Paavolainen, Santeri; Elo, Tommi; Nikander, Pekka

Risks from Spam Attacks on Blockchains for Internet-of-Things Devices

Published in:
2018 IEEE 9th Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON)

DOI:
10.1109/IEMCON.2018.8614837

Published: 01/01/2018

Document Version
Peer-reviewed accepted author manuscript, also known as Final accepted manuscript or Post-print

Please cite the original version:
Paavolainen, S., Elo, T., & Nikander, P. (2018). Risks from Spam Attacks on Blockchains for Internet-of-Things Devices. In *2018 IEEE 9th Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON)* (pp. 314-320). IEEE. https://doi.org/10.1109/IEMCON.2018.8614837

Risks from Spam Attacks on Blockchains for Internet-of-Things Devices

Santeri Paavolainen
School of Electrical Engineering
Aalto University, Helsinki, Finland
and LMF Ericsson, Finland
Email: santeri.paavolainen@aalto.fi

Tommi Elo and Pekka Nikander
School of Electrical Engineering
Aalto University, Helsinki, Finland
Email: {tommi.elo,pekka.nikander}@aalto.fi

***Abstract*—There has been increased interest in the use of blockchains to control Internet of Things devices either directly, or through smart contracts. Many blockchains, such as Ethereum and Fabric, have support for smart contracts. The use of public blockchains while attractive due to their decentralization and availability, do pose challenges, such as unpredictable transaction latencies and cryptocurrency price fluctuations. Transactions in the Ethereum network, such as invokations of smart contracts used to control an IoT device, have no fairness or eventuality guarantees. In this work we describe a "spam attack" method available to parties with sufficient cryptocurrency reserves to delay a statistically significant portion of transactions submitted to the Ethereum network. This paper derives estimations on the costs and effects of such an attack, and is based on an analysis of historical transactions.**

***Index Terms*—Internet of Things, Blockchain, Ethereum, Denial of Service, Smart Contracts**

I. Introduction

The interest in and the use of blockchains has increased in recent years. The introduction of blockchains into the general public knowledge came through the development of Bitcoin, a blockchain focused on trading in a cryptocurrency of the same name. The most common use of public blockchains remains cryptocurrency trading, with an estimate of the public cryptocurrency and cryptotoken market capitalization being in excess of $250 billion[1]. While cryptocurrency transfers are a foundation of all public blockchains, more recent blockchains, such as Ethereum, have introduced new functionalities, of which *smart contracts* are probably the most notable. Smart contracts are pieces of program code that become part of the blockchain, and can be used to implement features on the generic programming model that is available. Ethereum is the second most popular blockchain in terms of market capitalization, and thus the most popular one that has smart contract support.

The use of smart contracts to facilitate decentralized authorization and control operations on IoT devices have recently been investigated. In this context, Tapas et al. [1] describe a system where a smart contract provides information on the allowed operations specific roles are allowed to take on an IoT device. The use of smart contract allows auditability of all access policy changes, and the inherent decentralized and distributed nature of the blockchain removes the need of a centralized access control server. Novo describes a similar system in more detail [2], which takes the needs of resource-constrained IoT devices into account by introducing *management nodes*, that act as trusted proxies to the blockchain. Regardless of the existence of separate management nodes, the access policy is managed by a smart contract, and provides transparency on the system. Both of the described systems rely on the security of the blockchain, and, as we will show later, to meet real-world security assurances they also rely on timely execution of transactions for altering the authorization policies, for example, to revoke a permission.

The underlying security challenge of public and permissionless blockchains (such as Bitcoin and Ethereum) is the need to protect against Sybil attacks [3]. To achieve the needed distributed consensus in this situation, a proof-of-work aka Nakamoto consensus is employed [4]. In a proof-of-work consensus model, parties called *miners* continuously attempt to generate a proof of work that allows them to *mine* a new *block* (hence the name blockchain). Protection against Sybil attacks is provided by using a proof of work that is costly to produce, thus increasing the cost of mounting such an attack substantially. With the proof being difficult to produce, but easy to verify, other nodes on the network are able to validate the block and its proof of work, and will include the new block as the head of the blockchain. Thus, each mined block becomes part of the chain of blocks, and in turn, through its proof of work, validates earlier blocks. The decentralized and costly process of mining guarantees the integrity and immutability of the blockchain under the assumptions of Nakamoto consensus.

