Bitcoin Pool-Hopping Detection

Marianna Belotti, Sofiane Kirati, Stefano Secci

Bitcoin Pool-Hopping Detection

Marianna Belotti, Sofiane Kirati, Stefano Secci
Sorbonne Université, CNRS, LIP6, F-75005 Paris, France, Email: {firstname.lastname}@sorbonne-universite.fr

***Abstract*—In the Bitcoin blockchain, rewarding methods for remunerating miners participating in a pool have to meet certain requirements in order to guarantee the proper functioning of the cryptocurrency ecosystem. In particular, these allocation rules reward pool participants in proportion to their contribution in the transaction validation process. Deployed rewarding methods met fairness concerns at the expense of vulnerability to miners exploiting pools' attractiveness for deciding when to mine for a pool and when to 'hop' to another one resulting more attractive: a phenomenon called pool-hopping. The most used score-based methods are designed to prevent this practice, but are not completely hopping proof.**

In this work, we propose a methodology to analyze the pool-hopping phenomenon, focusing on the detection of pool-hoppers. Analyzing those Bitcoin transactions that pools create for rewarding its participants, it is possible to determine time epochs where miners worked. Thus, we analyze those miners that have worked intermittently for pools adopting a rewarding system which pays out for each validated block. This evaluation leads us qualifying the miners that have hopped along with their hopping behavior and financial performance.

I. Introduction

In the Bitcoin network, *mining* is the procedure through which *miners* can earn crypto coins by finding solutions to a computationally hard puzzle which validates a *block* of bitcoin transactions [1]. In order to be rewarded on a regular basis, miners can decide to cooperate reducing the uncertainty of the remunerations over time. Cooperation results in the formation of *pools*, which receive a reward for the validated block and split it among the participants.

The Bitcoin monetary policy establishes the reward amount one receives once it validates a block (12.5 BTC, valid until the third halving scheduled for 29 May 2020 when the block mining reward is decreasing to 6.25 BTC) – noting that a block can contain more than 2000 transactions, while respecting the block size limit of 1 MB. There exist several functions which reallocate block-validation bitcoins inside a pool for remunerating its miners. These rewarding methods have to meet certain requirements in order to guarantee the proper functioning of the ecosystem. The aim of forming pools is to share efforts and rewards thus, participants should be rewarded in a fair manner with respect to their contribution to the pool, i.e., proportionally to the hashing power invested in the pool. Original simple rewarding methods met this fairness criterion at the expense of vulnerability to some miners strategic behaviors. Schrijvers et al. describe in [2] weaknesses of pools' remuneration systems to withholding strategies [3].

We do address in this work the miners' strategy of exploiting pools' attractiveness to decide when to mine for a pool and when to leave it for joining another one. This practice, known as *pool-hopping*, has been deeply studied in [4], showing new (*score-based*) methods specifically designed to prevent the phenomenon. The most used ones are not completely hopping proof since pool participants may take advantage of changes in system or pool parameters for their own benefit. The gain of using such a strategy is a higher reward than the fair share for a miner's contribution. Since a pool with a given computational power receives a constant reward, such a behavior benefits pool-hoppers to the detriment of those miners who mine continuously. Thus, hoppers detection might be as useful for *pool-operators* as it is for the miners themselves (mining pools are managed by a pool manager, also known as pool operator).

Bitcoin public nature provides us with both *generation transactions* [5] and transactions used by pools to remunerate their participants. The first are those transactions by means of which new bitcoins are generated and issued in the system. The second are the so called "*rewarding transactions*" which are transferring the funds received through the generation process to pool participants in a chain of ownership [6], i.e., new minted bitcoins are transferred from pools' addresses (*input*) to miners' ones (*output*) in the form of Unspent Transaction Output, UTXO [7]. Proper analysis of the public ledger can reveal a lot of information on the network users. Deanonymization techniques applied on a selected set of users do contribute in enhancing the quality of the extracted information. In our analysis, the Bitcoin actors we are interested in are the ones involved in the rewarding transactions, i.e., pools and miners, remunerated according to pools policies; the first step of our work is determining the related *user network* [8].

