IOTA: A Cryptographic Perspective

Bryan Baek
Harvard University
baek@alumni.harvard.edu

Jason Lin
Georgia Institute of Technology
jlin401@gatech.edu

1 Introduction

IOTA is an open-source distributed ledger protocol that presents a unique juxtaposition to the blockchain-based Bitcoin. Instead of using the traditional blockchain approach, IOTA stores individual transactions in a directed-acyclic graph (DAG) called Tangle. IOTA claims that this allows several benefits over its blockchain-based predecessors such as quantum-proof properties, no transaction fees, fast transactions, secure data transfer, and infinite scalability. Currently standing at 16th place in the CoinMarketCap, it topped 7 during the height of the bullish period of late 2017. Given these claims and attractions, this paper seeks to analyze the benefits and how they create vulnerabilities and/or disadvantages compared to the standard blockchain approach. Based upon our findings, we will discuss the future of IOTA and its potential of being a viable competitor to blockchain technology for transactions.

2 Background

IOTA, which stands for Internet of Things Application [1], is a crypto technology in the form of a distributed ledger that facilitates transactions between Internet of Things (IoT) devices. IoT encompass smart home devices, cameras, sensors, and other devices that monitor conditions in commercial, industrial, and domestic settings [2]. As IoT becomes integrated with different facets of our life, their computational power and storage is often left to waste when idle. David Sonstebo, Sergey Ivancheglo, Dominik Schiener, and Serguei Popov founded the IOTA Foundation in June 2015 after working in the IoT industry because they saw the need to create an ecosystem where IoT devices share and allocate resources efficiently [3]. Embedded hardware with limited computational power are often the hardware of choice for IoT and are not suitable for energy-hungry mining mechanisms that consume increasing resources over time. A reason to use IOTA over Bitcoin is the elimination of transaction fees to facilitate micropayments that are costly to scale. As a currency, blockchain also has the problem of slow transaction speeds which impedes its adoption in everday uses [4]. Currently, the IOTA Foundation develops and supports IOTA, which seeks to be the platform of choice for machine-to-machine (M2M) transactions to allocate IoT resources [3].

Unlike blockchain that uses mining as a Proof-of-Work (PoW) measure in determining who gets to propose the next block, IOTA eliminates block mining and fees altogether [3]. It requires every new entry to the DAG to contribute roughly the same amount of hash power as a prevention for spam transactions [3]. The Tangle effectively removes the centralization of hash power [1], an undesired feature associated with increasingly concentrated mining pools. All of the coins on IOTA (2,779,530,283,277,761 IOTA) were created at the genesis of the network [3], and the absence of fees is critical to handle large-scale M2M transactions. In a typical scenario of IoT resource allocation, the network needs to handle high frequency micropayments to achieve real-time functionalities.

Touted as an evolutionary successor to blockchain, IOTA's major difference is its usage of the Tangle instead of a linked list. The Tangle builds consensus by requiring the device to verify two other randomly chosen transactions on the network in order to submit a new transaction [5]. Unapproved transactions, called tips, are selected via Markov Chain Monte Carlo sampling using a weighted random walk on an equilibrium distribution to discourage repeated approval of old transactions [6]. Confirmation is measured by the number of transactions directly or indirectly connected to the current node, where each validation increases the likelihood of a transaction being genuine. The creators of the IOTA network would set a threshold c that represents the number of confirmations needed for a transaction to be confirmed by the recipient address, where transactions would need $n \geq c$ confirmations to be final [7]. However, this does not prevent an adversary from artificially inflating the number of tips by issuing many transactions that approve a fixed pair of transactions [6].