Blockchains are known to be susceptible to various attacks that can either break their security guarantees, or prevent them from operating normally. These can be roughly categorized as either 1) attacks on the *security guarantees*, such as integrity of the blockchain, 2) attacks on the underlying *infrastructure*,

[1]From coinmarketcap.com, as of July 1st 2018.

2018 IEEE 9th Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON)

with either the goal of creation inconsistent blockchain views for different network participants, or just causing straightforward disruption on the blockchain network's operations, and 3) attacks on *smart contracts*, in which case the security of the programs themselves are subverted.

Originally the integrity guarantees of a blockchain were believed to be preserved as long as the majority of nodes were honest, with any attack requiring a majority of the network nodes[2] to be under attackers control, thus giving the name "51% attack" for these types of threats [4]. Later research by Eyal and Sirer showed that for Bitcoin, the security of the blockchain can be compromised if more than ⅓ of the miners are colluding [5], thus significantly reducing the potential cost of such an attack. Further research has looked into optimal selfish mining strategies, and its efficiency with various network and latency assumptions [6], [7].

Attacks against the blockchain infrastructure can take many forms. In Bitcoin, several types of attacks causing delays in transactions processing have been shown to be feasible. Gervais et al. discussed a mechanism where an attacker can manipulate the visibility of Bitcoin blocks on a victim, and described how this could be extended to a denial-of-service attack on the whole Bitcoin network [8]. Another type of an attack on the Bitcoin mining infrastructure is to hijack the Internet routing protocol and cause an increase in forks [9]. Natoli and Gramoli discuss a mechanism where the introduction of messaging delays into the Bitcoin network could create disjoint groups with similar mining power [10], allowing the attacker to manipulate the final chain selection to his or her advantage (e.g. to double-spend).

It is also possible to use smart contracts as a conduit of attacking the blockchain infrastructure. In Ethereum's early history, for example, an instruction executable by a smart contract was underpriced compared to the real-world computing resources required to execute it [11]. Finally, smart contracts themselves are susceptible to security failures [12] that can lead to monetary loss or prevent a particular smart contract from operating correctly [13].

While an attack may attempt to either break or disable a blockchain, sometimes even a minor attack can decrease the usefulness of the blockchain to its users. Weber et al. [14] analyzed the effect of an attack on the Ethereum blockchain on network user's transaction processing delays and costs. Even when an attack does not misuse the blockchain network, it is possible to cause service disruption by submitting otherwise harmless transactions. A transaction spamming attack against Bitcoin in 2015 was analyzed by Baqer et al. [15], who found that in this case, 23% of all Bitcoin transactions were associated with the attack during its 10-day run. They estimated that for other network users the transaction costs increased over 50%, and transaction processing times increased to almost three hours (compared to the typical delay of 20 minutes before the attack started). Finally, they estimated that the transaction spamming attack cost the attacker 201 BTC (approximately $49,000 at the time of the attack).

The possibility of a deluge of transactions causing disruptions can be witnessed through the sudden popularity of a a social game called CryptoKitties. Its surging popularity had a similar effect on Ethereum transactions as the previous spamming attack had on Bitcoin — the increased demand for trades in CryptoKitties, which at one point exceeded 15% of all transactions in Ethereum, led to a significant increase in transaction costs and transaction processing delays [16]. Eventually the increase in transactions fees and growth of block gas limit normalized the situation, showing the long-term adaptability of the network to a change, while demonstrating the difficulty of adjusting to rapid usage pattern changes in the short term.

In this paper we analyze transaction history of the Ethereum blockchain, and look at the cost and effectiveness of a spam attack as described by Baqer et al., e.g. one where the attacker does not attack the infrastructure per se, but attempts to influence the service quality and usage costs of the network. While this type of attack is known by the general Ethereum community, we have found that it has not been generally characterized in the scientific literature — there are even assertions on the financial impossibility of this attack such as a statement of "[Ethereum's] execution fees also protect against denial-of-service attacks" [12]. We show that performing a spam attack on the Ethereum network is not only feasible, but can be executed at a limited financial cost. Even if an attacker does not aim to disrupt the network in general, the use of limited time windows in smart contracts may open attack venues where introducing a delay to a transaction through transaction spamming may cause the transaction to miss the limited time window.