Rewarding methods with direct correspondences between block validation and miners' payout do help in hoppers detection. Through a targeted analysis of the transactions ledger, it is possible to place a miner – intent on working for a pool – within a time interval denoted as *epoch*. This time period is nothing more than an estimate of a miner's working time for a given pool. Hoppers detection consists in comparing miners' epochs in different pools. All the actors in a pool have full knowledge of the internal mining activities carried out over time, which is not the case for any user external to pool. Once pool-hoppers have been identified among all the miners interacting with multiple pools, it is possible to describe the hopping evolution over time.

The paper is organized as follows: Section II provides an overview on the rewarding systems used in the Bitcoin blockchain network and addresses pool-hopping practice – we focus our analysis on those score-based methods

that are not completely hopping proof. In Section III we present the procedure for determining the user sub-network resulting from the analysis of the generation, transferring and rewarding transactions – the sub-network we are interested in is the one showing direct connections between pools and users that are receiving new minted bitcoins. Section IV is devoted to hoppers detection. Section V presents the measurement results. We conclude the work in Section VI.

II. Rewarding methods and Pool-hopping

The Bitcoin validation procedure consists in computing hashes from a given input data-string until the hash value meets a predefined threshold. The latter, known as *difficulty target* – we refer to as D – represents the value at which all transactions blocks are validated every 10 minutes, on average. The difficulty D is adjusted every 2016 blocks in accordance with Bitcoin policies. Whenever a valid hash is found, the miner or the pool which release the valid block is rewarded with an amount of coins denoted by B. This amount halves approximately every 4 years.

Concerning mining pools, the pool operator pays out pool-members based on how much qualified work they perform. This work is quantified with the number of submitted *shares*, which are near to valid solutions: for getting a share, miners participating in a pool try to solve the original crypto-puzzle, but with a looser constraint on the hash they have to compute. Therefore, the pool manager redistributes B among miners according to the number of submitted shares. A variety of rewarding methods have been designed to fairly remunerate miners in proportion to their work. However, some methods are more vulnerable than others to miners' attempts of achieving a higher reward than the fair allocation.

A. Rewarding Methods

Original no-longer used rewarding methods [4] were paying out according to a division into *rounds*, where a round is the time elapsed between two different block validations performed by the same pool. Thus, pool distributes the block reward among miners at round end, in proportion to their contribution within the round. New methods replace such a round-based remuneration opting for the attribution of a *score* to each share. Score-based methods register the time of a share submission, assign a value to the share according to a scoring function and remunerate the submitter correspondingly. Thus, rewards calculation is no more linked to the concept of round but it is based on the scoring systems in use. However, payouts are still executed on a round basis.

Besides rewarding fairness, score-based methods also offer pool-hopping resistance. Meni Rosenfeld lists in [4] several score-based rewarding systems. We describe in the following the two most popular methods in use nowadays.

1) Pay-Per-Last-N-Shares (PPLNS) method [4]: It represents an allocation rule rewarding miners for their recently submitted shares – eventually computed at different rounds. The block reward B, instead of being distributed only among actual round participants, can also be assigned to those miners that have submitted shares in a previous round. Then, for short actual rounds, the rewarding calculation may consider also shares submitted in previous rounds. Note that, with the PPLNS system it can happen that some submitted shares may not be considered for the reward calculation (i.e., shares in long actual rounds). N varies with the difficulty parameter D; it can be up to twice D (rounded down to integer), as an upper-bound threshold. More accurate implementations tend to set $N < 2D$; e.g., the author in [10] proposes to score shares as the inverse of D, and determines N as a given portion of the score, hence $N \leq D$.

2) Slush method: Slush pool [9] has been the first mining pool developing a rewarding method aimed at combating pool-hopping. Block rewards are divided in proportion to the score miners obtained with their reported share(s) in the rounds. Slush's scoring system dynamically calculates a score for a specified share according to its submission time. More precisely, the score at time t_0 for a share s submitted at time t_s is given by:

$$score(s, t_0) = \exp\left(\frac{t_s - t_0}{\Lambda}\right) \tag{1}$$

where Λ is a constant value representing the speed at which the score decreases over time. An exponentially decreasing scoring is well suited due to its invariance to time shift.

