Upper Bound Probability of Double Spend Attack on SPECTRE

Lyudmila Kovalchuk
Input Output HK,
National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute"
Kyiv, Ukraine
lyudmila.kovalchuk@iohk.io

Mariia Rodinko
Input Output HK,
V. N. Karazin Kharkiv National University
Kharkiv, Ukraine
mariia.rodinko@iohk.io

Roman Oliynykov
Input Output HK,
V. N. Karazin Kharkiv National University
Kharkiv, Ukraine
roman.oliynykov@iohk.io

ABSTRACT

We obtained analytical expressions of upper estimates for success probability of a double spend attack on DAG-based consensus protocol SPECTRE, depending on network parameters. Using such estimates, it is possible to evaluate the number of confirmation blocks that is sufficient for prevention of such attacks.

CCS CONCEPTS

• **Security and privacy** → **Distributed systems security**.

KEYWORDS

blockchain, directed acyclic graph, proof-of-work, SPECTRE, double spend attack

ACM Reference Format:
Lyudmila Kovalchuk, Mariia Rodinko, and Roman Oliynykov. 2020. Upper Bound Probability of Double Spend Attack on SPECTRE. In *CryBlock '20: 3rd Workshop on Cryptocurrencies and Blockchains for Distributed Systems, September 25th, 2020, Virtual (online) Conference.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

The new generation of consensus protocols are based on the directed acyclic graph (DAG) structure. Such protocols are SPECTRE [5], PHANTOM [6], Graphchain [1], Tangle [4] and other proposals, where blocks or transactions form a DAG as a distributed ledger. The main objective is to maximally utilize the network capacity having strong security guarantees. Some characteristics of the Nakamoto consensus were really improved by these protocols, resulting in higher throughput and faster transaction confirmation, but not providing properties, like linear block ordering, in full.

In this paper, we will consider SPECTRE protocol [5]. It is one of the earliest and fastest DAG protocols with rather unusual consensus algorithm. In SPECTRE, a newly created block is published immediately, and it refers to all recent blocks in DAG. A protocol provides a pairwise voting algorithm that is carried out by all blocks. The value of a vote is defined by the block position in DAG. SPECTRE provides fast confirmation of blocks and good scalability, but only weak liveness. The authors state that this protocol is secure against different attacks, such as double spend and censorship, but give no proofs for this statement, only some empirical considerations.

In what follows, we propose some hybrid of censorship and double spend attacks. During this attack malicious miners not only generate alternative chains or subDAGs, but also choose which blocks to confirm. Such an attack is more effective than both mentioned attacks separately. We also show that this attack can be resisted (under certain restrictions on the fraction of an adversary's computational power) if the rules of transactions' acceptance by vendor are changed a little. We also present rigourously substantiated numerical estimates of probability of such attack depending on the network parameters and the number of the confirmation blocks.

2 SPECTRE DESCRIPTION

SPECTRE [5] is a Proof-of-Work protocol where blocks are ordered in a direct acyclic graph (DAG) $G = (C, E)$, where C are blocks, E are hash references. The rules for miners are simple [1]:

(1) When creating or receiving a block, transmit the block to all peers.
(2) When creating a block, embed in its header a list containing the hash of all leaf-blocks (blocks with in-degree 0) in the locally-observed DAG.

To avoid conflicting transactions, a voting procedure is used. Every block $z \in G$ is considered a voter. To describe the voting rules, some definitions should be introduced [5]:

- $past(z, G) \subset C$ denotes the subset of blocks reachable from z;
- $future(z, G) \subset C$ denotes the subset of blocks from which z is reachable;
- $cone(z, G)$ denotes the set of blocks that the DAG directly orders with respect to z;
- $virtual(G)$ denotes a hypothetical block that satisfies $past(virtual(G)) = G$.

If there are two blocks x, y that have conflicting transactions, voting should be used to decide $x \prec y$ (vote -1) or $y \prec x$ (vote +1) (if there is a tie, then a vote is 0). The voting rules are as follows [5]:

(1) if $z \in G$ is in $future(x)$ but not in $future(y)$ then it will vote in favour of x (i.e., for $x \prec y$).
(2) if $z \in G$ is in $future(x) \cap future(y)$ then z's vote will be determined recursively according to the DAG that is reduced

to its past, i.e., it has the same vote as $virtual(past(z))$. If the result of this vote is a tie, z breaks it arbitrarily.

(3) if $z \in G$ is not in the future of either blocks then it will vote the same way as the vote of the majority of blocks in its own future.

