurred during the existence of the network. Given the initial deployment of our hypothetical MANET in December 2016, all nodes joining at later stages have higher security guarantees than the initial nodes due to an increased block difficulty and price. As discussed in Section 4.2, the



(a) Bitcoin (b) Ethereum

Fig. 4: Average daily price in USD and block difficulty level between Dec. 2016 and Nov. 2017 (Source: [5,6,14]).

price is only a theoretical measurement for security guarantees, since it is up to the network operator to decide the minimum price of a node's identity. However, increasing token prices and therefore also increasing block prices, may also result in more expensive identities thereby raising the bar for a Sybil attacks.

Assuming a MANET setup at the beginning of November 2017, days before the decreasing Bitcoin price in (see Figure 4a), reflects the exact opposite where it becomes less expensive to introduce new identities to the system for a short period of time. However, as discussed previously, it is up to the network operator to decided whether to pick the maximum possible identity-per-block-price or a lower price. For practical reasons it is likely that most operators pick a lower price and therefore minor price declines do not affect the security guarantees of our example MANET a lot. Moreover, it is also up to the network operator to define a minimum identity-price that is higher than the identity-per-block-price since it is still unlikely that a malicious entity has the computational power to mine a block with a matching difficulty level, given the vast hashing power of Bitcoin's mining pools.

6.2 Bitcoin Price Analysis

Using the historic price data gathered for Figure 4a, going back until July 17th 2010, it is possible to calculate for each date, on which a network using *Unchained* could have been started, the highest drop in price and thus security level experienced by the network. While future developments cannot reliably predicted, this provides an intuition on the historic worst case performance of *Unchained*. In Table 1, the proportion of starting dates that would have lead to a drop on any subsequent day of at least a given percentage is given. Only for 0.1 % of possible starting dates the security level would have at any later point dropped below 10 % of the given date.

Historically, high drops in security level only occur very rarely. Smaller drops occur more frequently, with almost 50 % of possible starting days experiencing

Drop to	Affected started dates
$< 10\,\%$	$0.1\,\%$
$< 20\,\%$	$2.8\,\%$
$< 30\,\%$	$8.5\,\%$
$< 40\,\%$	$13.5\,\%$
$< 50\,\%$	$18.0\,\%$
$< 60\,\%$	$22.8\,\%$
$< 70\,\%$	$27.2\,\%$
$< 80\,\%$	$36.2\,\%$
$< 90\,\%$	$49.5\,\%$
$< 100\,\%$	$77.5\,\%$

Table 1: Affected starting dates after which the Bitcoin price drops below a certain percentage of the given day's price.

drops of at least 10 % at some point in the future. While most networks will be able to tolerate smaller drops in security level, raising Bitcoin prices can also be an issue, as they can make identities too expensive for regular users. Considering this, for networks intended to exist over long time frames, provisions for an update mechanism for $\text{networkParameter}_{\text{amount}}$ should be made. In case mass adoption occurs, the volatility level of cryptocurrencies and fiat currency is expected to converge. Hence, Unchained's level of security will stabilize as well.

6.3 Ethereum Difficulty Analysis

Unchained is blockchain agnostic as long as the underlying chain architecture uses a proof-of-work consensus algorithm. Therefore, we also analyze how the changing difficulty levels and token prices of the Ethereum blockchain and how this would have affected the Sybil attack prevention mechanism of our fictional MANET deployed in December 2016.

Similar to the Bitcoin price, the Ethereum price also increased heavily, starting around $8 in December 2016 to more than $450 at the end of November 2017, even though the Ethereum price suffered some decreases in July 2017. Ethereum's block difficulty, illustrated in Figure 4b, also increased over the last twelve month. However, a sudden drop occurred on October 16 due to a difficulty adjusting hard-fork of the Ethereum network [13]. Nevertheless, even the reduced difficulty level is far higher than the initial level in December 2016.

As a result, the security guarantee evaluation results are similar to the Bitcoin evaluation of Section 6.1. MANET nodes setup in December 2016 with the initial difficulty are cheaper and easier to create in terms of identity price and block difficulty. All nodes created afterwards provided higher security guarantees. When focusing on the timeframe briefly before and after the difficulty adjustment, identities created before the adjustment are less difficult than identities created afterwards. The same applies for the price of identities both before and after the price drop of Ether in July 2016 as illustrated in Figure 4b.

In both the case of the Bitcoin as well as the Ethereum blockchain, price and difficulty increased heavily within the last 12 month. As a result, the lower bound of provided security is defined by the earliest nodes that joined the test MANET when created, since their *identity proofs* depends on the lowest block difficulty and identity price. All following node identities provide security guarantees above this lower bound. Furthermore, given the case that the difficulty levels will likely not increase indefinitely and remain somehow static (with minor fluctuations) at some point in the future, *Unchained's* lower and upper bounds will also converge and be less volatile.

