DOSing Distributed Ledger Technology: IOTA

Mark A. Brady
University College Twente
Enschede, The Netherlands)
m.a.brady@student.utwente.nl

Ikram Ullah
Pervasive Systems Group, Dep. of
Comp. Science University of Twente
i.ullah@utwente.nl

Paul J. M. Havinga
Pervasive Systems Group, Dep. of
Comp. Science University of Twente
p.j.m.havinga@utwente.nl

2021 IEEE 5th International Conference on Cryptography, Security and Privacy (CSP) | 978-1-7281-8621-4/21/$31.00 ©2021 IEEE | DOI: 10.1109/CSP51677.2021.9357600

Abstract—**With advancements in connected technology, the number of ambitious applications involving Internet of Things (IoT) are drastically growing. This increases concerns related to security, scalability, and interoperability of IoT. As the network of connected devices grows, decentralized technologies become inevitable. Within this trend towards decentralization, distributed ledger technology (for instance IOTA) will be a significant driving force. IOTA is an innovative distributed ledger technology targeted towards low power devices, where energy efficiency is a high priority. Public research regarding security threats against IOTA especially denial-of-service (DoS) is essentially non-existent. In this paper we focus on exploring a DoS attack against IOTA. The proposed attack methodology takes advantage of the lack of fees along with the ability to transfer minuscule amounts. By sending many conflicting transactions as it results in a high number of re-attachments. The high number of re-attachments threatens IOTA's suitability for the IoT sphere. The implications of such attack, as well as the future of this issue in terms of the planned removal of the centralized coordinator are discussed.**

Index Terms—internet of things, decentralization, security, blockchain, IOTA, vulnerability

I. Introduction

IoT is an umbrella term that includes different technologies such as sensor networks, smart phones, and cloud services [1]. It is one of the most disruptive technologies of the present century [2]. Data generated in an IoT environment can be used to provide services that bring efficiency and ease to our lives. IoT applications are feasible thanks to the large amount of data shared among connected devices. However, the large bulk of data generated can be used for malicious purposes such as exploiting privacy [2], [3]. Existing IoT architectures are based on a centralized model, which requires all devices to be authenticated through a server [4]. Centralized models are not suitable for the fast outspread of connected IoT devices [5], as they pose many risks in terms of trust, security, overhead, and scalability [6]. As the number of connected devices increase, the demand of connectivity puts enormous pressure on existing centralized architectures.

One way to overcome these challenges is through decentralized architectures, such as Distributed Ledger Technologies (DLT) [7]. In essence, a DLT can be described simply as a publicly recorded ledger, in which many different individuals may take part in verifying the ledger. In DLTs the storage and validation of data does not need to be left in the hands of some third party running a server. Additionally, transactions made on the network may be viewed by anyone. Thus it allow businesses and governments to act more transparently. DLTs have the potential to enable economies independent of the financial institutions industry currently relies upon. DLTs also have the capacity to enable decentralized services which are resilient to malicious actors. Security and immutability are other main advantages in considering the suitability of a DLT. There are various DLTs, for instance Blockchain [6], Hashgraph [6] and *tangle* (IOTA) [8]. IOTA is built to be suitable for IoT applications. However, the bandwidth and costs associated with Blockchain technologies are significant barriers to adoption [9].

IOTA is a DLT which has aimed to combat issues in scaling, while also removing the necessity of transaction fees, and enabling its use in the IoT applications [8]. Were these two issues to be solved, it would enable minuscule transactions to be made, allowing for the adoption of this technology within the context of economies wherein fractions of a cent make up some portion of transactions. Such an envisioned economy is the Machine-to-Machine (M2M) economy. This system would enable a larger degree of autonomy and security in the context of many IoT applications. The integration of IoT and DLT provides many benefits, such as resiliency, security, immutability, and anonymity [4]. An example of real world IOTA application would be ownerless cars, which may lease out and get maintenance themselves [10]. Other potential use cases include factory automation and logistics, where every asset could have its own IOTA wallet [11], and instead of every actor having to log information with each other, they can log it on IOTA, which can be viewed almost immediately by whoever is authorized [12].

Currently, IOTA is still in development, though it is open for use. Efforts are underway to remove the Coordinator, going by the name of Coordicide, in order to make the protocol truly decentralized [13]. We have not seen much published research on the topic of the IOTA protocol. This seems like a significant research gap. In the context of IOTA security, published research largely focuses on methods to perform double spending [8], [14], [15], [16], wherein the attacker effectively spends the same tokens twice. Little to no research on methods to perform DoS attacks have been published as of July 2019. A DoS attack is where the aim is not to unfairly spend or accrue resources, but to reduce availability, preventing users from accessing the service, and to waste resources by doing so. In this paper, we have proposed a DoS attack methodology within the current IOTA implementation (version 1.7.1). This is among the first academic works investigating DoS attacks in the context of IOTA, and the results are promising, with a relatively inexpensive cost to execute the attack. Finally, the

978-1-7281-8621-4/21/$31.00 ©2021 IEEE

implications of these findings are discussed, particularly with respect to currently planned changes for the official IOTA Reference Implementation (IRI) mentioned in the Coordicide whitepaper [13].