The rest of this paper is structured as follows. In Section II, we provide further background on the Ethereum blockchain and how its economic and operational models work. In Section III, we describe the transaction spam attack model. In Section IV, we present the supporting results from our analysis of the historical Ethereum transaction data, and show the efficacy and costs associated with the transaction spam attack. Then, in Section V we present our final conclusions, and finally, in Section VI, we discuss some of the shortcomings of this paper and present ideas for future work.

II. Ethereum blockchain

Ethereum is a distributed network of distrusting nodes that are monetarily incentivized to provide a trustworthy decentralized computing environment [17]. The integrity of the Ethereum blockchain is guarded by a robust distributed consensus algorithm, currently the proof-of-work Nakamoto consensus. Ethereum supports *smart contracts*, pieces of program code stored on the blockchain that are able to execute arbitrary state-changing computations. Operationally, the nodes in the Ethereum network are incentivized through the mining and transaction processing rewards that are provided in Ethereum network's native cryptocurrency, the *ether* (one

[2]To be precise, what is needed for the attack is a majority of the puzzle-solving capacity of the network, but for simplicity, we refer to nodes instead.

ether is further subdivided into units called *wei* with one ether being equal to 10^{18} wei). The cryptocurrency can be acquired through mining, or purchased for real-world money at specialized exchange services.

The global state of the blockchain is manipulated through *transactions*, which upon their execution are grouped into *blocks*. To prevent Sybil attacks, a new block is accepted by the network nodes only if contains a valid proof of work. Thus, for *miners* to successfully generate a new block that is accepted by the network, the miner must engage in the process of *mining* where they attempt to solve the cryptographic puzzle that acts as the required proof of work. The miner is rewarded for creating a new block in the form of a *block reward*, and they are also able to collect *transaction fees* of any transactions they processed and included in the block.

Each block in Ethereum can contain only a limited number of transactions. The number of transactions that fit into a block is controlled by the *block gas limit*. The block gas limit is adjusted by miners via a decentralized mechanism, where the miner can slightly increase or decrease the block gas limit. The use of a block gas limit allows Ethereum to adapt to the transaction demand while keeping the number of transactions per block bounded.

Transactions in Ethereum are executed under a formally defined computational model called the Ethereum Virtual Machine (EVM)[3]. The EVM executes smart contracts in a deterministic manner, thus allowing all nodes in the network to verify the correct execution of any smart contract. The EVM model is Turing-complete, and thus cannot provide any generic execution time bound guarantees. To provide bounded execution time a concept of *gas* is used. Each EVM operation consumes a specific and deterministic amount of gas. The initiating transaction must supply the gas that the execution requires — if gas runs out, the execution is terminated. Conversely, any unused gas is deposited back to the transaction initiator. The unit of gas used by transactions is arbitrary and expressed as an integer number, and is simply called "gas".

The initiator of a transaction controls transaction processing fees through two transaction parameters, the *transaction gas limit* and the *gas price*. The gas limit defines the maximum amount of gas the transaction may consume, and the gas price specifies the amount of cryptocurrency per unit of gas that the initiator is willing to pay for. A typical gas price is in the order of few to hundreds of billions of wei per gas (e.g. nanoethers). The combination of the transaction gas limit and the gas price allows the initiator to have an upper bound on the transaction process fee. Finally, a transaction contains other parameters such as *value*, which is the amount of cryptocurrency being sent as part of the transaction.

The transaction processing fee forms the incentive for miners to use their resources for processing other parties' transactions. When no other relation exists between the miner and transaction sender, an economically rational miner should choose to process transactions expected to produce largest rewards for the miner. Some miners specify a minimum gas price to avoid running transactions that do not cover their real-world costs such as electricity and capital expenses. Consequently, all submitted transactions form a market in the Ethereum network where each transaction is competing with other transactions to be included in the next mined block. An initiator of a transaction can analyze the current gas price market, and decide on a suitable gas price they believe will result in miners including the transaction in a future block. Consequently the transaction processing market is dynamic and there is no predetermined or fixed price for one unit of gas.

In the Ethereum network, individual transactions are propagated through the network via a peer-to-peer mechanism. If a transaction is initially submitted to a sufficient number of independent nodes, the transaction will propagate through the Ethereum network, and has a very high likelihood of being seen by a node which successfully mines a block [14]. Miners will typically hold a large collection of pending transactions, effectively buffering transactions until they can be fitted into a block. The combination of dynamic peer-to-peer propagation, the connectivity of nodes where the transaction is submitted and the depth and behavior of miners' transaction pool behavior means that there is never a single, globally consistent state of pending transactions on the whole network.