(4) if z is the virtual block of G then it will vote the same way as the vote of the majority of blocks in G.

(5) finally, (for the case where z equals x or y), z votes for itself to succeed any block in $past(z)$ and to precede any block outside $past(z)$.

There are two types of attacks analyzed in the original paper: double-spending and censorship [5]. In a double-spending attack, the first mining rule is broken (an attacker builds blocks in secret and publishes two chains with conflicting transactions together). In a censorship attack, an attacker breaks the second rule and does not build on all fresh blocks, he mines only on those blocks that are included into the tree which contains an alternative transaction). We present below a hybrid attack, where two rules are broken by an attacker at the same time.

3 DESCRIPTION OF THE ATTACK

All stages of this attack are given on Figure 1 where:

- T_1 is the moment, when malicious miners generated CM_1 and start to generate CM_2;
- T_2 is the moment, when malicious miners publish CM_1;
- T_3 is the moment, when malicious miners receive goods and publish $CM_2 \cup CM3$.

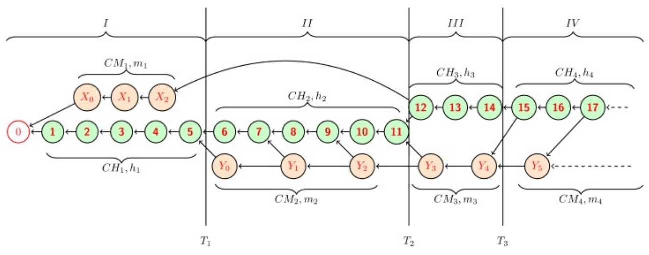

Figure 1: An example of the hybrid attack

To simplify Figure 1, we picture a definite number of blocks on it. But in the proof, we will refer to an arbitrary number of blocks in each chain. Also we need to note that when we replace any chain by a "tree" with the same number of blocks, nothing changes in the model of attack and in the proof of the theorem.

Stage I.

- Honest miners generate the chain CH_1 of the length h_1.
- Malicious miners generate a block with transaction X_0 and confirm this block, i.e. generate the chain CM_1 of the length m_1 (but do not open this chain).

Stage II.

Starts from the moment T_1, when malicious miners decide that they generate enough confirmation blocks for a transaction X_0.

- Malicious miners generate a block with a transaction Y_0 and confirm this block, i.e. generate the chain CM_2 of the length m_2 (but do not open this chain).
- Honest miners generate the chain CH_2 of the length h_2, which continues the chain CH_1.

Stage III.

Starts from the moment T_2, when malicious miners decide that they generate enough confirmation blocks for transaction Y_0.

- Malicious miners open the chain CM_1.
- Vendor waits for some time, and meanwhile honest miners generate the chain CH_3 of the length $h_3 \geq z$ for some appropriate z, and then sends goods or services.
- Meanwhile malicious miners generate the chain CM_3 of the length m_3 that continues to confirm the block with X_0. Malicious miners still do not open the chains CM_2 and CM_3.

Stage IV.

Starts from the moment T_3, when the vendor sends goods or services.

- Vendor sends the goods at the moment T_3.
- Immediately after that, malicious miners open two chains CM_2 and CM_3.
- Honest miners confirm both chains, CH_3 and CM_3, with the chain CH_4 of the length h_4.
- Malicious miners confirm only the chain CM_3 with the chain CH_4 of the length m_4 (combining double spend and censorship attacks).

Under these conditions and according to the voting protocol of SPECTRE, the voting procedure will be as follows:

- all the blocks of of CM_1 vote for X_0 (m_1 blocks);
- all the blocks of CH_3 vote for X_0 ($h_3 \geq z$ blocks);
- all the blocks of CM_2, CM_3, CM_4 vote for Y_0 ($m_2 + m_3 + m_4$ blocks);
- all the blocks of CH_1 and CH_2 vote as the majority in their futures ($h_1 + h_2$ blocks);
- all the blocks of CH_4 vote as the majority in their pasts (h_4 blocks).

It should be noted that the vendor's behavior in this case should be different from the standard one, namely: he should wait for no less than z blocks after the moment he sees the transaction X_0 for the first time, even if at the time of its first appearance the block containing this transaction has no less than z descendants. This requirement will not allow an adversary to attack with the probability close to 1.

Now we are going to prove the theorem on the upper bound of the success probability of a double spend attack. In what follows, we need some designations and notations from [2, 3].