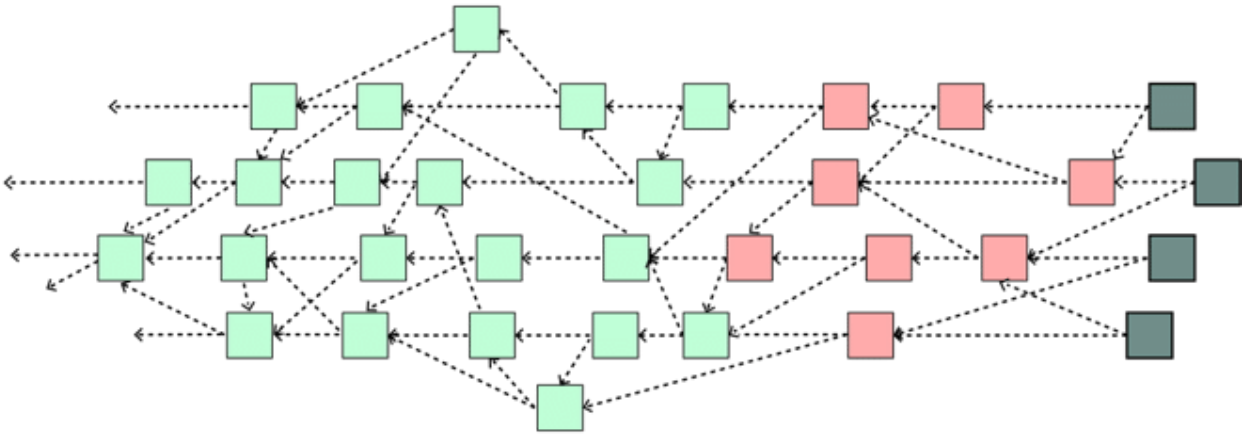

Figure 1: Graphic Representation of the IOTA Tangle

Another claimed benefit of IOTA is that it uses Winternitz one-time signature (WOTS) to be quantum secure. Quantum computers are projected to undermine public-key cryptography in the near future and cryptographic signatures that rely on traditional schemes face risks of quantum attacks by Shor's algorithm. Shor [8] proved that quantum computers utilizing subatomic qubits that exist in 1, 0 or superposition of both states to achieve superior optimization performance than their sequential counterparts [9], can be used to compute integer factorization exponentially faster than classical computers in polynomial time. This is relevant in blockchain, which uses the Elliptic Curve cryptography scheme that depends on mathematical properties such as the inability to efficiently factor out large primes and compute discrete logarithms for security. IOTA on the other hand, uses the WOTS scheme; it is proven [10] to be post-quantum secure by its one-way hash functions that assumes security from its underlying pseudo randomness.

The concept of using WOTS in IOTA is promising if properly conducted. Rompel showed that a secure signature scheme exists if and only if a secure hash-based signature scheme exists [11], and by reduction, the security of WOTS scheme is decoupled from the strong requirement of collision resistance [10] when instantiated with pseudorandom functions. Whereas Elliptic Curve digital signatures (ECDSA) are not quantum-proof because it establishes security on infeasibility to find the discrete logarithm of a random elliptic curve element, the effectiveness of hash-based signatures can at most be undermined by Grover's algorithm [12], a quantum algorithm that finds with high probability the unique input to a black box function that produces a particular output value. Grover's algorithm poses a much less severe attack than Shor's algorithm on public-key cryptography, and it can be defended against by determining an effective security lower-bound and increasing the capacity and output size of hash functions [12].

WOTS addresses many practical issues with just using One-Time Signatures (OTS). OTS can only be used once as it reveals part of the private key when provided with public key in its process for verification. WOTS reduces the signature size by introducing 1) repeated-hashing space-time tradeoffs and 2) a Merkle tree to sign multiple bits at once. To prevent signature forgery, multiple one-time private-public key pairs are generated to sign multiple bits at once into a Merkle tree whose root can be efficiently distributed. The prover would then only need to provide a signature, public key p_k and $O(n)$ proof of p_k membership to a verifier, resulting in significant cost-savings compared to linear growth in size of OTS signatures. Because each signature requires a new key pair, there is actually an implicit dependency on third party key generators, which can ironically lead to increased vulnerability to an attack. For example, a centralized key generator trusted by many IOTA users for their digital wallet was compromised last year and resulted in $11m funds stolen [13].

As a proposed benefit, IOTA uses balanced ternary, or base three encoding, in its construction. Analogous to bits $\in (0, 1)$, a ternary digit is a trit $\in (1, 0, 1)$. According to founder David Sønstebø, the IOTA team chose ternary because they believed it is the optimal radix and a "superior technological solution" [14]. It's based on the claim that e has the lowest radix economy, or it needs the fewest digits needed to express a number. With fewer digits, calculations would be faster and more space-efficient. Since 3 is closer to $e \approx 2.71$ than 2, they chose the ternary system [14] and used it to modify WOTS to somehow strengthen the one-wayness property against hash-reversing quantum attacks [15].