III. Transaction spamming

The underlying assumption of a decentralized blockchain is that all miners are *economically rational* — they attempt to selfishly maximize their own profits. The mining incentives are designed to align the interests of miners with the global utility, thus providing positive value to all participants on the blockchain.

As described in the work of Luu et al. [18], transaction processing and verification in Ethereum can be expensive. While non-mining nodes may decide to skip transaction validation and settle for lesser integrity guarantees, the full cost of transaction processing must be carried by miners. Consequently, a miner must balance the real-world costs associated with transaction processing with fees gained from processing them. Thus, a rational miner, when presented with multiple transactions at different gas price levels, can be assumed to preferentially select those transactions which offer a higher transaction processing fee[4]. The effectiveness of a spam attack is based on this selfish rationality of the miners.

As a consequence, **we envision that an attacker with sufficient resources, is able to preferentially get their transactions included in future blocks by paying a higher transaction processing fee.** This is similar to the Bitcoin transaction spam attack described by Baqer et al. [15]. The effect of spamming the transaction queue is that transactions

[3]Technically a transfer of cryptocurrency from an account to another does not run an EVM computation, however these can be seen as implicit degenerate smart contracts, and in this manner they fit within the computational model.

[4]This has been verified to be the operating mode in popular Ethereum node software such as Parity and Geth, e.g. when mining, transactions with a higher gas price are preferred over transactions with a lower gas price.

at a lower gas price are not included in the next block. By continuously generating spam transactions, the attacker is able to indefinitely prevent any lower-value transactions from completing (or until attacker's resources, e.g. cryptocurrency, is exhausted). While conceptually simple, in reality the effectiveness of an attack is limited due to four reasons: 1) the transaction queue processing has asynchronous latencies which introduce inherent randomness, 2) miners often prioritize their own transactions, thus transactions from miners themselves are difficult to block, and 3) block gas limit is dynamic and miners may choose to increase it during an attack, thus increasing the cost of the attack, and finally 4) gas price itself is controlled by transaction initiators, and they are likely to react to changes in the gas price market. Each of these factors is discussed in detail below.

While it is reasonable to assume that an attacker would actively seek out nodes with maximum transaction distribution capabilities for transaction queue spamming, the dynamic structure of the Ethereum network makes it possible that a successful miner will see only a small subset of the attacker's transactions, thus allowing transactions with gas price lower than the attacker's threshold value to be included into the mined block. Therefore, it is not possible to guarantee that *all* lower-value transactions are blocked even if an overwhelming amount of spam transactions are created. Regardless, we hypothesize that a *majority* of lower-value transactions can be blocked, especially in a situation where the attack is sustained for a long period of time, allowing sufficient number of spam transactions to reach all nodes in the network.

While miners are assumed to be economically rational, e.g. maximizing their own rewards, miners also have the incentive of *not* paying for transactions they themselves have initiated. While successfully mining a block is highly unlikely for a single machine, a collection of co-operating miners called a *mining pool,* can have a sufficiently large likelihood of successfully mining a block within a limited amount of time. Consequently, if transactions from within the mining pool are prioritized, they will be included in the blockchain even when priced at a lower level than the spam attack's gas price threshold. Thus, a spam attack is unable to prevent low-value transactions from miners and mining pools that prioritize transactions originating from participants in the mining pool.

The Ethereum protocol allows miners to modify the block gas limit in a small relative increment or decrement. Miners seeing a full transaction queue can increase the block gas limit, allowing them to include more transactions in the mined block and to collect a higher transaction processing reward. This would also increase the number of spam transactions included in the block, thus driving the attack cost up. The block gas limit is dynamic, albeit its growth rate is limited to about 26% increase over approximately an hour (240 blocks). It is unlikely this cost increase would have an effect on a short-term attack, but over a longer time span, it can increase the attack costs prohibitively.

The transaction initiator is able to observe the current Ethereum transaction market and select a gas price that meets their transaction latency requirements. It seems likely that during an attack, users of the network would either abstain from creating new transactions, or would increase the gas price of the submitted transactions to compensate. Thus, the market forces of supply and demand would drive up the median gas price of the new transactions. To block a certain portion of all transactions the attacker would need to increase the gas price of spam transactions, which in turn, would drive the attack cost up.