7 Conclusion and Future Work

Detecting Sybil node attacks is major issue of large-scale P2P networks where malicious nodes threaten the security of the overall system. After joining a network, detecting such nodes is a cumbersome and inaccurate task. In this work we introduce a protocol for a decentralized blockchain-based identity system with offline verification that raises the difficulty of introducing high numbers of Sybil nodes to a network by providing economic disincentives.

Unchained uses blockchain technology to bind ad hoc network node identities to blockchain-based wallet addresses, i.e., public/private key pairs. In order to join the network, a proof-of-identity is created for each device. The proof is derived from a deposit transaction made from the wallet address to a deposit address and flashed to the node afterwards. Nodes validate each others' identities using the uniquely generated *identity proofs*.

Circumventing the protocol and introducing a Sybil node is equivalent to investing more financial assets than it would cost to create a malicious block on the blockchain. In addition, *identity proofs* are designed in such a way that no Internet access or direct connection to the underlying blockchain is required after the initial setup of the ad hoc network, thereby raising the bar to introduce Sybil nodes to even highly partitioned networks.

We detail the network parameters and update mechanism of *Unchained* and discuss upper- and lower-bounds of security guarantees. Finally, an evaluation based on a hypothetical MANET deployed leveraging the Bitcoin and Ethereum blockchain is used to analyze the protocol's security properties depending on the block difficulty and token prices between December 2016 and November 2017.

For future work, we plan to generalize the protocol and not only focus on MANETs or other ad hoc networks and instead integrate *Unchained* into IoT environments. Furthermore, we intend to explore the feasibility of adapting existing Sybil attack prevention or detection algorithms to consider the node's *identity proof* block difficulty and block price as attributes for their trust scoring systems.

We also aim to implement and deploy the *Unchained* protocol on the Bitcoin as well as Ethereum blockchain and evaluate real-world use-cases.

References

1. Mining - Bitcoin Wiki https://en.bitcoin.it/w/index.php?title=Mining&oldid=64115#Reward, (Accessed December 11, 2017)
2. Transaction - Bitcoin Wiki https://en.bitcoin.it/w/index.php?title=Transaction&oldid=63712, (Accessed December 11, 2017)
3. Abbas, S., Merabti, M., Llewellyn-Jones, D., Kifayat, K.: Lightweight Sybil Attack Detection in MANETs. IEEE systems journal 7(2), 236–248 (2013)
4. Bitcoin Project: Bitcoin Developer Guide. https://bitcoin.org/en/developer-guide#proof-of-work (2017), (Accessed December 18, 2017)
5. Bitcoincharts: Bitcoincharts API, Price data (MtGox, BTC-e, BitStamp, Coinbase), https://api.bitcoincharts.com/v1/csv/, (Accessed November 29, 2017)
6. Blockchain.info: Bitcoin Blockchain, Difficulty, https://api.blockchain.info/charts/difficulty?format=csv, (Accessed November 29, 2017)
7. Bonneau, J., Miller, A., Clark, J., Narayanan, A., Kroll, J.A., Felten, E.W.: SoK: Research Perspectives and Challenges for Bitcoin and Cryptocurrencies. In: 2015 IEEE Symposium on Security and Privacy. pp. 104–121 (May 2015)
8. Bussmann, O.: The Future of Finance: FinTech, Tech Disruption, and Orchestrating Innovation. In: Equity Markets in Transition, pp. 473–486. Springer (2017)
9. Christidis, K., Devetsikiotis, M.: Blockchains and Smart Contracts for the Internet of Things. IEEE Access 4, 2292–2303 (2016)
10. Dai, P., Mahi, N., Earls, J., Norta, A.: Smart-Contract Value-Transfer Protocols on a Distributed Mobile Application Platform. https://qtum.org/uploads/files/a2772efe4dc8ed1100319c6480195fb1.pdf (2017), (Accessed November 22, 2017)
11. Dhamodharan, U.S.R.K., Vayanaperumal, R.: Detecting and Preventing Sybil Attacks in Wireless Sensor Networks Using Message Authentication and Passing Method. The Scientific World Journal 2015 (2015)
12. Douceur, J.R.: The Sybil Attack. In: International Workshop on Peer-to-Peer Systems. pp. 251–260. Springer (2002)
13. Ethereum Team: Byzantium HF Announcement. https://blog.ethereum.org/2017/10/12/byzantium-hf-announcement/ (2017), (Accessed November 30, 2017)
14. Etherscan: Ethereum Charts and Statistics. https://etherscan.io/charts (2017), (Accessed November 30, 2017)
15. Horst, H.A., Miller, D.: Digital Anthropology. A&C Black (2013)
16. John, R., Cherian, J.P., Kizhakkethottam, J.J.: A Survey of Techniques to Prevent Sybil Attacks. In: Soft-Computing and Networks Security (ICSNS), 2015 International Conference on. pp. 1–6. IEEE (2015)
17. Leiding, B., Cap, C.H., Mundt, T., Rashidibajgan, S.: Authcoin: Validation and Authentication in Decentralized Networks. In: The 10th Mediterranean Conference on Information Systems - MCIS 2016. Cyprus, CY (September 2016)
18. Leiding, B., Memarmoshrefi, P., Hogrefe, D.: Self-Managed and Blockchain-Based Vehicular Ad-Hoc Networks. In: Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing: Adjunct. pp. 137–140. ACM (2016)
19. Macker, J.: Mobile Ad-Hoc Networking (MANET): Routing Protocol Performance Issues and Evaluation Considerations, RFC 2501 (1999)
20. McCorry, P., Shahandashti, S.F., Clarke, D., Hao, F.: Authenticated key exchange over bitcoin. In: International Conference on Research in Security Standardisation. pp. 3–20. Springer (2015)