To find a nonce that satisfies proof of work requirements, a quantum computer can hypothetically reduce the runtime from O(2^n) to O($2^{\sqrt{n}}$) (e.g. Grover's algorithm) [12]. As a solution, IOTA caps the maximum weight, or e amount of

work that the issuing node invested into it, [6] so that there are fewer nonces to check for (2^{64} vs 3^8), rendering a quantum computer's advantage less significant. However, as a protocol-level mechanism running on a binary computer network, using the ternary system grants no significant advantages. Compared to the binary logic common in cryptographic hash functions (i.e. SHA256), the ternary modification along with its in-house cryptographic primitive actually presents a weaker hash function that is a focus in the latter section. Meanwhile, since all computer hardware uses binary, IOTA converts to ternary in software, which is less efficient and more complex.

3 Vulnerabilities

3.1 Curl is not a collision resistant hash function

Perhaps the biggest problem of IOTA is the use of their in-house cryptographic hash function. IOTA replaced fundamental cryptographic primitives with proprietary, unvetted hash function called Curl as a prevention against quantum attack. It served as a vital component of signatures in IOTA such as address creation, message digest creation, PoW and hash-based signatures. The founder of IOTA infamously claimed:

> *"Don't roll your own crypto" is a compulsory uttered mantra that serves as a good guiding principle for 99.9% of projects, but there are exceptions to the rule." [16]*

In September 2017, Heilman et al. published a vulnerability report revealing the critical flaws in Curl's collision resistance property that can result in invalid transactions to be posted. [17] Recall that for a hash function h to be collision resistant, an adversary cannot produce two inputs x, x' s.t. $h(x) = h(x')$ with non-negligible probability k. In response, IOTA argued that because the centralized Coordinator validates transactions and helps address the problem with collisions, it is sufficient for the security of the schemes to rely only on Curl's one-wayness property. Yet, there is no security proof for this, and much of the network's central functionality is closed source. Below we prove why Curl is not collision-resistant and why this property is still relevant and crucial to the overall security scheme. Contrary to IOTA's response, this vulnerability makes it possible to forge signatures on IOTA payments regardless of the Coordinator.

Curl follows the pattern of a Sponge Construction. First, it initializes an all zero state memory S of length 729 trits. Then, it breaks the message into n message blocks, each with 243 trits. In each of the 27 rounds, each message block is copied into the first third of S, and then the state S is recursively altered via the transform function. Finally, after iterating through all of the message blocks, Curl returns the first third of the final state as the hash output. It is not hard to see that Curl is deterministic.

```
def Curl(message):
  #the state consists of 729 trits. It is initialized to all zero.
  state = [0]*729

  # The message is broken into message blocks of size 243
  MB_0, MB_1, ... MB_n = split(message)
  for MB_i in MB_0, MB_1, ... MB_n:
      # The current message block is copied into the first 243 trits of the state
      state[0:243] = MB_i
      state = Transform(state)

    # The output is the first 243 trits of the state
    output = state[0:243]
  return output

sbox = [[1, 1, -1],
      [0, -1, 1],
      [-1, 0, 0]]

def permute(x, y):
      Return sbox[x + 1][y + 1]

def Transform(state):
    for round in range(27):
      i = 0
      new_state = [0]*729
```

```
27
28      for pos in 729:
29          i = j
30          j += (364 if j < 365 else -365)
31          x = state[i]; y = state[j]
32        new_state[pos] = permute(x,y)
33
34        state = new_state
35  return new_state
```

Listing 1: Overview of Curl

We can create a collision by constructing two messages of the same length that are different at a single trit position and still hash to the same output value. First, we choose two messages that are at least three message blocks long and differ at only a single trit. Ideally, after 26 calls to the transform function on these blocks, differences between the messages will be captured in just the first third of the resultant states. Then, Curl would overwrite the first third of the state with the next message block, ultimately leading to a full state collision.

The technique used to analyze the propagation patterns of difference between multiple inputs is formally called differential cryptanalysis. Specifically, Heilman et. al found a differential trail, a probabilistic bias of how a set of differences will propagate to argue that Curl has a strong bias towards maintaining a 1-trit difference across rounds, and thus it is not collision resistant. After many rounds of the transformation function, we want to limit the propagation of the differential. Thanks to Curl's construction, after l rounds after a state with only one differential, the number of differentials is at most 2^l, and finding two messages that would create a collision requires at most 7.6 million or less than 2^{23} queries for Curl [17]. We can therefore brute force different messages to increase the number of rounds for which a 1-trit difference is maintained and ultimately find a collision.