B. Pool-Hopping

Pool-hopping is a strategic behavior influenced by the attractiveness of mining in terms of expected payoffs [11]. In other words, pool-hopping consists in mining for a pool only when its attractiveness is high and leaving it when it is not (i.e., directing computing power towards another pool).

Meni Rosenfeld shows in [4] that pools using proportional systems encourage this practice in the sense that pool-hopping results more profitable than mining continuously. Since for proportional methods each share is paid out according to the total number of submitted shares in the round, the longer is the round the less is the reward per share. Each share payoff corresponds to $\frac{B}{S}$, where S is the total number of shares submitted to the pool during a round. Thus, submitting shares to a pool as long as the rounds are short and then hopping elsewhere results in a higher gain than the fair share [2]. What makes this practice, known as *early mining*, feasible is the miners' knowledge of the round start time and the number of shares taken into account for the reward calculation. PPLNS and Slush methods propose scoring functions that do not encourage early mining practice, thus the pools are preventing hopping behaviors on round length basis.

However, as the Slush score is a function of the time elapsed since a share submission, pools' attractiveness is eventually affected by the hash-rate fluctuations over time. Pools' hashing power directly influences the number of shares found and the round length. In addition, both PPLNS and Slush systems' 'hoppability' is affected by difficulty (D) and reward (B) adjustments. Therefore, advanced pool-hopping strategies are possible for both Slush and basic PPLNS implementations. Nevertheless, the two methods present different levels of hoppability as described in [10].

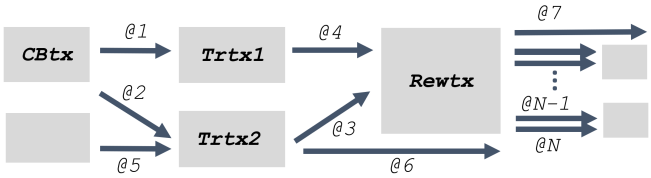

Fig. 1: Transaction sub-network. Each arrow is labeled with sender's address. CBtx, Trtx, Rewtx are respectively coinbase, transferring and rewarding transactions.

III. User Sub-network Inference Process

Bitcoin transactions move funds from one or more 'inputs' to one or more 'outputs' containing all the references of previous and next owners. Users receive bitcoins in the form of unspent transaction output (UTXO) in their *addresses*, i.e., strings identifying sources and destinations for bitcoin payments. In order to be spent, funds contained in the output of a transaction must turn into the input of a new one. Inputs and outputs must match in the sense that the entire value of the previous transaction(s) has to be transferred. Thus, whenever a user aims at sending an amount lower than the UTXO value, a second output containing a *changing address* is added. That is, a bitcoin address held by the sender appears as an output with the purpose of transferring funds back.

A. Coinbase, rewarding and transferring transactions

Each transactions block contains a generation transaction remunerating the pool or the miner that validates it. These type of transactions does not specify any previous UTXO to spend, its input field (denoted as *coinbase*) contains arbitrary data; generation transactions, also known as *coinbase transactions*, provide funds that cannot be spent for at least 100 confirmed blocks. Due to this requirement, Bitcoin system rewards only for valid blocks in the *longest chain* (i.e., the longest blockchain branch). While mining pools are financed by coinbase transactions, miners are remunerated through ordinary transactions we denote as *rewarding transactions*. Ideally, they should transfer the coinbase UTXO, however this is not always the case. Some pools use additional transactions – we refer to as *transferring transactions* – that move funds to specified addresses within the pool, which reward miners later on.

All these transactions are publicly readable from the blockchain ledger and can be easily gathered through Bitcoin-Core [12]. This information leads to construct the **transaction sub-network** [8] representing the flow of bitcoins between a specific subset of transactions – of coinbase, rewarding and transferring types – over time. Fig. 1 example shows pool-miners interactions where coinbase transactions are inputs of transferring transactions whose outputs consist in rewarding ones. Arrows represent the funds flow.

