Upper Bound Probability of Double Spend Attack on SPECTRE

Lyudmila Kovalchuk[1,2]
Joint work with Mariia Rodinko[1,3], Roman Oliynykov[1,3]

[1] Input Output HK, Hong Kong
[2] National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute", Kyiv, Ukraine
[3] V.N. Karazin Kharkiv National University, Kharkiv, Ukraine

September 25th, 2020

New generation of consensus protocols - DAG

DAG - Directed Acyclic Graph

- GHOST
- SPECTRE
- PHANTOM
- Graphchain
- Tangle

Blocks of transactions form a DAG as a distributed ledger

Properties of DAG consensus

Advantages:

- higher throughput;
- faster transaction confirmation time.

Disadvantages:

- difficult and often impossible to set linear order (only partial order);
- difficult to estimate or prove security against main attacks (double spend attack, splitting attack, etc.);
- difficult to set rules for conflict situations.

For today, there are no DAG protocols with strictly proved estimations for probabilities of attacks, including the most popular double spend attack.

SPECTRE. Properties

- One of the earliest and fastest DAG protocols.
- Each block refers to all "leaves" of DAG.
- Very unusual voting procedure for conflict solution (vote of block depends on its position in DAG).

SPECTRE. Description

Mining rules (are very simple)

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.

In what follows we assume that honest miners always act according these mining rules, but malicious miners act as they want, without any rules.

SPECTRE. Main designations

- G - DAG, block $B \in G$;
- $past(B, G) \subset G$ denotes the subset of blocks reachable from B;
- $future(B, G) \subset G$ denotes the subset of blocks from which B is reachable;
- $cone(B, G)$ denotes the union of $past(B, G)$ and $future(B, G)$;
- $virtual(G)$ denotes a hypothetical block that satisfies $past(virtual(G)) = G$.

SPECTRE. Voting rules for conflict solution (I)

- X, Y are blocks with conflicting transaction.
- 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).

Voting rules:

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 $past(Z)$.

SPECTRE. Voting rules for conflict solution (II)

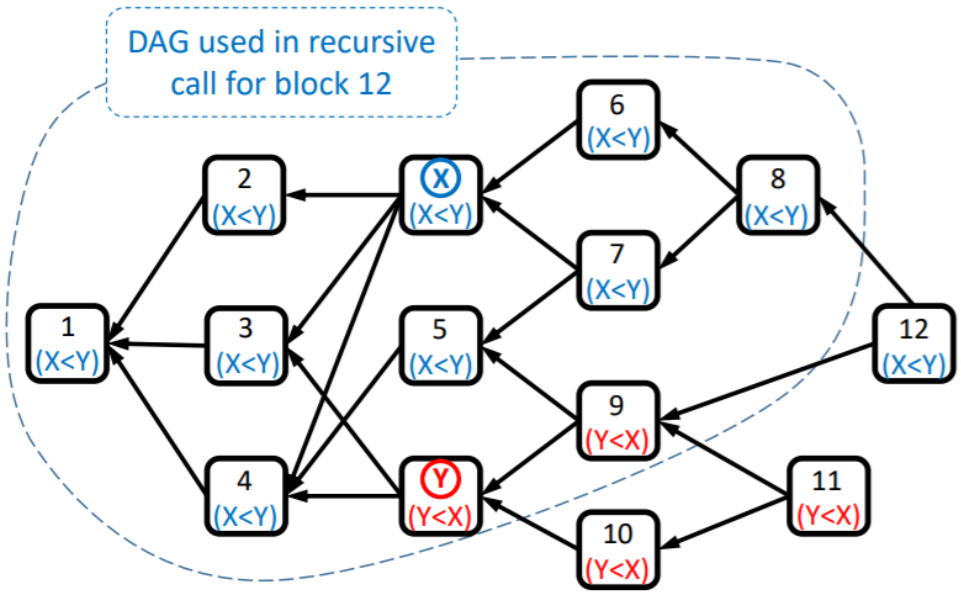


Fig. 1: An example of the voting procedure on a simple DAG. Block x and blocks 6-8 vote $x \prec y$ as they only see x in their past, and not y. Similarly, block y and blocks 9-11 vote $y \prec x$. Block 12 votes according to a recursive call on the DAG that does not contain blocks 10,11,12. Any block from 1-5 votes $x \prec y$, because it sees more $x \prec y$ voters in its future than $y \prec x$ voters.

Figure: An example of the voting procedure on a simple DAG
Source: Yonatan Sompolinsky, Yoad Lewenberg, and Aviv Zohar. SPECTRE: Serialization of Proof-of-work Events: Confirming Transactions via Recursive Elections

Description of the double spend attack (I)

All stages of this attack are given on Figure 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 $CM_2 \cup CM3$.