In summary, while in practice a spam attack blocking 100% of other parties' transactions in not feasible, and a long attack would progressively see increasing costs, there is nothing that would prevent a short-term attack from blocking the execution of a large number of transactions. The actual cost and effectiveness of such an attack is analyzed in the next section.

IV. Analysis

A. Methods

Ethereum blockchain data was collected by running the Parity program in archival mode. In this mode, the program will collect and store all blocks, transactions and the historical blockchain state. After the archival node had successfully performed a full synchronization of all of the blockchain history, the data was queried over the Ethereum standard RPC-JSON API and stored into a suitable format for later loading into a Postgresql database, from which it was further narrowed down and exported to the statistical analysis program R for final analysis and generation of figures.

B. Historical block characteristics

The initial analysis range was from the genesis block (block 0, July 30 2015) until block 5,325,329 (March 26 2018). There is a distinct development pattern during the history of Ethereum showing that number of transactions per block, block gas limit and block gas usage have increased substantially over the the last few million blocks, as shown in Figure 1. As shown in the figure, since about block 4,000,000 an increasing number of blocks have no space for transactions, e.g. blocks are full. The graph also clearly shows the remediation effort's effects of a denial-of-service attack against the network at around block 2,400,000 on the block gas limit (the attack is described in [11]).

C. Transactions

As described above, the usage patterns of the Ethereum network have changed substantially over time. For this reason we limit further analysis to only recent history between blocks 3,000,000 up to block 5,299,999 for a total of 2.3 million blocks. These blocks span the time interval from from January 15th 2017 to March 22nd 2018. This range contained a total of 172,465,920 transactions with the gas price ranging from 0 wei/gas to 11.9×10^{15} wei/gas, with the mean gas price of 29.8×10^9 wei/gas and median of 21.0×10^9 wei/gas. The 99th percentile of transaction gas price is 129×10^9 wei/gas. The data also shows that transaction gas prices are highly stratified,

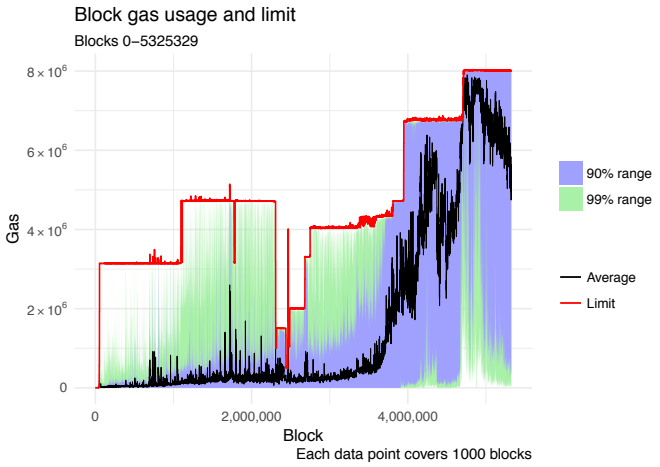

Figure 1: Gas usage and block gas limit by block, showing the average, 90th and 99th percentiles of block gas usage of blocks in increments of a thousand blocks, and the maximum of the gas limit over the same ranges.

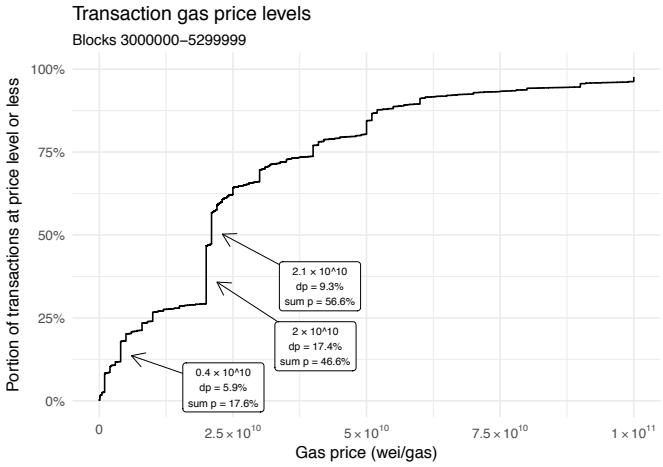

Figure 2: Relative portion of transactions with gas price at or less than a given level. Several of the highly stratified prices are highlighted in the graph, representing distinct gas prices that are used unusually frequently in transactions.

with 83.1% of all transactions having the gas price of an integer multiple of 10^9. The cumulative gas price distribution is shown in Figure 2.