The topology of Fig. 1 example synthesizes a generic view of the Bitcoin network, where one may detect the relationships between various input and output addresses. For coinbase, rewarding and transferring transactions, the addresses involved certainly belong to pools and miners however, it is not known to which user they belong exactly since each of them can have multiple addresses. Moreover, it is considered a good practice for confidentiality to generate and use new addresses for each operation on the network. As hoppers detection requires the analysis of the interactions among Bitcoin specific users, it is necessary to map each bitcoin actor with its associated addresses.

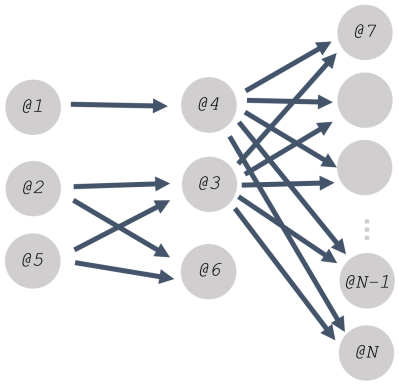

Fig. 2: Address sub-network related to Fig. 1 example.

B. Address-User Mapping Procedure

The address-user mapping problem can be solved using two basic heuristics introduced in [13], which derive from the expert knowledge of Bitcoin protocols and common practices. The mapping procedure we use is based on the concatenated application of the two heuristics. The first heuristic, addressing transaction inputs only, exploits the concept of *multi-input transactions*: it assumes that all the inputs in a transaction belong to the same bitcoin user. This heuristic results reasonable in its simplicity since (i) different users likely very rarely collaborate in a single transaction, and (ii) very often it happens that users move funds from their multiple addresses to one or multiple deposit accounts. The second heuristic deals with changing addresses: it associates input addresses with output ones when they result to be completely new (i.e., they never appeared in the blockchain before).

Therefore, applying the mapping procedure one can derive the user sub-network from the address sub-network. The relationship between addresses can be easily inferred from the transactions sub-network; Fig. 2 shows the address sub-network corresponding to the Fig. 1 example. Once the address sub-network is derived, addresses are associated with their users following the two heuristics; Fig. 3 shows an instance of user sub-network for the given example.

It is not possible to obtain a perfectly true **user sub-network**, since heuristics are not perfect and addresses do not contain any information about their owners. In order to maximize the precision, due to the limited number of coinbase, transferring and rewarding transactions with respect to the overall volume of transactions, we have applied the mapping procedure to the entire transaction network (including ordinary transactions as well). Doing so, we can provide an accurate representation of miners and pools interactions, since we are associating with these specific users the highest possible number of employed addresses.

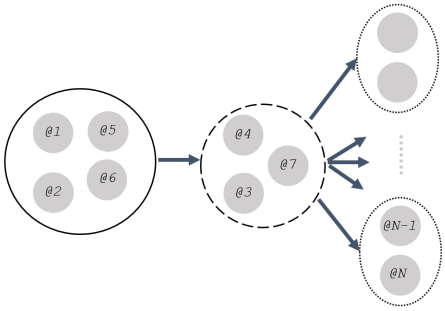

Fig. 3: User sub-network related to Fig. 1 example.

The application of such deanonymization techniques for the mapping allows us building an interaction network between pools and miners. Pool-hopping consists in mining for two or more pools during a round, thus hoppers submitting shares to multiple pools are remunerated accordingly by multiple pools. Hoppers detection requires the analysis of those miners who are connected to several pools.

IV. Hoppers detection

The previous section provides us with those network pre-processing steps for detecting pool-hoppers. From the constructed user sub-network it is possible to select those miners that are remunerated from multiple pools. In order to determine whether a miner that submits shares to more than one pool is really implementing a hopping strategy, it is necessary to focus on its rewarding transactions. Our analysis addresses those pools adopting rewarding methods which are paying out miners on a round basis.

A. Simplistic case

For pools using per-round rewarding systems, rewarding transactions should be ideally linked to the corresponding rounds they are paying for. This link could be found in their inputs, namely in the coinbase transactions providing pools with the rewards that have to be redistributed. It would then be possible, by analyzing the rewarding transactions, to place a miner in a certain round. In Fig. 4 we show how to identify miners' working period: a miner rewarded at time A is associated with the corresponding round of pool A that is marked in yellow. Then, considering multiple pools and their rounds, it would be enough to check if the same miner is present in two overlapping rounds of different pools. The procedure results quite trivial once pools' rewarding transactions are ordered according to their corresponding rounds, that is, in accordance with the corresponding coinbase transactions.