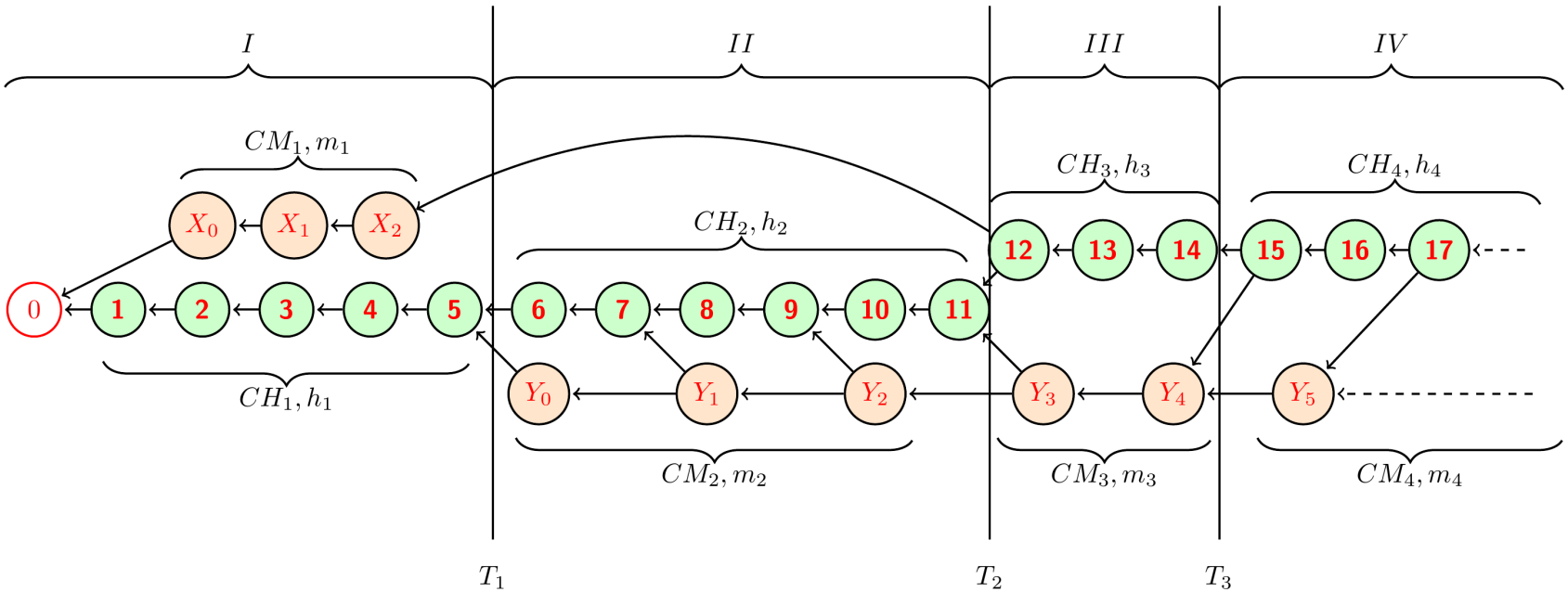
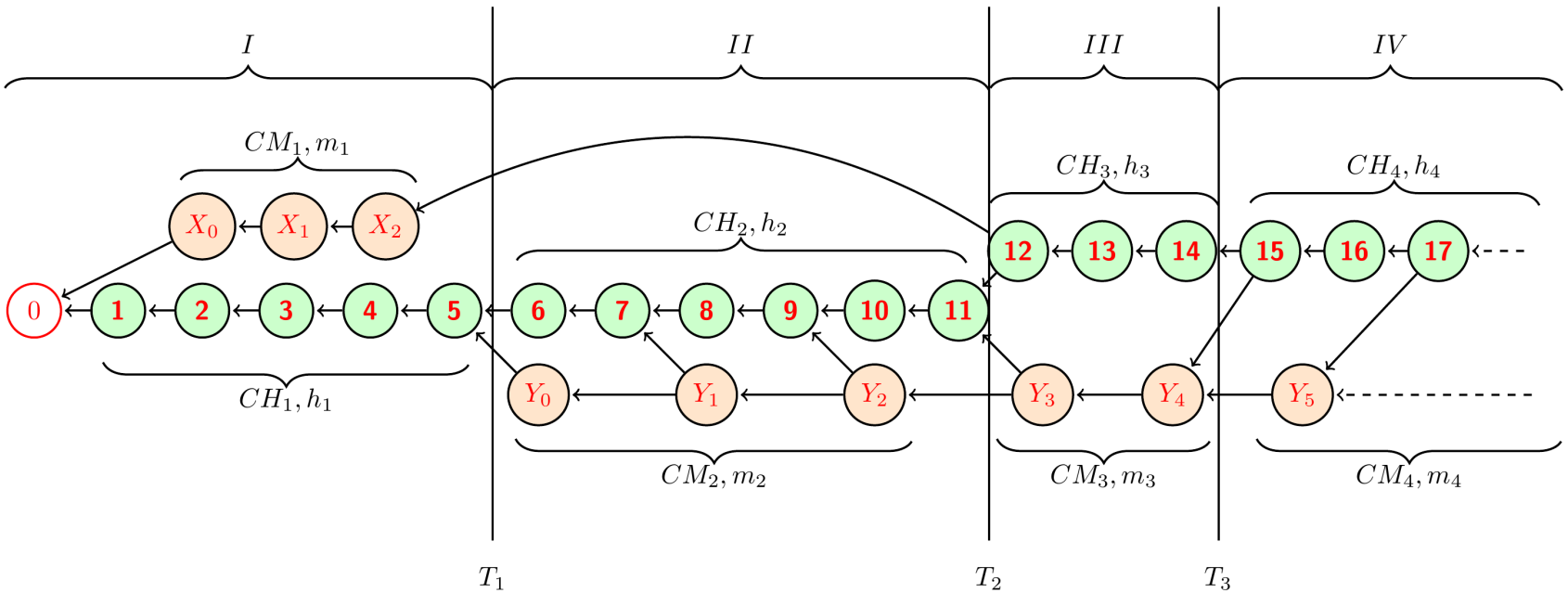