Using this process, here is an example of a collision:

```
hash(ACMUXEIFDOIVQMVZNXPNWGSA9JGCN9RIMWOYNFLAVLBKR
JPKRAYFCGSD9CAJEFVPHIWRZEKQHUHCAKKSTXMDZMMVEVVCTQF
RTMDR9QLPG9QUWBHBQBVOPDWDIOFUWBK9IREKOUVRHDOD
LLXCLMJWZZXENYXDUSVDGU)= hash(ACMUXEIFDOIVQMVZNXPNWGSA9JGCN9RIMWOYNFLAVLBKR
JPKRAYFCGSD9CAJEFVPHIWRZEKQHUHCAKKSTXMDZMMVEVVCTQF
RTMDR9QLPG9QUWBHBQBVOPDWDIOGUWBK9IREKOUVRHDOD
LLXCLMJWZZXENYXDUSVDGU)= BUEXRNXFUP9HUMBOJWJZBQKDTZKOUVUXSJAXGKM
NH9I9EWNBXPBCFNEPBFCQFDYZZCBMXOTP9DOIMKEZ9
```

Via differential cryptanalysis, it is possible to generate collisions for messages such that IOTA's signature scheme is not secure against the Existential Unforgeability under a Chosen Message Attack (EU-CMA). EU states that an adversary cannot forge a signature for any message m. CMA is an attack in which before producing the forged signature on the message m', the adversary can obtain signatures for a specifically chosen message m. Thus, EU-CMA security guarantees that even if the adversary chooses two messages m, m', and the host signs m, the adversary should not be able to find a valid signature for m' in polytime.

Because IOTA's signature scheme signs $h(m)$ and $h(m) = h(m')$, a signature valid for m' will be identical to a signature for m. There exists a polytime way to find a colliding pair m, m' for Curl, s.t. it's possible to query a signature for m' and output it as a forged signature for the message m. Therefore, Curl's collision resistance property is necessary for the EU-CMA security of IOTA's signature scheme.

By producing pairs of different bundles that hash to the same value and share the same signature, it is possible to create invalid IOTA payments. A bundle is a unit of IOTA transaction, which consists of input, output, and meta transactions. With the possibility of generating collisions from two valid IOTA bundles, a following attack can occur when Alice tries to pay Bob:

1. Bob creates two colliding bundles: bundle1 and bundle2.
   (a) Bundle1 spends 10000 IOTA from Alice's address and pays 100 to Bob and 9900 to Alice.
   (b) Bundle2 spends 10000 IOTA from Alice's address but pays 1000 to Alice and 9000 to Bob.
2. Bob asks Alice to pay by signing bundle1.

3. Bob takes bundle1's signature and uses it on bundle2. He then broadcasts this to the network.
4. When bundle2 is confirmed, Bob stole 8900 IOTA from Alice.

Given Heilman's finding, the IOTA foundation has since then changed the code, hard-forked their system and changed all user addresses [16] [18]. In order to perform the upgrade, deposits and withdrawals had to be halted on Bitfinex for 3 days [18] [19], and all users who held IOTA had to upgrade their wallets and addresses [18]. In the end, they replaced Curl with a hash function called Kerl, which is a ternary version of the SHA-3 hash function [20].

3.2 Splitting Attack

Low number of users in a distributed ledger network can lead to slower tip approval speeds and reduced security, which is why bootstrapping is essential to initial adoption for DAG-based cryptocurrencies like IOTA. To aid in its effort, creators of IOTA have built in a centralized authority which they call "Coordinator" to safeguard IOTA's growth. Designed as a temporary measure, the Coordinator creates special transactions called Milestones that set the general direction for the Tangle growth and checkpoint valid transactions at certain time intervals. Transactions directly or indirectly referenced by milestones would be considered as confirmed.

However, the delegation of authority to a centralized supervisor poses threat in the event it is adversely manipulated. A malicious Coordinator could issue two different milestones which would split the network into two parts. The Coordinator could then keep incrementing the two milestones subtangle separately. Simultaneously, to ensure that 1) no subtangle overtakes the other and 2) to prevent the network from converging, the Coordinator could abuse IOTA's low hash power as a result of capping the maximum work per node for quantum security. Both subtangles would be sustained by spamming transactions on the smaller subtangle, giving enough time for the double-spent transaction to be confirmed.

3.3 34% Attack

As with any cryptocurrency, IOTA also has to prevent double-spending attacks. A double-spend attack is a successful attempt in approving a transaction that uses the same balance from a previous valid transaction that was later addressed to a different receiver twice and accepted as valid path in the graph. To attack the Tangle, an adversary would need to control 34% hash power of the network, which is lower than blockchain's 51%, to be able to create and verify false transactions. In fact, 34% is realistically attainable: a legitimate mining pool in Bitcoin called GHash.io attained 50% hash power in early 2014 [21]. Thus, IOTA, by the virtue of its nature, is more vulnerable to a double-spending attack.