II. Our Contributions

In this research, a potential DoS attack methodology is investigated. The mechanics of this attack are considered, as well as a potential optimized variation on this attack. This attack is tested on a local network, as well as on the official developer network.

III. Iota

IOTA is a tangle-based distributed ledger technology [8]. The *tangle* is a directed acyclic graph (DAG) used as the primary data structure to store transactions. A new transaction that is not approved by any other transactions yet, is called a tip. For a transaction to enter the *tangle*, it must be signed by the one performing the transaction, two other tips must be selected as shown in the Figure 1, and a proof of work (PoW) must be performed to hash this transaction. PoW is perform in order to prevent spam [8]. A tip selection algorithm is applied to select which tip transactions to validate. The current tip selection algorithm is a Monte Carlo Markov Chain walker (MCMC) [8]. These walkers transition from transaction to transaction randomly, but with a certain probability [8] [17]. This is based on the difference of cumulative weights of a starting (H_x) and potential (H_y) site [9]. The probability for a transition is shown here [8], where α is a variable, and the H represent the cumulative weight of their respective sites:

$$P_{xy} = exp(-\alpha(H_x - H_y))*(\textstyle\sum_{z:z\rightarrow x}\ exp(-\alpha(H_x - H_z)))^{-1}$$

Every transaction, has a set of weights associated with it. These include a transactions own weight, the cumulative weight, and the score. The own weight is dependent on the amount of effort put in by the issuer [8]. Cumulative weight is the sum of this own weight, and the own weights of all transactions that approve the considered transaction. These weights give certain confidence in a transaction validity [8]. To build weight in a tip, either more work needs to be put into validating transactions, or other transactions must see it, recognize it does not conflict with other referenced transactions, and reference it.

IOTA relies on the assiduous honest majority [18]. This can be partially achieve through the existence of the PoW, limiting spam on the network. However, the attacker might render the network vulnerable if he has computational power of over a third of the power being used by everyone else on the *tangle*. Right now IOTA is not that mature and the rate of transactions is still low as the technology has not widely been adopted, so this threat can significantly impact the IOTA credibility [19] [20] [21]. Should a large portion of the hashing power lie in the hands of some adversary, who could potentially perform double-spending. So therefore instead of MCMC, a centralized transaction issuer, called a coordinator, was developed [22]. This coordinator issues a transaction (a milestone) every number of seconds, and for a given transaction to be considered valid, it must be referenced (directly or indirectly) by a milestone.

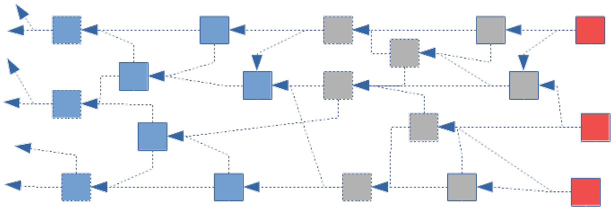


Fig. 1. Typical tangle format, where blue is deep enough to be confirmed, grey is yet to gain enough weight for confirmation, and red are tips without any references.

IV. Iota Security

Like any other DLT, IOTA is potentially vulnerable to certain attacks such as: Large weight attack [8], Waste money attack [14], Steal money attack [14], 34% attack [19], Replay Attacks [23], Curl-P hashing collisions [14], and the Splitting attack [8]. There are also concerns of its Centralization [14] [24], and vulnerabilities to related issues. In this paper we explore and exploit DoS attack.

A. Denial-of-Service Attack

A DoS attack occurs when an attacker directs enough traffic to a system such that little legitimate traffic can access it [25]. A distributed denial-of-service (DDoS) attack is one wherein an attacker can prompt other sources to direct more requests to their target than is typically allowed. An example of this would be the Dyn cyber-attack in 2016 [26]. Such attack may worsen as the IoT becomes more prevalent as there will be more devices to infect.

B. IOTA Denial-of-Service Attack

In the current implementation (version 1.7.1) [27], it's the case that a single spammer with specialized hardware can significantly reduce network throughput for as long as they like. There are a number of targets an attacker could focus on, should they want to disrupt the services of an IOTA based application. It is typical for users to tunnel their transactions to a node before it becomes authenticated on the *tangle*. These nodes may be public or private. As long as these nodes' addresses are known, it may be targeted. Alternatively, if the address of the targeted node is not known, it is possible that the neighbors of the node are, and through disrupting them, the targeted node could subsequently be denied service. In terms of targeting single nodes, there is no built in defense mechanism provided by the IOTA protocol, but it is typically handled through a reverse proxy such as Nginx [28].