In a 2-pool case, if the same miner is present in two consecutive (ordered) rewarding transactions belonging one to a pool and one to another one, we can declare it as a hopper. There may be doubtful situations whenever the same miner is found in two rewarding transactions of different pools which are not directly consecutive (i.e., after ordering we could have two transactions created by pool A, one created by pool B and the same miner present in the first and third transaction). In this cases it is mandatory to analyze the rounds length to establish if the miner adopted or not a hooping strategy.

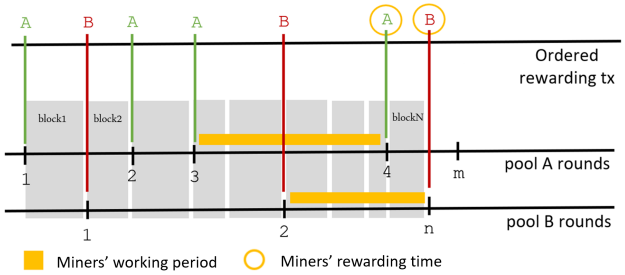

Fig. 4: Simplistic miners' positioning. Blocks are indicated with gray boundaries. Vertical lines associate each round end with the corresponding rewarding transaction.

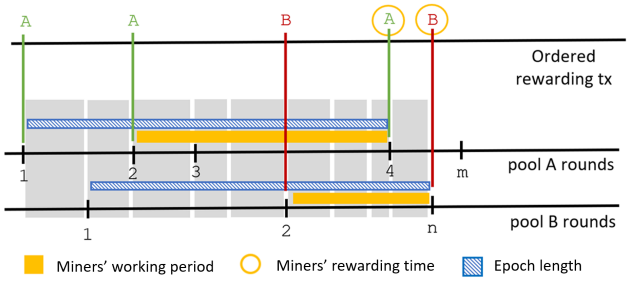

Fig. 5: Realistic miners' positioning.

B. Realistic case

In reality, pools do not associate exactly a paying transaction to each round. Miners receive their payouts once they cross a predefined *sending threshold* (Slush pool adopts a threshold of 0.001 BTC [9]). This means that in reality pools pay for miners' work on multiple rounds with a single rewarding transaction. In Fig. 5 pool A is rewarding a miner for its work in two rounds. Rewarding transactions no longer use the reward provided by a single coinbase but they have several generation transactions as inputs. In this way, it is no longer possible to place a miner in its working round.

Miners paid out with the rewarding transactions can be placed within a time interval that we denote as "*epoch*". This time period can coincide with one or more rounds. We set the beginning of an epoch with the oldest coinbase used as an input in the rewarding transaction. The epoch ends with the round allowing miners to reach the sending threshold, namely the round closing about 100 blocks before the one in which the rewarding transaction takes place. As shown in Fig. 5 the epoch end coincides with the last round end. The epoch start depends on the coinbase transactions used in the rewarding transaction. Pools may pay out miners with funds generated during the rounds they work for or they may use older coinbase transaction as in Fig. 5. Note that we assume payments are executed as soon as funds are available (after 100 confirmations), which results in neglecting the processing time for the rewarding transactions.

By positioning miners in epochs, we cannot identify the exact rounds they work for. They may have worked continuously in all the rounds within the epoch or they could have worked intermittently. The idea is always to check if a miner is present in two overlapping time periods, however it can be defined as a hopper with a certain probability depending on epochs' length. As for the ideal case,

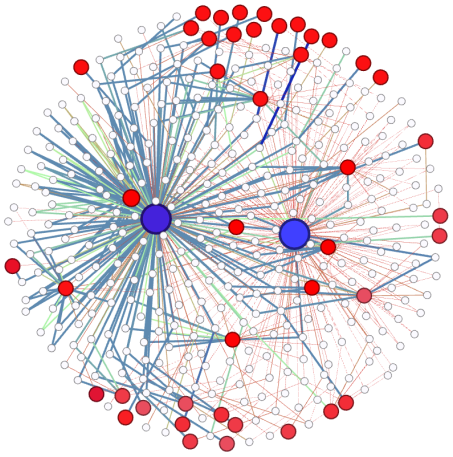

Fig. 6: Pool-miners graph representation. The left blue node is Kano pool and the right blue node is Slush pool. Detected pool-hoppers are highlighted in red. Edge thickness and color depend on the frequency of users interactions.