Let $\alpha_H, \alpha_M > 0$ be some values that characterize the intensity of block generation for honest miners and malicious miners, respectively, and $\alpha = \alpha_H + \alpha_M$ is the general intensity. We define these values as the parameters of distribution functions:

$$\begin{aligned} F_{T_H} &= P(T_H < t) = 1 - e^{-\alpha_H t}, \\ F_{T_M} &= P(T_M < t) = 1 - e^{-\alpha_M t}, \end{aligned} \tag{1}$$

where T_H, T_M are random variables that measure the time it takes to mine a block for honest miners and malicious miners, respectively.

Then the probability that honest miners generate the next block before malicious miners is

$$p_H = \frac{\alpha_H}{\alpha_M + \alpha_H}, p_M = 1 - p_H = \frac{\alpha_M}{\alpha_M + \alpha_H}. \tag{2}$$

We also assume that D_H denotes the time it takes for honest miners to share a block (after it was generated) for all (at least all honest) nodes in network; D_M is the respective time for malicious miners, and here we assume $D_M = 0$, i.e. an adversary is well-synchronized.

Also let us define:

p'_H is the probability of the event that the honest miners generate and share the next block for all (at least honest) nodes before malicious miners do this;

$p'_M = 1 - p'_H$ is the probability of the alternative event.

Then, according to [3], Lemma 1,

$$\begin{aligned} p'_H &= e^{-\alpha_M D_H} p_H \\ p'_M &= 1 - e^{-\alpha_M D_H} p_H. \end{aligned} \tag{3}$$

Designate as $P_z(k)$ the probability that malicious miners generate exactly k blocks until honest miners generate and share z blocks. Then, according to Lemma 4 in [3],

$$\begin{aligned} P_z(k) = \sum_{i=0}^{k} \left[\binom{z+i-1}{i} p_H^z p_M^k e^{-\alpha_M n D_H} \right] \times \\ \times \frac{(\alpha_M n D_H p_H)^{k-i}}{(k-i)!}. \end{aligned} \tag{4}$$

Note that in the case when $D_H = 0$, the probability (4) has a much simpler expression:

$$P_z(k) = \binom{z+k-1}{k} p_H^z p_M^k \tag{5}$$

as in the sum (4) only one summand is left: for $k = i$; all others vanish because of $(\alpha_M n D_H)^{k-i}$.

Now we are ready to formulate the main result.

Theorem 3.1. *Let the vendor wait for z blocks after he saw for the first time the transaction X_0. Then, in the notations (1)-(5), the upper bound of the probability $P_z(\alpha_M, \alpha_H, D_H)$ of the success by the MM is:*

$$\begin{aligned} & P_z(\alpha_M, \alpha_H, D_H) \le \\ & \le 1 - \sum_{k=0}^{z-1} P_z(k) \cdot \left(1 - \left(\frac{p'_M}{p'_H}\right)^{z-k} \left(\left(p'_H\right)^{z-k-1} + 1\right)\right) \end{aligned} \tag{6}$$

and in the particular case when $D_H = 0$

$$\begin{aligned} & P_z(\alpha_M, \alpha_H, 0) \le \\ & \le 1 - \sum_{k=0}^{z-1} p_H^z p_M^k \binom{z+k-1}{k} \times \\ & \times \left(1 - \left(\frac{p_M}{p_H}\right)^{z-k} \left(\left(p_H\right)^{z-k-1} + 1\right)\right). \end{aligned} \tag{7}$$

If $p'_M \ge p'_H$, then $P_z(\alpha_M, \alpha_H, D_H) = 1$.

Proof. Let B_1 be the first block that sees X_0 and Y_0 in its past (in Figure 1 B_1 is the block 15). This block and all the blocks in its future will vote as the majority in their pasts. So, the votes of the blocks from CH_4 essentially depend on the votes of the blocks in the past of (B_1).

Let us analyze votes in the $past(B_1)$:

- all blocks from CH_3 vote for X_0;
- all blocks from CM_3 vote for Y_0;
- the last block of CH_2, B_2 (in Figure 1 B_2 is the block 11), votes for Y_0 if $h_3 \le m_3$, and so will all blocks from CH_1 and CH_2. Therefore one of conditions of a successful attack is $h_3 \le m_3$ that we designate as the condition $C1$:

$$h_3 \le m_3.$$

When $(C1)$ does not hold, malicious miners still may have success in the case if blocks in the $past(B_2)$ vote for Y_0. Let us analyze when it is possible, if $C1$ does not hold, then B_2 votes for X_0. Let $h_3 = m_3 + l$. Then the block that is just before B_2, named B_3 (in Figure 1 this is 10), will vote for Y_0 only if during the time between B_3 and B_2, malicious miners generate not less than $l + 1$ blocks. Similarly, if B_3 and B_2 vote for X_0, the necessary condition for the previous block B_4 to vote for Y_0 is that during the time between B_4 and B_3 malicious miners generate not less than $l + 2$ blocks, and so on.