D. Costs of a transaction spam attack

For further calculations the block gas limit was assumed to be 8×10^6 gas/block[5]. This cost model is based only on the distribution of transaction gas prices as shown in Figure 2, and the portion of blocked transactions is simply the percentage of transactions below a specific threshold gas price. The blocking ratios compared to attack costs are shown in Figure 3.

[5]The average block gas limit has fluctuated consistently around 8 million gas per block during early 2018.

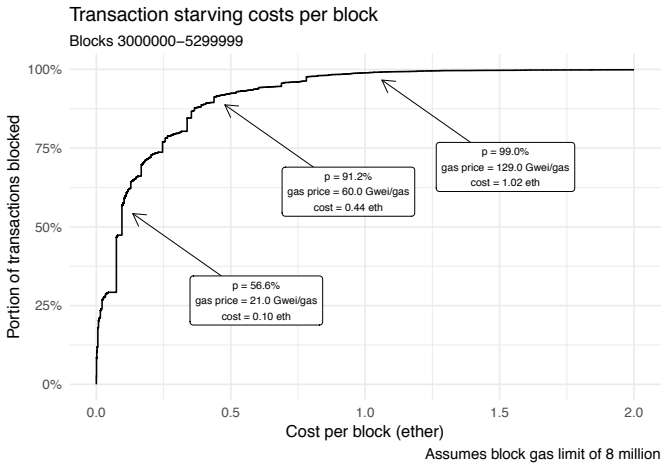

Figure 3: The effect of cost per block of the transaction spam attack on the portion of transactions blocked. The cost is based on block gas limit of 8×10^6 gas per block. Some examples of attack cost versus the portion of blocked transactions p are highlighted.

The costs and effectiveness of a spam attack at several gas price levels is shown in Table I. The table provides costs in ethers per hour for assumptions of a static block gas limit, and a dynamic block gas limit that increases with the maximum rate after the onset of the attack. The maximum block gas limit increase rate is 1 unit in 1024 per block which equates to the attacker needing to pay an increase of 12.6% over the unchanging block gas limit scenario. The block gas limit itself would have increased to over 10 million gas per block after one hour[6]. The attack costs in Table I are in Ethereum's own cryptocurrency unit. The conversion rate between ethers and dollars fluctuates greatly — according to ethereumprice.org the peak ether cost has been $11,422.47 on January 14th 2018, and the lowest value after the peak until March 22nd was $454.80. If we consider these as lower and upper costs of ether for the spam attack at 95% effectiveness, its execution for one hour would have cost between $78 590 and $276 800.

V. Conclusions

We have analyzed the potential costs and effectiveness of mounting a spam attack on the Ethereum network based on the distribution of gas prices of historical transactions. The ability of an adversary to delay transactions may severely affect the real-world security assurances that an IoT system using the public Ethereum network can provide.

This type of a denial of service attack itself is generally known by the Ethereum community, and has also been described earlier in the context of Bitcoin by Baqer et al. We

[6]The block period in Ethereum is approximately 15 seconds, thus about 240 blocks are mined in an hour. The sum of the geometric series $\sum_{k=0}^{n-1} ar^k = a\left(\frac{1-r^n}{1-r}\right)$ for $n = 240$, $a = 1$ and $r = {}^{1025}/_{1024}$ is 270.3, a relative increase over the baseline of (270.3–240)/240 = 12.6%. The last 240th block would have a block gas limit increased from the initial value by $\left(\frac{1025}{1024}\right)^{240} - 1$ = 26%.