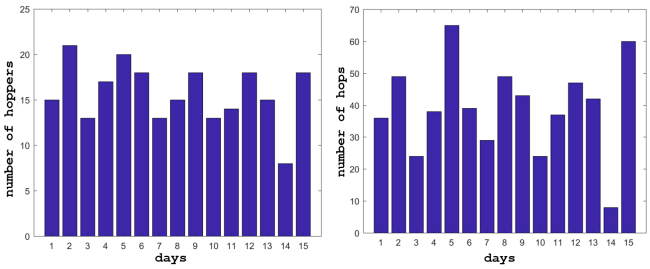

Fig. 7: Daily number of hoppers and hops.

transaction ordering should simplify the detection procedure. Rewarding transactions can be sorted by membership blocks, that is, ordering procedure refers to the coinbase transactions corresponding to epoch ends. For the 2-pool case in which a miner is found in two consecutive rewarding transactions with different creators, we can conclude with any doubt that the miner is a pool-hopper. This is due to epoch definition, since it ends with that round in which miners submitted the number of shares necessary to reach the threshold value. Hence, there is the certainty of miners' presence in the epochs' final rounds.

By considering only consecutive transactions, other cases where hopping is possible are overlooked. First of all, this approach detect only hoppers in epochs' final rounds without considering the other rounds within them. Moreover, as in the simplistic case, we can have hoppers paid out in two rewarding transactions that are not adjacent. Epochs length could help us in detecting miners who are likely to hop, thus in this cases we are talking about *potential hoppers*.

V. Measurement Results

We analyze the user sub-network involving miners interacting with the two most popular mining pools adopting score-based rewarding methods: Slush pool [9] and Kano pool [14]. Since these pools opt for a per-round rewarding system, we can apply the hopping detection technique showed in Section IV. We aim at capturing *real-hoppers* (i.e., miners that have definitely hopped) and the result of

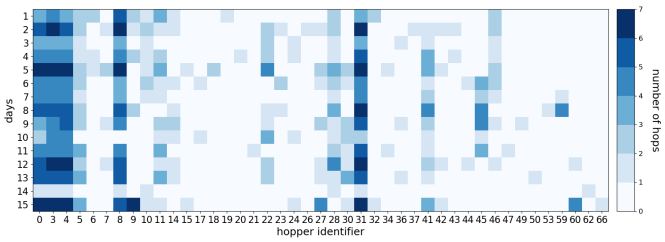

Fig. 8: Heat map: daily hopping intensity per hopper.

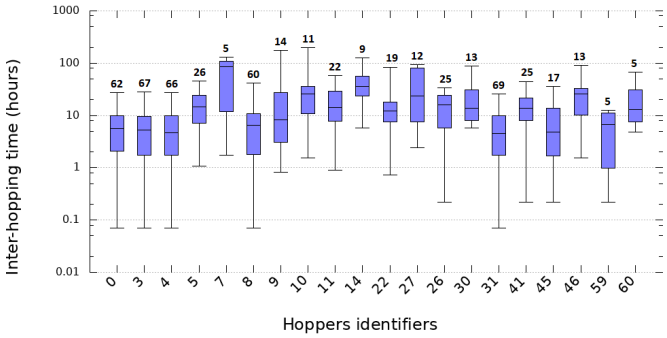

Fig. 9: Inter-hopping time boxplot distributions (logscale).

their behavior. Thanks to the user sub-network, we can describe the phenomenon evolution over a predetermined time window.

The analyzed period is April 6-20, 2016, chosen for its massive transactions flow (over 5 millions transactions) such that Kano pool recorded its maximum hashing power [14]: with a substantial number of miners attracted in joining the pool it has been a remarkable period for hopping opportunities.