So, if $C1$ does not hold, the necessary condition for the event "some block number i in the chain CH_2 votes for Y_0" is the condition $C2$:

"$\exists\ B_5 \in CH_2$ that during the period between B_5 and the next block malicious miners generate not less than $l+1+u$ blocks, where u is the number of blocks in CH_2 that are in $future(B_5)$".

If $C1$ and $C2$ do not hold, malicious miners still have the chance to realize the attack. The necessary condition for this is $C3$:

"at some moment $T > T_3$ the following inequality holds: $m_4 \ge h_4 + l$".

So, for success of an attack at least one of the conditions $C1$, $C2$, $C3$ has to hold.

Note that we assume that block generation corresponds to Poison process, so block generations on different time intervals and on different branches (or subDAGs) are independent. Then the events $C1$, $C2$ and $C3$ are independent.

According to the condition of theorem, $h_3 \ge z$. Then, using (4) and Lemma 4 in [3] we get that

$$P(C1) = 1 - \sum_{k=0}^{z-1} P_z(k). \tag{8}$$

Next,

$$\begin{aligned} & P(C1\ doesnot\ hold, C2\ holds) = \\ & = P(C1\ doesnot\ hold) \times P(C2\ holds) = \\ & = \sum_{k=0}^{z-1} P_z(k) \left((p'_M)^{z-k} + (p'_M)^{z-k+1} + \ldots + (p'_M)^{z-k+(k_2-1)} \right) \le \\ & \le \sum_{k=0}^{z-1} P_z(k) \left(\frac{(p'_M)^{z-k}}{1 - p'_M} \right) = \sum_{k=0}^{z-1} P_z(k) \left(\frac{(p'_M)^{z-k}}{p'_H} \right). \end{aligned} \tag{9}$$

The probability

$$\begin{aligned} & P(C1\ and\ C2\ doesnot\ hold,\ C3\ holds) = \\ & = (1 - P(C1))(1 - P(C2)) \cdot P(C3) \le \\ & (1 - P(C1)) \cdot P(C3). \end{aligned}$$

Note that the condition $C3$ means "malicious miners will catch up from at least $z-k$ blocks behind", so

$$P(C3) \le \left(\frac{p_M^{'}}{p_H^{'}}\right)^{z-k}.$$

Then

$$P(C1\ and\ C2\ doesnot\ hold,\ C3\ holds) \le$$
$$\le \sum_{k=0}^{z-1} P_z(k)\left(\frac{p_M^{'}}{p_H^{'}}\right)^{z-k}. \tag{10}$$

So, from (10) - (8), we obtain

$$P_z(\alpha_M, \alpha_H, D_H) \le$$
$$\le 1 - \sum_{k=0}^{z-1} P_z(k) + \sum_{k=0}^{z-1} P_z(k)\frac{\left(p_M^{'}\right)^{z-k}}{p_H^{'}} +$$
$$\sum_{k=0}^{z-1} P_z(k)\left(\frac{p_M^{'}}{p_H^{'}}\right)^{z-k} =$$
$$= 1 - \sum_{k=0}^{z-1} P_z(k)\left[1 - \frac{\left(p_M^{'}\right)^{z-k}}{p_H^{'}} - \left(\frac{p_M^{'}}{p_H^{'}}\right)^{z-k}\right] =$$
$$= 1 - \sum_{k=0}^{z-1} P_z(k)\left[1 - \left(\frac{p_M^{'}}{p_H^{'}}\right)^{z-k}\left((p_H^{'})^{z-k-1} + 1\right)\right].$$

The theorem is proved. □

Tables 1 and 2 give the minimal number of confirmation blocks for which the probability of attack is not greater than 10^{-3}, for various network parameters (calculated using (6)). We calculated these probabilities for two different values of general intensity α: for $\alpha = \frac{1}{600} = 0.00167$, as for BTC, and for $\alpha = \frac{1}{60} = 0.0167$, or 10 times bigger than for BTC.

Also note that these Tables are essentially different form the Tables 4 and 5 in [3], which also give the number of confirmation blocks for different network parameters. The matter is that in [3] we considered "classical" blockchain while in this paper we consider DAG with more complicated consensus protocol.