21. van der Meulen, R.: Gartner Says 8.4 Billion Connected "Things" Will Be in Use in 2017, Up 31 Percent From 2016. https://www.gartner.com/newsroom/id/3598917 (2017), (Accessed November 01, 2017)
22. Nakamoto, S.: Bitcoin: A Peer-to-Peer Electronic Cash System. https://bitcoin.org/bitcoin.pdf (2008), (Accessed January 26, 2017)
23. Newsome, J., Shi, E., Song, D., Perrig, A.: The Sybil Attack in Sensor Networks: Analysis & Defenses. In: Proceedings of the 3rd international symposium on Information processing in sensor networks. pp. 259–268. ACM (2004)
24. Nguyen, Q.K.: Blockchain - A Financial Technology for Future Sustainable Development. In: Green Technology and Sustainable Development (GTSD), International Conference on. pp. 51–54. IEEE (2016)
25. Nordrum, A.: Popular Internet of Things Forecast of 50 Billion Devices by 2020 Is Outdated. https://spectrum.ieee.org/tech-talk/telecom/internet/popular-internet-of-things-forecast-of-50-billion-devices-by-2020-is-outdated (2016), (Accessed November 01, 2017)
26. Ouaddah, A., Elkalam, A.A., Ouahman, A.A.: Towards a Novel Privacy-Preserving Access Control Model Based on Blockchain Technology in IoT. In: Europe and MENA Cooperation Advances in Information and Communication Technologies, pp. 523–533. Springer (2017)
27. Popov, S.: The Tangle - Version 1.3. https://iota.org/IOTA_Whitepaper.pdf (2017), (Accessed November 22, 2017)
28. Raza, N., Aftab, M.U., Akbar, M.Q., Ashraf, O., Irfan, M.: Mobile Ad-Hoc Networks Applications and Its Challenges (2016)
29. Su, K., Li, J., Fu, H.: Smart City and the Applications. In: Electronics, Communications and Control (ICECC), 2011 International Conference on. pp. 1028–1031. IEEE (2011)
30. Tangpong, A., Kesidis, G., Hsu, H.y., Hurson, A.: Robust Sybil Detection for MANETs. In: Computer Communications and Networks, 2009. ICCCN 2009. Proceedings of 18th Internatonal Conference on. pp. 1–6. IEEE (2009)
31. Todd, P.: BIP 65 - OP_CHECKLOCKTIMEVERIFY (2014), https://github.com/bitcoin/bips/blob/6295c1a095a1fa33f38d334227fa4222d8e0a523/bip-0009.mediawiki, (Accessed December 11, 2017)
32. Wood, G.: Ethereum: A Secure Decrentralized Generalised Transaction Ledger. http://gavwood.com/paper.pdf (2014), (Accessed November 22, 2017)
33. Xiao, B., Yu, B., Gao, C.: Detection and Localization of Sybil Nodes in VANETs. In: Proceedings of the 2006 workshop on Dependability issues in wireless ad hoc networks and sensor networks. pp. 1–8. ACM (2006)
34. Yu, H., Gibbons, P.B., Kaminsky, M., Xiao, F.: Sybillimit: A Near-Optimal Social Network Defense Against Sybil Attacks. In: Security and Privacy, 2008. SP 2008. IEEE Symposium on. pp. 3–17. IEEE (2008)
35. Yu, H., Kaminsky, M., Gibbons, P.B., Flaxman, A.: Sybilguard: Defending Against Sybil Attacks Via Social Networks. In: ACM SIGCOMM Computer Communication Review. vol. 36, pp. 267–278. ACM (2006)