Figure: An example of the hybrid attack

Description of the double spend attack (II)

Stage 1.

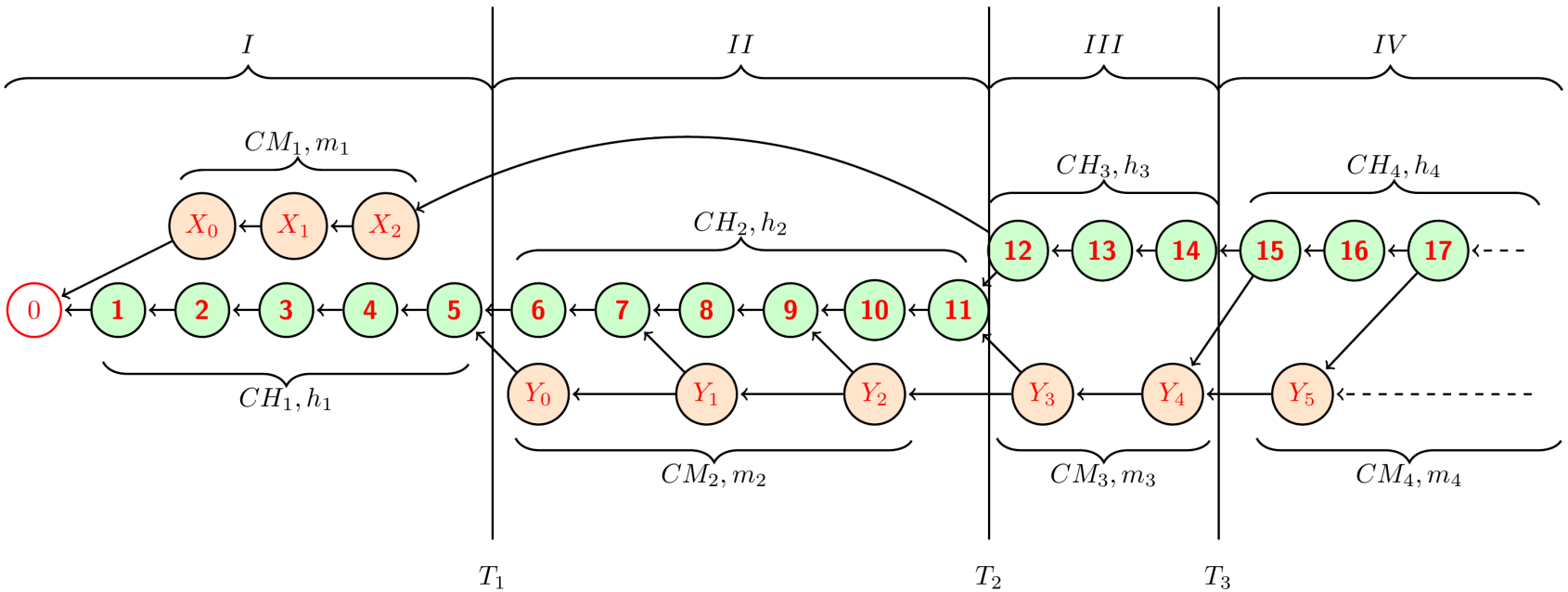
- Honest miners generate the chain CH_1 of the length h_1.
- Malicious miners generate a block X_0 with transaction x_0 and confirm this block, i.e. generate the chain CM_1 of the length m_1 (but do not open this chain).