The behaviour of many individual conflicting transactions to delay transaction confirmation is typically utilized to perform double-spending, wherein one would create a split (splitting attack), and try to maintain two separate branches, allowing the same tokens to be spent differently on each branch. In the current implementation (version 1.7.1), this splitting attack is not viable as a double-spending attack, as a transaction is only considered confirmed once the coordinator posts a milestone referencing one of the branches and the previous milestone [29]. However, the concept of repeated conflicting transactions can be used to cause delays thus denying service.

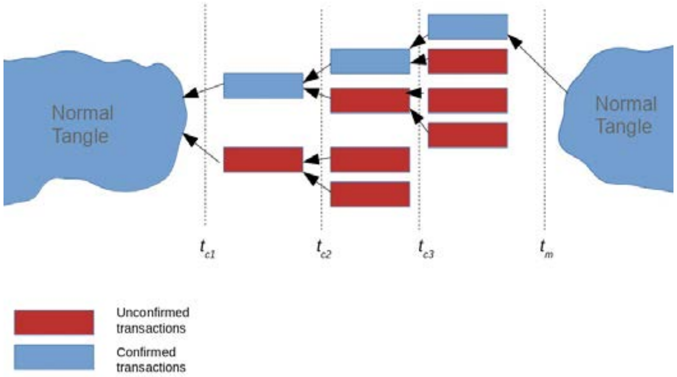


Fig. 2. Illustration of split-spam attack.

When a new transaction is attached to the *tangle*, it also must select two new tips to reference. When there are a number of conflicts among the tips, any may be chosen. However, transaction may not refer to a set of conflicting transactions, or transactions which refer to conflicts. As a result, branches may start to form at every conflict, 'fraying' the *tangle*. Only one of these branches may be considered valid, so once the coordinator places another milestone, all transactions off of this primary branch no longer have a chance to be validated. An illustration of this is shown in Figure 2. This lowers the amount of confirmed transactions per second, and may do so significantly either with a sufficiently optimized algorithm, or a sufficient number of splitting transactions.

In this paper, we explore two different attack methodologies: Naive Split-Spam, and Optimal Split-Spam to perform DoS attack. Naive Split-Spam attack is practically exploited and proof-of-concept is presented. Optimal Split-Spam which is an optimized form of the Naive Split-Spam attack is proposed. The optimal attack discussed has not been tested, but merely reasoned about. Both the methodologies are described as follows.

C. Naive Split-Spam Attack

In order to perform this attack, from some primary seed, tokens are transferred to many additional wallets, each receiving one token. These additional wallets are also controlled by the attacker. Once these have been confirmed by the coordinator, the attack may begin. Some *n* transactions are sent concurrently from one of these additional wallets at a time, with each transactions being the total value contained in the wallet, and each transaction going to a different address. The tip selection algorithm used by the attacker may be the default MCMC walker, though this is likely not optimal. This allows for an *n*-way split of the *tangle* branch on which they are placed. This is done as many times as possible in between milestones, and in that time, the *getTransactionsToAcknowledge* API call may recommend any of the attacker's split branches when honest transactions try to attach to the *tangle*. Once a milestone is sent by the coordinator, then a specific branch is chosen and the others will no longer be referenced. This may be repeated indefinitely. As shown in the Algorithm 1. The resources required for this attack, compared to its impact, are minimal.

Algorithm 1: Naive Split-Spam Attack

```
foreach i in seeds do
    from primarySeed send 1 IOTA to i;
end
while transactions are not confirmed do
    wait;
end
foreach i in seeds do
    from i send 1 IOTA to receiverSeed1;
    from i send 1 IOTA to receiverSeed2;
    wait for next milestone;
    recollect IOTA from receiverSeed1 and
      receiverSeed2 into primarySeed;
end
```

However, there are certain inefficiencies in Naive Split-Spam attack, since at the moment a set of conflicts are first sent, they are as likely to be chosen as any other tip. However, as time progresses and more transactions come in, there is some chance that weight will grow unevenly on these conflicts, leading one branch to being chosen significantly more. This reduces the amount of transactions to become orphaned. To maximize the number of transactions orphaned, the attacker would want to attempt to balance these branches, again, similar to the methods a splitting attack aimed at double-spending would employ.

D. Optimal Split-Spam Attack

An optimal attack based off of this mechanic requires the ability to maintain splits for some time. Suppose it takes t_s seconds to send a batch of n transactions spending the same iota. There are some x transactions per second being issued, of which approximately 100% would normally be confirmed. At every milestone, issued at intervals of length t_m, all branches except for one collapse. Within that time, t_m/t_s conflicting batches may be issued. The batches should be attached to one specific branch, resulting in an addition of $n-1$ new branches. Attaching them to different branches does not create further branches, as within the viable branch, or sub-*tangle*, to which they belong, there will be no conflict. Assuming all of these branches are sustained, the new rate of confirmation becomes:

$$\frac{x}{(n+(t_m/t_s)*(n-1))}$$

If there is no conflict, the rate of confirmation remains at x. The first time a conflicting transaction is sent after a milestone, only the main *tangle* is being split, though in subsequent conflicts one of the attackers branches must be attached to, causing the $n-1$ new splits. This formula for the new rate was derived from the expected behaviour, though it has not been verified.