In IOTA, the weight of a transaction denotes its importance and a likelihood measure of its validity proportional to the amount of work the issuing node has invested. In the DAG structure, where each transaction approves two other transactions, at least 34% of the hashing power is needed to create a larger weight transaction that will eventually outpace the network. This is because in order to double spend, an adversary must verify a valid transaction to steal funds from, and secondly approve an invalid transaction that spends the expended balance. A subsequent honest node can then choose to either ignore the false transaction and outgrow its confidence level, or counteract the double spend with an opposing offset transaction if it is already approved (i.e. damage has been done). IOTA is especially vulnerable to the 34% attack in its early stages when the network is small. The temporary solution to the 34% attack is also addressed by the Coordinator, which serves as a single administrator that signs the latest good state of the system. Thus, both the platform and currency are centralized today, meaning every user has to trust the IOTA foundation's Coordinator.

4 Conclusion

The lack of a financial incentive to provide hash power in a proof-of-work (PoW) system opens an avenue for new perspectives in distributed consensus in a decentralized network. By game theory, it is in rational human nature to take actions that maximize utility benefiting themselves, which Blockchain-based cryptocurrencies cleverly utilize. On the other hand, IOTA requires a different mechanism to have its participants well-behave. In fact, IOTA nodes are regulated by fair participation, where only in following the protocols are they able to receive the complete functionalities of IOTA. IOTA's creators designed it for M2M transactions and advocated feeless transactions as essential for scalability in the M2M transaction model. However, it seems like an oxymoron to assume self-governance yet rely so much of the system and functionalities on a centralized, omnipotent Coordinator. The presence of the centralized Coordinator also implies the product is unfinished and does not hold up to the image it presents as the next generation platform of promise. The developers claim the Coordinator merely serves as the training wheel in the beginning stage. However, almost four years after its creation, the Coordinator is still in-place. In addition, details and code regarding the Coordinator is undocumented or closed-source, meaning there's no way for general users to check how intertwined and perhaps dependent the whole IOTA system is to the Coordinator.

The cryptocurrency market is defined by agile environment, where developers are strongly incentivized to churn out the hottest features and create buzz, leaving less time to formalize their arguments and develop rigorous security proofs. However, it's looked down upon to invent a custom crypto primitive in production only to be found to violate fundamental industry best practices. Most people do not have the time or technical background to thoroughly vet the software with rigorous due diligence, meaning the trust is still needed. The general public should be aware of investigations like this, and large organizations definitely should not lend their names and reputation to technology they have not vetted. Currently, because of the crucial dependency of the Coordinator in resolving its critical security flaws, IOTA remains nothing more than an ambitious theory.

5 Appendix

The following process describes how the collision can be created between two messages that are identical except for a single, chosen trit position. It consists of two phases: a constraint phase and a brute force phase.

In the constraint phase, we can exploit the Curl's transform function's slow propagation of differentials. If a trit is modified in the state of the sponge, the transform function propagates this difference through the entire state very slowly. For a sponge state s_l, if there is a difference at position p, this difference will propagate to at most the positions $727p \pmod{729}$ and $727p - 1 \pmod{729}$ in s_{l+1}.

In the Curls' transform function, a set of indices $i_0, \ldots, i_{728}$ is generated by $i_j = 364j \pmod{729}$. To find the value at position j in state s_{i+1}, we apply S-box to the values $i_j = 364j \pmod{729}$ and $i_{j-1} = 364(j+1) \pmod{729}$ in the state s_l. 727 is the modular inverse of $364 \pmod{729}$. Given that position j in state s_l is determined by positions $364j$ and $364(j-1)$ in state s_{l+1}, a given state p must satisfy either $p \equiv 364j \pmod{729}$ or $p \equiv 364(j-1) \pmod{729}$. Then, $j \equiv 727p \pmod{729}$ or $j \equiv 727p - 1 \pmod{729}$, and only these states are affected. Thus, at any state where only one differential is present, the next state can have at most two differentials, and at each round of transform, we can at most double the number of differentials in our state.