Description of the double spend attack (III)

Stage 2.
Starts from the moment T_1, when malicious miners decide that they generate enough confirmation blocks for a transaction x_0.

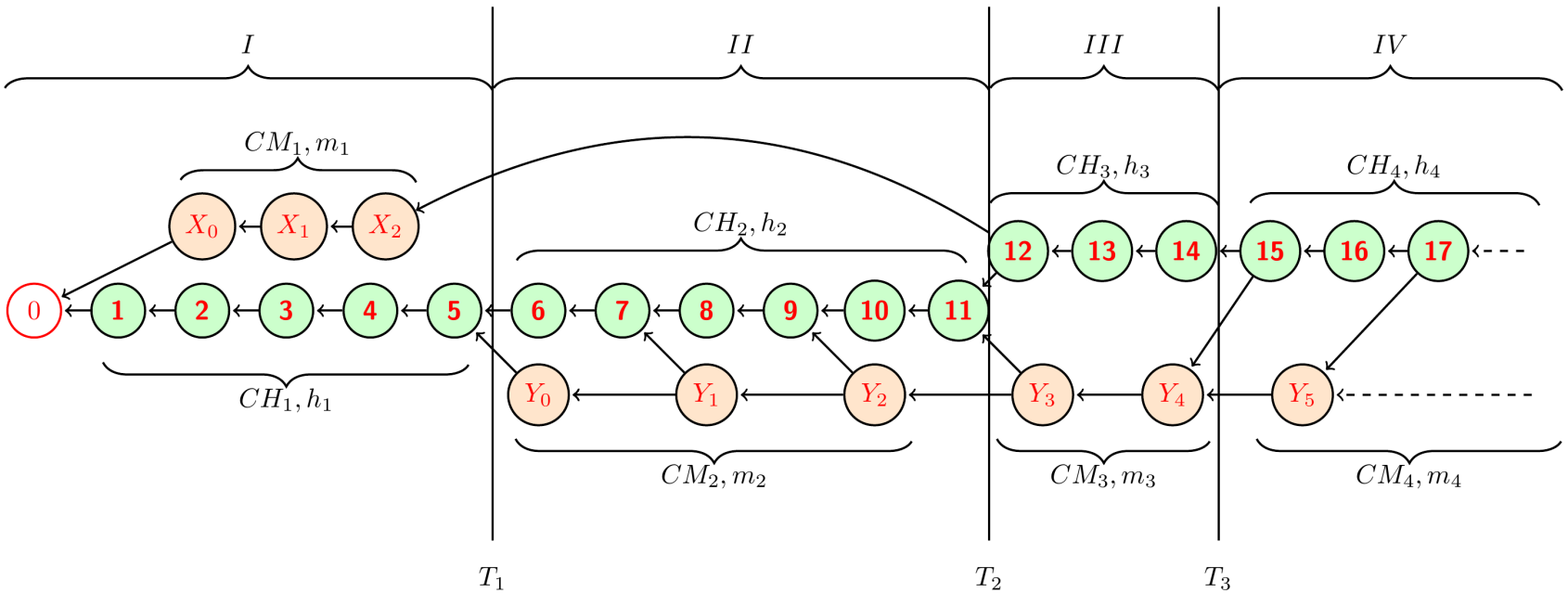
- Malicious miners generate a block Y_0 with a transaction y_0 and confirm this block, i.e. generate the chain CM_2 of the length m_2 (but do not open this chain).
- Honest miners generate the chain CH_2 of the length h_2, which continues the chain CH_1.



Description of the double spend attack (IV)

Stage 3.
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 X_0 with x_0. Malicious miners still do not open the chains CM_2 and CM_3.