There are certain challenges in maintaining multiple branches. To sustain multiple branches, the state of the network must be monitored closely. If any branch grows too much, incoming transactions will prefer to reference that branch, thus further creating an imbalance. This must be prevented by the attacker by sending an appropriate amount of transactions to the shorter branches to bring them close to the longest branch. Problems arise on a larger network, where the ledger state between two nodes might be different enough that users of that node may be creating significant weight or alternative branches that the attacker is unaware of until some time later. The rate of transactions may also be too great for an attacker to easily overcome, if a branch collapses too quickly, there may be little an attacker can do. Were it simple to maintain balance between separate branches, the only consideration the attacker would make is making as many different branches as possible. This may encourage a large amount of splitting transactions within a single batch, at the cost of a lower number of these splitting batches. However, as previously stated, maintaining branches is not necessarily an easy task. As a result, a smaller number of splits per batch in favour of producing more batches may be preferable, depending on how quickly a branch is outweighed to the point of being orphaned. With a smaller number of splits of larger weights, there is not a significant amount of time for a branch to be significantly outpaced by other honest branches without splits being created - though there is a chance that these other branches may persist naturally.

V. Denial-of-Service attack Implementation

A. Attack Environment

We have tested Naive Split-Spam attack methodology in two environments - a local network using IOTA Reference Implementation (IRI) (version 1.7.1), utilizing the newly implemented open source coordinator, and the public developer (main) network, or Devnet, running IRI version 1.7.1 as shown in Figure 3 and Figure 4. Along with topological differences, these two networks are also different in the amount of proof-of-work required. Additionally, these two networks differ in the amount of load on them. Within the main *tangle*, 10 transactions per second may be seen. However, on the developer network, the rate is typically closer to 0.2 or 0.3 transactions per second. While this method was tested at base speed, and at higher speed. Tests performed at higher speed, zero-value transactions were continuously sent through the networks to increase the rate to something more akin to what would be experienced in real world applications. On the local *tangle*, there was no external traffic. As a result, all of the traffic used to test was also generated through zero-value transactions. The local network as shown in Figure 3 consists of three nodes. One node run Compass, the opensource coordinator, while the other two only run IRI (version 1.7.1), the standard full node implementation. This set of experiments aimed to explore the response and mechanics involved when many conflicts are created on a network. The primary factor considered within this environment is network latency, with the thought that it may produce differing ledger states. The question then is would this Split-Spam behaviour be more effective if different ledger states could be held, compared to when the ledger state is consistent, as is the case when a single node is targeted. Latency between the neighbours is induced by splitting behavior, with tokens being sent to different addresses based on which node was being communicated with, for instance the total balance of a seed was sent to *address1* through *node1*, while this same amount was sent to *address2* through *node2* simultaneously. These two nodes introduce latency on the IRI neighbouring ports via the tc tool [30].

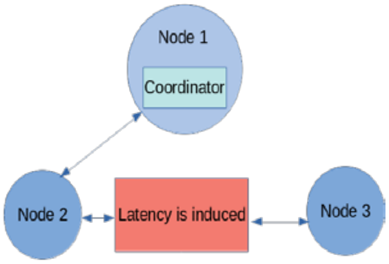


Fig. 3. Local Network with latency induced between test nodes.

The main network has many independent nodes, peering[1] with other 'neighbours' to share the transactions that they receive through a gossip protocol. In this case, a transaction may have to make many hops to arrive at any given node, and so the ledger state for two different nodes could be different at any time. In both the developer network and the local network experimented with, there is a minimal number of nodes between any two endpoints. This makes differing ledger states unlikely. To try to replicate this, network latency was introduced between two nodes on the local network in one of the tests. Data was collected through IRI nodes using ZMQ [31].

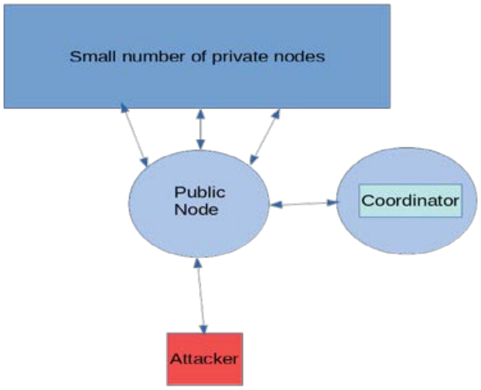


Fig. 4. Illustration of split-spam attack.