First, we detect all the miners, interacting with the two pools, that can potentially adopt a hopping strategy. Afterward, we capture the real-hoppers among them. Moreover, we perform a daily analysis on the number of hoppers and the number of 'hops' made, both globally and individually. Fig. 6 captures the interactions of the miners with Kano and Slush pools (represented by blue nodes). Overall, 43 pool-hoppers were detected within a population of 69 miners active with the two pools (26 miners are not hopping within the time period, thus they are mining continuously for one of the two pools), and an approximate global population of 150.000 miners. Fig. 7 shows the trend of the number of pool-hoppers and their hops during the considered 15 days.

Fig. 8 further characterizes the phenomenon on a daily basis, showing for each of the identified hoppers the number of hops per day. We can roughly spot 3 different types of hopper: (i) *frequent hoppers*, miners that opt for a hopping strategy more than 10 times (e.g., #0, #3, #4), (ii) *occasional hoppers*, performing a limited number of hops, less than 10 times (e.g., #6, #7, #14), and (iii) *changing-pool hoppers*, miners that hop just once in the time window (e.g., #19, #20, #21). Changing-pool hoppers may have simply changed the pool to work for, since they do not present an intermittent behavior. Frequent and occasional hoppers represent together the 46.4% of the hopping miners, while 11 miners perform only a single hop.

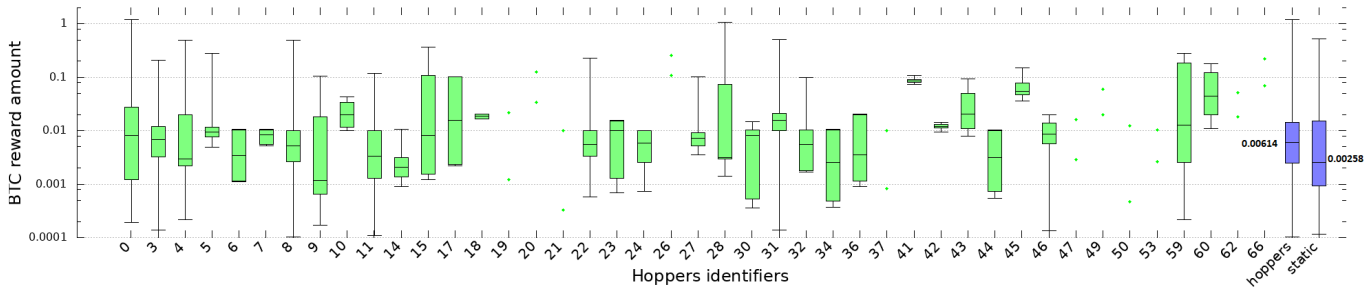

Fig. 10: Boxplot distributions of rewarding transactions BTC values (logscale).

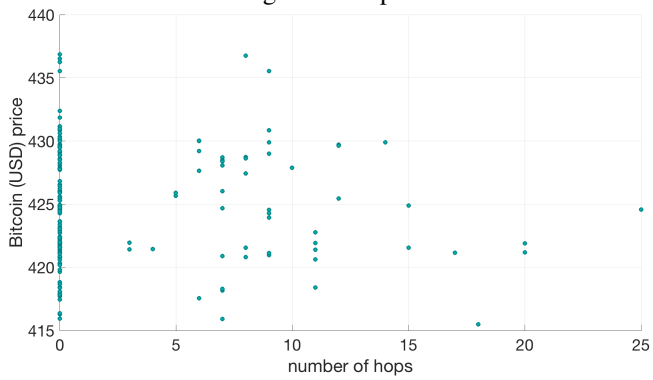

Fig. 11: Correlation plot: BTC price vs number of hops.

In addition we quantify the hopping frequency for those miners that are changing pool more than 5 times in the observed period. Fig. 9 presents the distribution of the *inter-hopping* times (i.e., the time elapsing between two successive hops performed by the same hopper), with boxplots (reporting a quartile box, minimum and maximum) indicating over the number of hops. Roughly half of the miners shows a median time less than 12 hours, with small variance, while the other half shows a higher median with an inter-quartile range of many hours. There is no evidence of the adoption of a strict hard-coded frequency, hence the hopping decision appears to be event-based as expectable.