Coverage	Gas price	Costs		
		Static	Static	Dynamic
	wei	ether/block	ether/h	
50.0 %	21 000 000 000	0.2	40.3	45.4
90.0 %	60 000 000 000	0.5	115.2	129.7
95.0 %	90 000 000 000	0.7	172.8	194.6
99.0 %	129 000 000 000	1.0	247.7	279.0
99.9 %	280 609 467 870	2.2	538.8	606.8

Table I: Cost of a transaction spam attack based on the distribution of transaction gas prices in the analysis range. The costs are based on an assumption of an initial block gas limit of 8×10^6 gas per block.

would like to point out that the spam attack is a denial of service type of attack that operates *within* the economic model of the blockchain. It does not try to attack the integrity of the blockchain, and neither does it try to prevent the Ethereum nodes from functioning or from processing transactions. From the viewpoint of the mining nodes there is no attack occurring — all of the transactions initiated by the attacker *are normal transactions*. The miners are incentivized to prefer the attacker's transactions by the use of the elevated gas price. The attack thus is an attack against Ethereum *transactions*, not against the infrastructure, the blockchain, or smart contracts.

This poses a problem for IoT devices that use the Ethereum blockchain for control or management information, or use it to transmit information. While it is possible to characterize "normal" blockchain latency and transaction price variation, and define a gas price strategy to provide a high guarantee of timely transaction execution, such strategies are unlikely to work reliably against a purposeful spam attack. Thus, from IoT-DLT integration point of view, even a blockchain that provides security guarantees of integrity and non-repudiation may still prove to be manipulatable in ways that can be used to break security or safety models in the real world.

While this type of an attack is unlikely to be economically within the reach of an individual, we have shown that a time-limited attack is within the reach of well-resourced organizations such as large corporations, criminals and nation-states. We believe that given a sufficiently lucrative target, this type of attack is economically feasible and profitable to implement. While the cost of an attack is likely to increase during a prolonged spam attack, the changes are likely to be initially small and lagged, leaving a window of opportunity for such an attack to be successful.

VI. Further work

The cost estimation model in this paper assumes that all transactions are random and memoryless samples from the existing distribution. In reality, transaction gas prices vary throughout the day and month, exhibiting delayed reactions to sudden changes in transaction demand and gas prices. For example, instead of making a static assumption on the gas price distribution, the existing distribution could be taken as a repository of transaction traces against which the effectiveness of the attack could be compared against. Similarly, an analytical model of the market behavior to gas price changes would be an interesting proposition.

Looking from an attacker's point of view, an interesting question is the feasibility of an adaptive attack, where the attacker is assumed to be able to inspect the transaction queue and adapt the gas price of spam transactions accordingly. Here, the attack could be even tailored to attack transactions of a specific account, potentially having a significantly lower cost of the attack. An another possibility is to use of an eclipse attack to trick targeted parties into a network partition where a critical transaction is delayed, yet, when the eclipsed portion of the blockchain network joins the main network, all attack costs are voided since the eclipsed portion and all its transactions are "lost".

While not described here in detail, we performed some ad hoc analysis of the blockchain's history and found several sequences of blocks with transactions that appear to have no other purpose than to take all the transaction space. However, as hiding the source and purpose of a spam transaction is easy, a statistical clustering approach such as taken by Baqer et al. would be necessary to reliably distinguish between regular and spam transactions.

Acknowledgment

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 779984. We would also like to thank Jenni Huttunen for her feedback and comments on this paper.

References

[1] N. Tapas, G. Merlino, and F. Longo, "Blockchain-Based IoT-Cloud Authorization and Delegation," in *2018 IEEE International Conference on Smart Computing (SMARTCOMP)*, 2018-06, pp. 411–416. DOI: 10.1109/SMARTCOMP.2018.00038.

[2] O. Novo, "Blockchain Meets IoT: An Architecture for Scalable Access Management in IoT," *IEEE Internet of Things Journal*, vol. 5, no. 2, pp. 1184–1195, 2018-04, ISSN: 2327-4662. DOI: 10.1109/JIOT.2018.2812239.

[3] J. R. Douceur, "The Sybil Attack", in *Peer-to-Peer Systems*, ser. Lecture Notes in Computer Science, Springer, Berlin, Heidelberg, 2002-03-07, pp. 251–260, ISBN: 978-3-540-44179-3. DOI: 10.1007/3-540-45748-8_24.

[4] S. Nakamoto, *Bitcoin: A peer-to-peer electronic cash system*, 2008. [Online]. Available: https://bitcoin.org/bitcoin.pdf.

[5] I. Eyal and E. G. Sirer, "Majority Is Not Enough: Bitcoin Mining Is Vulnerable", in *Financial Cryptography and Data Security*, ser. Lecture Notes in Computer Science, Springer, Berlin, Heidelberg, 2014-03-03, pp. 436–454, ISBN: 978-3-662-45471-8. DOI: 10.1007/978-3-662-45472-5_28.