B. Attack Execution

Several thousand IOTA tokens were stored with a source seed. A number of new seeds were generated, and addresses were generated for each using the Pyota [32] method *get_new_addresses*. To each of these new addresses, a single IOTA transaction was generated as a *ProposedTransaction* object from the source seed. These transactions were bundled and sent together all at once using the *send_transfer* method, which also does the PoW and broadcasts to the IRI node. After some time, a milestone would refer to this bundle, giving the new seeds a single IOTA each. Once this was done, two single

[1] UDP was used as opposed to TCP due to some error within the Docker environment preventing TCP usage.

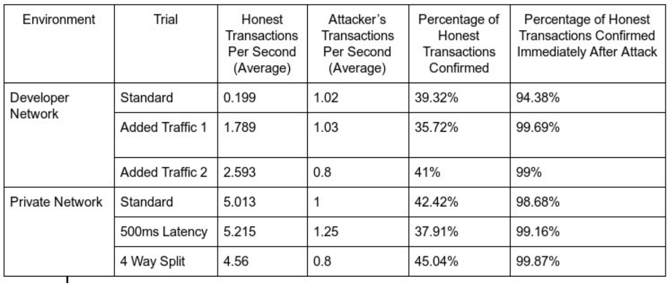
IOTA transactions were prepared, going to two different addresses. A list of APIs for each new seed was cycled through, and from each, these two different single IOTA transactions were sent (again using *send_transfer*), creating a conflict. Once all new seeds were sent from, the attack ceased. In the case where the effect of latency is being tested, each one of the two conflicting transactions was sent to a different node. Additionally, spam was generated to test the effects at different transaction rates. This was done by generating zero-value transactions from a single seed, and repeatedly using the *get_transactions_to_approve* method, which uses the standard tip select algorithm from the node the API object is declared with. The PoW is done through the *attach_to_tangle* method, and then it's sent using *broadcast_and_store*. Again, when testing the effects of network latency, this process is run concurrently for each of the two nodes being used.

A. Results and Discussion

As mentioned earlier, the proposed attack methodology is performed on local network and developer network. Evaluation for both the networks is provided as following.

1) Local network: The results of latency induced among two nodes is shown in Figure 5. No significant effect was visible with the varying latency rates, the drop in confirmation rates were roughly consistent across all the tests. For instance, as shown in the Table 1 the percentage of confirmed transactions dropping by approximately 58% without induced latency, and 62% with induced latency of 500ms. In this context, a variation of 4% is not dramatic enough to say latency had an effect on the efficacy of the attack, suggesting the mechanism of attack is local, and not due to network effects. Inspection of the visualization of the *tangle* during the attack in these two environments also revealed no significant difference. On the successful demonstration of this attack in this environment, it was discovered that conflicts were accepted as valid tips even if a node was aware of the conflict.

TABLE I. RESULTS FROM SEVERAL TESTS



Environment	Trial	Honest Transactions Per Second (Average)	Attacker's Transactions Per Second (Average)	Percentage of Honest Transactions Confirmed	Percentage of Honest Transactions Confirmed Immediately After Attack
Developer Network	Standard	0.199	1.02	39.32%	94.38%
	Added Traffic 1	1.789	1.03	35.72%	99.69%
	Added Traffic 2	2.593	0.8	41%	99%
Private Network	Standard	5.013	1	42.42%	98.68%
	500ms Latency	5.215	1.25	37.91%	99.16%
	4 Way Split	4.56	0.8	45.04%	99.87%

Conflict is caused by simply sending the balance of a seed to different addresses. In first case, the conflicting transactions are sent to two different addresses. In this case the observed drop in confirmation rate is 58%. In the second case, the conflicting transactions are sent to four different addresses. In this case, the confirmation rate dropped by 55% for honest transactions. This was not a significant difference, though that may be due to the rate of transactions being reduced in favour of these larger bursts of conflicting transactions. The lack of effect may be explained by the fact that twice the amount of transactions takes twice as long to send, and every new branch created is equally likely to collapse.

2) Developer Network: Various tests with different transactions per second were run on the developer network. The amount of traffic generated varies between the tests. For instance, a test with average 2.59 transactions per second is shown in Figure 6. The exact drop in confirmation rates is 58% respectively. Keep in mind that the number of attacker transactions remained constant throughout. The number of transactions orphaned varies, as traffic was not consistent. The impact is not necessarily constant. It can vary with changes in traffic. The amount of computational power needed is dependent on the PoW, for the most part, but also on other bottlenecks in the network. Results of various trials are shown in Table 1. As a proof of concept conflicting transactions from the developer network are shown in appendix. Note that the only trials with transactions searchable via devnet.thetangle.org are those performed on the developer network. Source code, visualizations, and all other resources may be accessed through our git repository at [33].