Table 1: The minimal number z of confirmation blocks for which $P_z(\alpha_M, \alpha_H, D_H) \le 10^{-3}$ for various network parameters Δ (in seconds) and p_M, and for $\alpha = 0.00167$ (as for BTC)

p_M	D_H			
	0 sec	***15 sec***	***30 sec***	***60 sec***
0.1	6	6	7	7
0.15	9	9	10	10
0.2	14	14	14	15
0.25	21	21	22	24
0.3	33	35	36	40
0.35	60	65	69	81
0.4	137	154	174	228

Table 2: The minimal number z of confirmation blocks for which $P_z(\alpha_M, \alpha_H, D_H) \le 10^{-3}$ for various network parameters Δ (in seconds) and p_M, and for $\alpha = 0.0167$

p_M	D_H			
	0 sec	***15 sec***	***30 sec***	***60 sec***
0.1	6	7	8	11
0.15	9	11	13	20
0.2	14	17	23	43
0.25	21	29	44	172
0.3	33	55	114	$P_z = 1$
0.35	60	139	$P_z = 1$	$P_z = 1$
0.4	137	203	$P_z = 1$	$P_z = 1$

To calculate the number of confirmation blocks for Tables 1 and 2, we used formulas (6), (7) and calculated

$$\min_{z\ge1} P_z(\alpha_M, \alpha_H, D_H) \le 10^{-3},$$

for fixed α_M, α_H and D_H.

We chose D_H as $0 \le D_H \le 60$ sec that is wider than range of values of D_H which occurs in practice: for example, for BTC, the most often values for D_H are from 0 to 20 sec.

We should note that the first rows (for $D_H = 0$) in Table 1 and Table 2 are identical, while all other rows are very different. The explanation of this fact is the next: the larger time delay D_H for honest miners, the larger advantage exists for malicious miners. Moreover, when $D_H \ne 0$, this advantage essentially depends on the general intensity α of block generation: for large α malicious miners have a large advantage. This fact was proved analytically in [3], and we can see it in formulas (3): the larger $\alpha_M \cdot D_H$, the larger p_M'; the larger p_M', the smaller the security threshold (for $D_H = 0$ the security threshold is 50%). For example, as we can see from the last row of the Table 2, for $D_H = 60$ sec and $\alpha = 0.0167$, the security threshold is not more than 0.3, which means that under these parameters malicious miners cam succeed even with ratio 30%.

4 CONCLUSIONS

Several consensus protocols based on DAG were proposed in last 5-7 years. It should be noted that implementation of these protocols, though, increases efficiency of transactions processing, but carries some risks. There were no rigorous proof (from the mathematical point of view) of resilience to splitting and double spend attacks for any of DAG-based protocols. Constructing of such proofs is an analytically difficult task, even in the case of using of a very simplified mathematical model of network operation.

In this paper, we make a first step in construction of security estimates of DAG-based consensus protocols, building the upper bounds of a double spend attack probability for SPECTRE protocol. To increase the security of SPECTRE consensus protocol, we change a bit a standard vendor's behavior: he should wait for z confirmation blocks which must appear under his observation. The important property of the obtained estimates is the possibility to use them for calculation of both specific values of attack success probability and of the required number of confirmation blocks, depending on the network parameters.

The method for building such estimates essentially depends on consensus protocol itself: it is suitable for SPECTRE but not for all DAG protocols. But some ideas from the method might help to build such estimates for other DAGs.

REFERENCES

[1] Xavier Boyen, Christopher Carr, and Thomas Haines. 2018. Graphchain: A blockchain-free scalable decentralised ledger. In *Proceedings of the 2nd ACM Workshop on Blockchains, Cryptocurrencies, and Contracts*. 21–33.

[2] Cyril Grunspan and Ricardo Pérez-Marco. 2017. Double spend races. *CoRR* abs/1702.02867 (2017). http://arxiv.org/abs/1702.02867

[3] Lyudmila Kovalchuk, Dmytro Kaidalov, Andrii Nastenko, Mariia Rodinko, Oleksiy Shevtsov, and Roman Oliynykov. 2020. Decreasing security threshold against double spend attack in networks with slow synchronization. *Computer Communications* 154 (2020), 75–81.

[4] Serguei Popov. 2016. The tangle. *cit. on* (2016), 131.

[5] Yonatan Sompolinsky, Yoad Lewenberg, and Aviv Zohar. 2016. SPECTRE: A Fast and Scalable Cryptocurrency Protocol. *IACR Cryptology ePrint Archive* 2016 (2016), 1159.

[6] Yonatan Sompolinsky and Aviv Zohar. 2018. Phantom. *IACR Cryptology ePrint Archive, Report 2018/104* (2018).