[6] K. Nayak, S. Kumar, A. Miller, and E. Shi, "Stubborn Mining: Generalizing Selfish Mining and Combining with an Eclipse Attack," in *2016 IEEE European Symposium on Security and Privacy (EuroS P)*, 2016-03, pp. 305–320. DOI: 10.1109/EuroSP.2016.32.

[7] J. Göbel, H. P. Keeler, A. E. Krzesinski, and P. G. Taylor, "Bitcoin blockchain dynamics: The selfish-mine strategy in the presence of propagation delay," *Performance Evaluation*, vol. 104, pp. 23–41, 2016-10-01, ISSN: 0166-5316. DOI: 10.1016/j.peva.2016.07.001.

[8] A. Gervais, H. Ritzdorf, G. O. Karame, and S. Capkun, "Tampering with the Delivery of Blocks and Transactions in Bitcoin," in *Proceedings of the 22Nd ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '15, New York, NY, USA: ACM, 2015, pp. 692–705, ISBN: 978-1-4503-3832-5. DOI: 10.1145/2810103.2813655.

[9] M. Apostolaki, A. Zohar, and L. Vanbever, "Hijacking Bitcoin: Routing Attacks on Cryptocurrencies," in *2017 IEEE Symposium on Security and Privacy (SP)*, 2017-05, pp. 375–392. DOI: 10.1109/SP.2017.29.

[10] C. Natoli and V. Gramoli, "The Balance Attack or Why Forkable Blockchains are Ill-Suited for Consortium," in *2017 47th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*, 2017-06, pp. 579–590. DOI: 10.1109/DSN.2017.44.

[11] J. Wilcke. (2016-09-22). The Ethereum network is currently undergoing a DoS attack, [Online]. Available: https : / / blog . ethereum . org / 2016 / 09 / 22 / ethereum - network - currently - undergoing - dos - attack/ (visited on 2018-03-28).

[12] N. Atzei, M. Bartoletti, and T. Cimoli, "A Survey of Attacks on Ethereum Smart Contracts (SoK)", in *Principles of Security and Trust*, ser. Lecture Notes in Computer Science, Springer, Berlin, Heidelberg, 2017-04-24, pp. 164–186, ISBN: 978-3-662-54454-9. DOI: 10.1007/978-3-662-54455-6_8.

[13] L. Kiffer, D. Levin, and A. Mislove, "Stick a Fork in It: Analyzing the Ethereum Network Partition," in *Proceedings of the 16th ACM Workshop on Hot Topics in Networks*, ser. HotNets-XVI, New York, NY, USA: ACM, 2017, pp. 94–100, ISBN: 978-1-4503-5569-8. DOI: 10.1145/3152434.3152449.

[14] I. Weber, V. Gramoli, A. Ponomarev, M. Staples, R. Holz, A. B. Tran, and P. Rimba, "On Availability for Blockchain-Based Systems," in *2017 IEEE 36th Symposium on Reliable Distributed Systems (SRDS)*, 2017-09, pp. 64–73. DOI: 10.1109/SRDS.2017.15.

[15] K. Baqer, D. Y. Huang, D. McCoy, and N. Weaver, "Stressing Out: Bitcoin "Stress Testing"", in *Financial Cryptography and Data Security*, ser. Lecture Notes in Computer Science, Springer, Berlin, Heidelberg, 2016-02-26, pp. 3–18, ISBN: 978-3-662-53356-7. DOI: 10.1007/978-3-662-53357-4_1.

[16] M. Hrones. (2017-12-05). CryptoKitties Creates Massive Backlog on the Ethereum Network, [Online]. Available: http : / / bitcoinist . com / cryptokitties - creates - massive - backlog - on - the - ethereum - network/ (visited on 2018-03-28).

[17] G. Wood, "Ethereum: A secure decentralised generalised transaction ledger," *Ethereum Project Yellow Paper*, vol. 151, 2014.

[18] L. Luu, J. Teutsch, R. Kulkarni, and P. Saxena, "Demystifying Incentives in the Consensus Computer," in *Proceedings of the 22Nd ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '15, New York, NY, USA: ACM, 2015, pp. 706–719, ISBN: 978-1-4503-3832-5. DOI: 10.1145/2810103.2813659.