VI. CONCLUSION

In this paper we have discussed the mechanics of various DoS attack vectors. Based on our analysis, this form of attacks are computationally inexpensive and quite effective at causing service disruption. The results shown suggest a strong possibility of a weakness within the current IOTA implementation (version 1.7.1), the observed drop in confirmation rates being largely consistent across all trials. Additionally, the rate at which the attacker sends transactions to perform the attack was varied, so the effectiveness of the attack as a function of the attackers sending rate is determined. If a higher rate of attacker transactions per second results in a proportional decrease in honest confirmations per second, the effectiveness of this attack is much greater than it seems, with a reasonable amount of resources, it could shut down the entire network. Finally, considering the intent of IOTA is to function on low power devices. The energy cost of a single transaction is already an issue, forcing to attach a transaction multiple times may halve the battery life of these devices. The impact of this attack may differ given the current developments in IOTA protocol.

ACKNOWLEDGMENT

This work has been partially supported by the EFRO, OP Oost program in the context of Countdown project.

APPENDIX PROOF OF CONCEPT

1) MXQQXSPUSAONUJ9HQT9YFCYOPQDM9KFUED
PMLQMEFOUYQTLWMHQHYATCPLMMLSTDHE
PVMZRFEBLWPC999 (2019-06-30)
2) AVWUVUUJTXDEKOZHH9VYGKEWQILANNBFW
RORSNMPHVKNGGYNTIJHURXUJDBYMFWQSG
HJXNZZYBTDVI999 (2019-06-30)

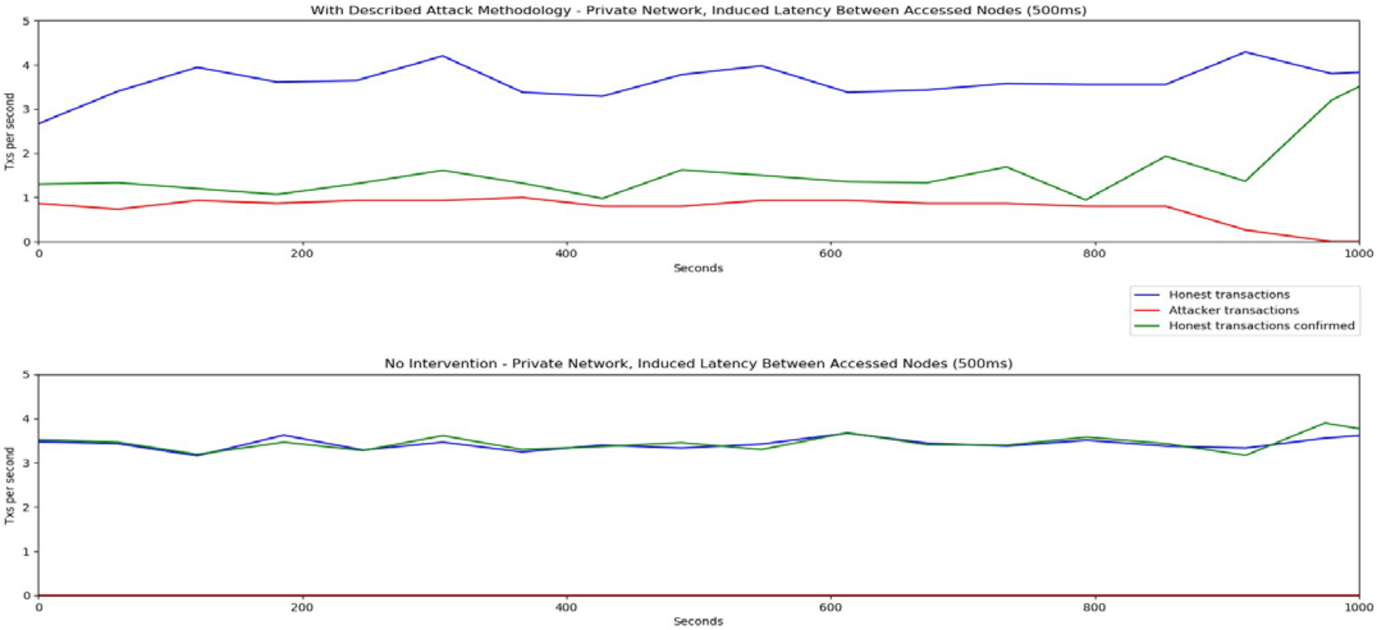


Fig. 5. 500 ms latency introduced between *node1* and *node2*, the lower figure is without any attack taking place, and the upper figure with the attack underway. The blue line represents the number of honest transactions sent to the network per second, red representing the number of transactions the attacker is sending per second, and finally the green line representing the amount of honest transactions which are confirmed per second.