Moreover, we are interested in quantifying the advantage in hopping. The distribution of hoppers' revenues over the time window is presented in Fig. 10 with boxplots; the hoppers' order is the same than the one in Fig. 8; for some hoppers, only few transactions exist and in that case instead of the boxplot only few points are plotted. The last two boxplots on the right-side of Fig. 10 cover all hoppers together and all static miners (that mine continuously without hopping) in the time period; we find a median transaction reward 3 times higher for hopping miners than for static ones. This highlights a very important gain justifying pool-hopping; note that miners' computational power cannot be inferred from the ledger, and in practice for pool-hoppers it may significantly differ from that of the 'typical' miner.

Finally, we report in Fig. 11 the correlation between the number of hops and the Bitcoin (USD) price evolution over the 15 days; we compute the average Bitcoin price and the number of hops performed within a time window of two hours. No evident correlation between the two distributions emerges, i.e., we find here a confirmation that pool-hopping strategies do not appear to be hard-coded in miners behavior as a function of the Bitcoin actual value.

VI. Summary

This paper addresses the issue of pool-hopping phenomena detection for round base rewarding systems adopted by the Slush and Kano pools. For this purpose we use known deanonymization techniques in order to identify miners and pools type of users of the Bitcoin blockchain. We propose a particular hoppers detection technique based on rewarding transactions ordering.

Our analysis determines that the hopping phenomenon is relatively limited in a time window which could have led to a high hopping propensity (only 0.02% of the miners did perform pool-hopping between the two selected pools). We show that hoppers' rewards significantly exceed those of static miners.

Besides these facts, we have also determined that pool-hoppers have disparate behaviors, likely not correlated, and not dependent on the actual value of the cryptocurrency. This suggests that there may not be a shared methodology to implement a hopping strategy, and that pool-hoppers do proceed on a do-it-in-your-own, manual, fashion.

References

[1] Satoshi Nakamoto, "Bitcoin: A peer-to-peer electronic cash system." (2008). Available at https://bitcoin.org/bitcoin.pdf (May 7, 2018)

[2] Schrijvers, Okke, et al. "Incentive compatibility of bitcoin mining pool reward functions." *International Conference on Financial Cryptography and Data Security*. 2016.

[3] Bag, Samiran, Sushmita Ruj, and Kouichi Sakurai. "Bitcoin block withholding attack: Analysis and mitigation." *IEEE Transactions on Information Forensics and Security* 12.8 (2017): 1967-1978.

[4] Rosenfeld, Meni. "Analysis of bitcoin pooled mining reward systems." *arXiv preprint arXiv:1112.4980* (2011).

[5] Antonopoulos, Andreas M. *Mastering Bitcoin: unlocking digital cryptocurrencies*. O'Reilly Media, Inc., 2014.

[6] Ametrano, Ferdinando M., Hayek Money: The Cryptocurrency Price Stability Solution (August 13, 2016). Available at SSRN: http://dx.doi.org/10.2139/ssrn.2425270 (May 7, 2018).

[7] Bitcoin Wiki. "Transaction". Available on-line at https://en.bitcoin.it/wiki/Transaction (May 7, 2018).

[8] Koshy, Philip, Diana Koshy, and Patrick McDaniel. "An analysis of anonymity in bitcoin using p2p network traffic." *International Conference on Financial Cryptography and Data Security*. 2014.

[9] Slush Pool. "Reward System". Available on-line at https://slushpool.com/help/manual/rewards (May 7, 2018).

[10] Rosenfeld, Meni. "Analysis of hashrate-based double spending." *arXiv preprint arXiv:1402.2009* (2014).

[11] Raulo. "Optimal pool abuse strategy", 2011. Available on-line at http://bitcoin.atspace.com/poolcheating.pdf (May 7, 2018).

[12] BTC Core website: https://bitcoin.org/end/bitcoin-core (May 7, 2018).

[13] F. Reid, M. Harrigan. "An analysis of anonymity in the bitcoin system." *Security and privacy in social networks*. Springer. 2013.

[14] BTC.com. "Statistics". Available on-line at https://btc.com/stats/pool/KanoPool (May 7, 2018).