Because we change one trit in the input and a single differential is introduced in the initial state, we must contain the spread of this differential in the goal of finding a collision. Fortunately, because of the Curl's S-box's modular structure when computing the permutation, the differential will appear in the first position of one permute operation and the second position of another operation. If the differential appears in the second position, a first input of 1 allows the differential $\{1, 0\}$, and the first input of 1 allows the differential $\{0, 1\}$.

		y		
		-1	0	1
x	-1	1	1	-1
	0	0	-1	1
	1	-1	0	0

Figure 2: Propagation of differentials in permute

Thus, to limit the propagation of a differential throughout the rounds of transform, we simply express each round as a recursive function of S-boxes and ensure that the first input is constrained when a differential occurs in the second input of an S-box such that the differential does not propagate. This creates a set of constraints that ensure that any input that satisfies the constraints will not propagate the differential. Note that these constraints will constrain both positions in the input and positions in the initial state of the sponge. Therefore, we must brute force search for an input that generate such a state.

Unfortunately, this is only feasible for a small number of rounds (e.g. 8) due to the exponential blow-up of these constraints. Once a match is found, however, we have a message that can be modified at a large set of positions so that the difference at our fixed position will not propagate for eight rounds. Because the digest is simply extracted from the sponge state's first 243 trits, we must ensure that all differentials are in the sponge state's last two-third trits (size 486) after all 27 rounds. The number of differentials is at most 2^l after l rounds on a state with only one differential. We note that there is a maximum of at most $2^7 = 128$ differentials after 7 rounds, all in a contiguous region. We have produced a collision as long as this region lies entirely within the last two-thirds of the sponge state.

Therefore, we need to find an input satisfying the precomputed constraints that contains only one collision after an additional 12 rounds of hashing. If this is the case, with high probability, the entire differential region will lie with high probability in the sponge state not used to produce the digest, creating a collision. There are a total of $486 - 128 + 1 = 359$ offsets at which the final differential may begin such that they will completely reside within

the last two-thirds of the sponge state. Since the differential position after the state of l is bijective in p, the original differential position, our final differential will lie in the desired range with probability $\frac{231}{729} = 0.492$. Because the probability of differential propagation at each round is relatively low, the brute force search is feasible on a commodity hardware.

References

[1] B. Garner, "Description of iota," 2018.

[2] J. Clark, "What is the internet of things?," 2016.

[3] "Iota history."

[4] K. Croman, C. Decker, I. Eyal, A. E. Gencer, A. Juels, A. Kosba, A. Miller, P. Saxena, E. Shi, E. Gün Sirer, D. Song, and R. Wattenhofer, "On scaling decentralized blockchains," in *Financial Cryptography and Data Security* (J. Clark, S. Meiklejohn, P. Y. Ryan, D. Wallach, M. Brenner, and K. Rohloff, eds.), (Berlin, Heidelberg), pp. 106–125, Springer Berlin Heidelberg, 2016.

[5] "Meet the tangle."

[6] D. S, "The tangle," 2018.

[7] "Iota wiki."

[8] P. W. Shor, "Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer," *SIAM review*, vol. 41, no. 2, pp. 303–332, 1999.

[9] J. Lin, "Quantum computing and its effects on deciphering public key encryptions," *Ethical Hacking and Systems Defense*, 2015.

[10] J. Buchmann, E. Dahmen, S. Ereth, A. Hülsing, and M. Rückert, "On the security of the winternitz one-time signature scheme," *International Journal of Applied Cryptography*, vol. 3, no. 1, pp. 84–96, 2013.

[11] J. Rompel, "One-way functions are necessary and sufficient for secure signatures," in *Proceedings of the twenty-second annual ACM symposium on Theory of computing*, pp. 387–394, ACM, 1990.

[12] L. K. Grover, "A Fast quantum mechanical algorithm for database search," 1996.

[13] J. P. Njui, "Europol arrests suspect behind the theft of $11.3 million in iota (miota)," Jan 2019.

[14] D. Sønstebø, "Iota founder's explanation for using ternary," 2018.

[15] D. S, "Iota multi-signature scheme," 2017.

[16] D. S, "Upgrades updates," *IOTA*, Aug 2017.

[17] E. Heilman, N. Narula, G. Tanzer, J. Lovejoy, M. Colavita, M. Virza, and T. Dryja, "Cryptanalysis of curl-p and other attacks on the iota cryptocurrency,"

[18] iFinex Inc., "Iota protocol upgrade," *Bitfinex: Announcement*, Aug 2017.

[19] D. S, "New gui + transition phase explained," *IOTA*, Aug 2017.

[20] E. Alon, "Iota kerl." `https://github.com/iotaledger/kerl`, 2017.

[21] A. Wilhelm, "51% fears rattle the bitcoin community," 2014.