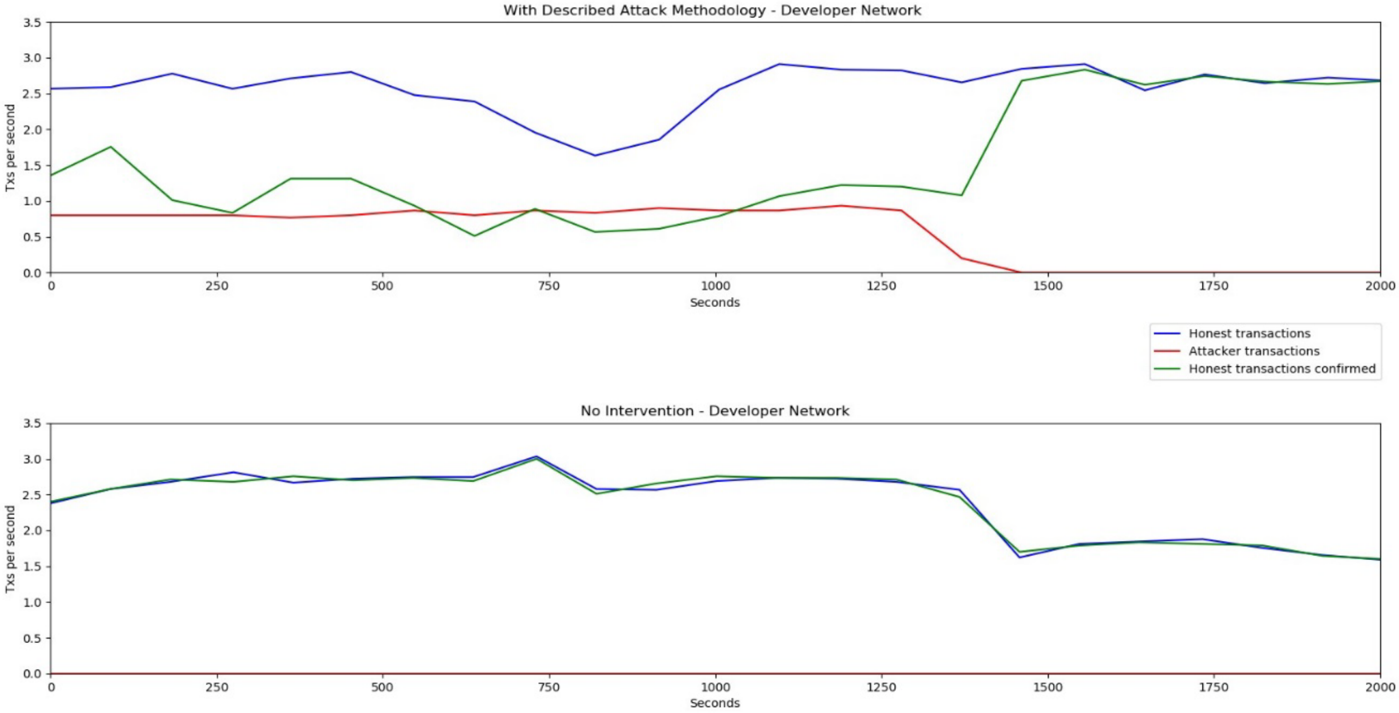


Fig. 6. Developer network trial 3, average honest TPS of 2.59

References

[1] J. H. Ziegeldorf, O. G. Morchon, and K. Wehrle, "Privacy in the internet of things: threats and challenges," *Security and Communication Networks*, vol. 7, no. 12, pp. 2728–2742, Jun. 2013. [Online]. Available: https://doi.org/10.1002/sec.795

[2] A. Dorri, S. S. Kanhere, and R. Jurdak, "Blockchain in internet of things: Challenges and solutions," *CoRR*, vol. abs/1608.05187, 2016. [Online]. Available: http://arxiv.org/abs/1608.05187

[3] M. L. Das, "Privacy and security challenges in internet of things," in *Distributed Computing and Internet Technology*. Springer International Publishing, 2015, pp. 33–48. [Online]. Available: https://doi.org/10.1007/978-3-319-14977-6_3

[4] H. F. Atlam, A. Alenezi, M. O. Alassafi, and G. B. Wills, "Blockchain with internet of things: Benefits, challenges, and future directions," *International Journal of Intelligent Systems and Applications*, vol. 10, no. 6, pp. 40–48, Jun. 2018. [Online]. Available: https://doi.org/10.5815/ijisa.2018.06.05

[5] E. Karafiloski and A. Mishev, “Blockchain solutions for big data challenges: A literature review,” in *IEEE EUROCON 2017 -17th International Conference on Smart Technologies*, July 2017, pp. 763–768.

[6] B. Cao, Y. Li, L. Zhang, L. Zhang, S. Mumtaz, Z. Zhou, and M. Peng, “When internet of things meets blockchain: Challenges in distributed consensus,” *CoRR*, vol. abs/1905.06022, 2019. [Online]. Available: http://arxiv.org/abs/1905.06022

[7] R. Maull, P. Godsiff, C. Mulligan, A. Brown, and B. Kewell, “Distributed ledger technology: Applications and implications,” *Strategic Change*, vol. 26, no. 5, pp. 481–489, Sep. 2017. [Online]. Available: https://doi.org/10.1002/jsc.2148

[8] S. Popov., “The tangle (white paper),” Tech. Rep., 2016. [Online]. Available: https://www.iota.org/research/academic-papers