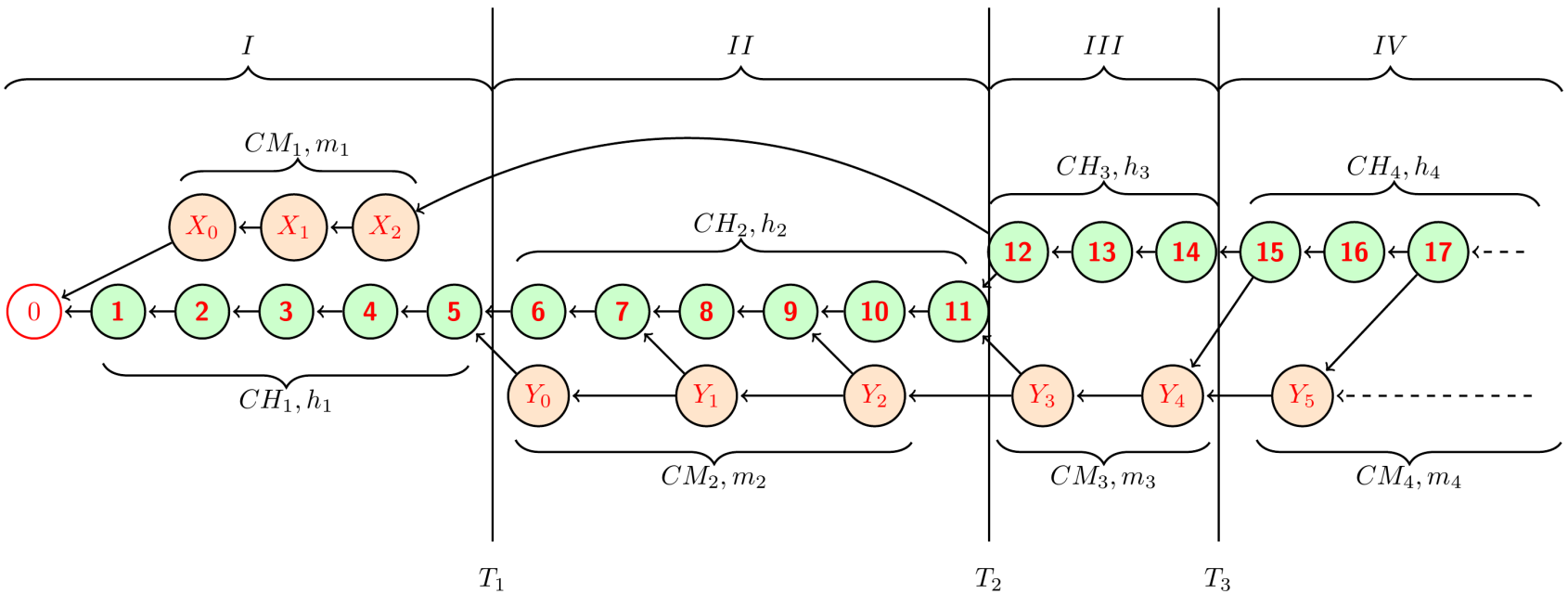


Description of the double spend attack (V)

Stage 4.

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).



Description of the double spend attack for SPECTRE

Results of the voting procedure:

- 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).

Probability of the double spend attack on SPECTRE. Main designations (I)

- $\alpha_H, \alpha_M > 0$ – intensity of block generation for honest miners and malicious miners, respectively.
- $\alpha = \alpha_H + \alpha_M$ – general intensity.
- Parameters of distribution functions:

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

  where T_H, T_M – time before next block.
- The probability that honest miners generate the next block before malicious miners is

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

- D_H – synchronization time for honest miners.

Probability of the double spend attack on SPECTRE. Main designations (II)

- $p_H^{'}$ – 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; $p_M^{'} = 1 - p_H^{'}$ – the probability of the alternative event.

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

- $P_z(k)$ – the probability that malicious miners generate exactly k blocks until honest miners generate and share z blocks.

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

- If $D_H = 0$, then

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

Probability of the double spend attack on SPECTRE

Theorem 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:

$$P_z(\alpha_M, \alpha_H, D_H) \leq 1 - \sum_{k=0}^{z-1} P_z(k) \cdot \left(1 - \left(\frac{p_M^{'}}{p_H^{'}}\right)^{z-k} \left(\left(p_H^{'}\right)^{z-k-1} + 1\right)\right) \tag{6}$$

and in the particular case when $D_H = 0$

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

If $p_M^{'} \geq p_H^{'}$, then $P_z(\alpha_M, \alpha_H, D_H) = 1$.

Numerical results (I)

Table: The minimal number z of confirmation blocks for which $P_z(\alpha_M, \alpha_H, D_H) \leq 10^{-3}$ 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

Numerical results (II)

Table: The minimal number z of confirmation blocks for which $P_z(\alpha_M, \alpha_H, D_H) \leq 10^{-3}$ 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	$P_z = 1$	$P_z = 1$

Conclusions

- 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.
- 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.