[9] J. Yli-Huumo, D. Ko, S. Choi, S. Park, and K. Smolander, “Where is current research on blockchain technology?—a systematic review,” *PLOS ONE*, vol. 11, no. 10, p. e0163477, Oct. 2016. [Online]. Available: https://doi.org/10.1371/journal.pone.0163477

[10] Mobility & automotive. Accessed 2019-06-15. [Online]. Available: https://www.iota.org/verticals/mobility-automotive/

[11] Industrial IoT. Nov 2019. [Online]. Available: https://www.iota.org/verticals/industrial-iot

[12] Global trade &supply chains. Accessed 2019-06-15. [Online]. Available: https://www.iota.org/verticals/ global-trade-supply-chains

[13] C. Team., “The coordicide,” Tech. Rep., 2019, accessed 2019-06-15. [Online]. Available: https://www.iota.org/research/academic-papers

[14] E. Heilman, N. Narula, T. Dryja, and M. Virza, “Iota vulnerability report: Cryptanalysis of the curl hash function enabling practical signature forgery attacks on the iota cryptocurrency,” https://github.com/mitdci/tangled-curl/blob/master/ vuln-iota.md, 2017, accessed 2019-06-15.

[15] Y. Hu, A. Yip, and M. Shin, “IOTA Tangle and Cryptographic Vulnerabilities,” https://www.youtube.com/watch?v=vmwYCjcbUc8, 2017, online; accessed 2019-06-15.

[16] P. Staupe., “Quasi-analytic parasite chain absorption probabilities in the tangle,” Tech. Rep., 2017, accessed 2019-06-15. [Online]. Available: https://www.iota.org/research/academic-papers

[17] A. Gal, “The tangle: an illustrated introduction,” accessed 2019-06-15. [Online]. Available: https://blog.iota.org/ coordinator-part-1-the-path-tocoordicide-ee4148a8db08

[18] Q. Bramas., “The stability and the security of the tangle,” Tech. Rep., 2018, accessed 2019-06-15. [Online]. Available: https://www.iota.org/research/academic-papers

[19] D. Sønstebø. “The Transparency Compendium,” Apr. 2018. [Online]. Available: https://blog.iota.org/the-transparency compendium26aa5bb8e260

[20] “Official IOTA Foundation Response to the Digital Currency Initiative at the MIT Media Lab — Part 3. . . ,” Jan. 2018. [Online]. Available: https://blog.iota.org/official-iota-foundation-response-to-thedigital-currency-initiative-at-the-mit-media-lab-part-3-6433b55c7d57

[21] “Official IOTA Foundation Response to the Digital Currency Initiative at the MIT Media Lab — Part 4. . . ,” Jan. 2018. [Online]. Available: https://blog.iota.org/official-iota-foundation-response-to-thedigital-currency-initiative-at-the-mit-media-lab-part-4-11fdccc9eb6d

[22] “Coordinator. part 1: The path to coordicide,” accessed 2019-06-15. [Online]. Available: https://blog.iota.org/ coordinator-part-1-the-path-tocoordicide-ee4148a8db08

[23] J. Rebstock, “Replay attacks in iota,” accessed 2019-06-15. [Online]. Available: https://github.com/joseph14/iota-transactionspammer-webapp/blob/master/replay%20attack.md

[24] E. Wall, “Iota is centralized,” accessed 2019-06-15. [Online]. Available: https://medium.com/@ercwl/iota-is-centralized-6289246e7b4d

[25] “Cyber and infrastructure security agency. understanding denial-ofservice attacks | cisa,” accessed 2019-06-15. [Online]. Available: https://www.us-cert.gov/ncas/tips/ST04-015

[26] N. Perlroth, “Hackers used new weapons to disrupt major websites across u.s. - the new york times,” accessed 2019-06-15. [Online]. Available: https://www.nytimes.com/2016/10/22/business/internetproblems-attack.html5

[27] Jakub Cech., “Tip-selection time-out mechanism vol. 2 Issue 1421 iotaledger/iri,” https://github.com/iotaledger/iri/issues/1421.

[28] G. Mittendorfer, “Api call rate limiting issue 1304 iotaledger/iri,” accessed 2019-06-15. [Online]. Available: https://github.com/iotaledger/iri/issues/1304

[29] I. Foundation, “Iota reference implementation,” accessed 2019-06-15. [Online]. Available: https://github.com/iotaledger/iri

[30] “tc(8) - linux man page,” accessed 2019-01-09. [Online]. Available: https://linux.die.net/man/8/tc

[31] “Zmq events,” accessed 2019-06-15. [Online]. Available: https://docs.iota.org/docs/iri/0.1/references/zmq-events

[32] “Pyota,” accessed 2019-06-15. [Online]. Available: https://github.com/iotaledger/iota.lib.py

[33] M. Brady, “Split-spam dos iota,” accessed 2019-08-09. [Online]. Available: https://gitlab.com/MarkMk2/split-spam-dos-iota.git