This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TIFS.2016.2623588, IEEE Transactions on Information Forensics and Security

Bitcoin Block Withholding Attack : Analysis and Mitigation

Samiran Bag⋆, Sushmita Ruj† and Kouichi Sakurai‡

⋆ Newcastle University, UK. Email: samiran.bag@newcastle.ac.uk
†Indian Statistical Institute, India. Email: sush@isical.ac.in
‡ Kyushu University, Japan. Email: sakurai@csce.kyushu-u.ac.jp

***Abstract*—We address two problems: firstly, we study a variant of block withholding (BWH) attack in Bitcoins and secondly, we propose solutions to prevent all existing types of BWH attacks in Bitcoins. We analyze the strategies of a selfish Bitcoin miner who in connivance with one pool attacks another pool and receives reward from the former mining pool for attacking the latter. We name this attack as 'sponsored block withholding attack'. We present detailed quantitative analysis of the monetary incentive that a selfish miner can earn by adopting this strategy under different scenarios. We prove that under certain conditions, the attacker can maximize her revenue by adopting some strategies and by utilizing her computing power wisely. We also show that an attacker may use this strategy for attacking both the pools for earning higher amount of incentives.**

More importantly, we present a strategy that can effectively counter block withholding attack in any mining pool. First, we propose a generic scheme that uses cryptographic commitment schemes to counter BWH attack. Then we suggest an alternative implementation of the same scheme using hash function. Our scheme protect a pool from rogue miners as well as rogue pool administrators. The scheme and its variant defend against BWH attack by making it impossible for the miners to distinguish between a partial proof of work and a complete proof of work. The scheme is so designed that the administrator cannot cheat on the entire pool. The scheme can be implemented by making minor changes to existing Bitcoin protocol. We also analyze the security of the scheme.

***Index Terms*—Bitcoin mining, Block withholding attack, Selfish miner, Mining pool, Commitment schemes.**

I. Introduction

Bitcoin is a popular cryptocurrency first proposed by Satoshi Nakamoto [1] in 2008. The transactions are put in a publicly verifiable ledger called a *blockchain*. A blockchain consists of many blocks, which in turn verify multiple transactions. Users who create and verify blocks are called miners. Miners receive newly created Bitcoins as incentive. To regulate the flow of Bitcoins, blocks are created once in approximately 10 minutes. The miners have to solve a puzzle (as a proof of work - PoW) in order to claim the incentive. Though there are alternative currencies like Permacoin [2] and Retriecoin [3] that use storage as opposed to computation for minting currency, the proof of work used by Bitcoin is well recognized to be the best proof of work scheme as of now.

In [4], Kroll et al. showed that Bitcoin mining is not "the fixed, rule-driven, incentive-compatible system" as some of its patronizers claim. Block withholding attacks [5], [6], [7], [8] are well known and are discussed in Bitcoin forums. In this attack rogue miners try to increase their incentive by reducing the winning probability of other miners. In Bitcoin network, multiple miners join hands in order to form a mining pool to sum up their computing power with the aim of yielding a massive computing powerhouse. In such a mining pool every miner needs to regularly submit a proof of work to the pool administrator to demonstrate their work towards solving the proof of work associated with a Bitcoin block. Solving this proof of work is less difficult than solving the proof of work that could be associated with a Bitcoin block. These are called partial proofs of work. The partial proofs are a superset of the Bitcoin proofs of work. Hence, a full proof of work that can fetch an incentive of 25BTC is also a partial proof for any pool. So, these partial proofs serve two purposes. Firstly, they can be used to verify that a miner is indeed spending her computing resources for solving the PoW of Bitcoin system. Secondly, computation of partial proofs are valid work towards solving the Bitcoin PoW and computation of these partial proofs is not a wastage of the computing power of the pool. However, checking all partial proofs incur an extra overhead on the pool manager which can be reduced by choosing a particular difficulty level of the partial proofs.

Every miner of a pool attempts to find a proof of work on a transaction set that contains a coinbase transaction which separates this transaction set from the transaction sets of other mining pools or solo miners. This coinbase transaction leads the pool administrator to claim the mining reward in case the pool wins the mining game. Thus, in case a miner belonging to some pool P can finds a solution of the puzzle, it will have only two choices, either to submit it to the pool administrator, or to conceal the fact that it has indeed found a solution. The miner can not submit the solution to the Bitcoin network through some other pool. Also even if the pool protocol allows the miner to submit a block with a full PoW directly to the Bitcoin network, the pool will claim the reward thanks to the coinbase transaction included in the block.

In BWH attack, a rogue miner shares with the pool administrator only those partial proofs of work that are not full proofs and conceals all full proofs computed by her. The pool administrator remains oblivious about the withheld block and thinks that the miner, like every other miner, is indeed trying to utilize her computing power in

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TIFS.2016.2623588, IEEE Transactions on Information Forensics and Security

order to solve the PoW puzzle. The pool, unaware of this malicious behavior of the miner, shares its revenue with her. Thus, the malicious miner earns incentive at the expense of the honest miners of the pool without requiring to do anything useful for the pool.

Block withholding attack was first proposed in [5]. In [9] Courtois and Bahack showed that a rogue miner may act dishonestly for her personal gain and put the reputation of Bitcoin in danger by indirectly using the computing power of honest miners for her own gain and thus depriving them from the incentive they deserve. In [8] Luu et al. gave a quantitative analysis of the amount of incentive a miner may gain by carrying out block withholding attack on a victim pool. They showed that the gain of the rogue miner who uses BWH attack is because she reduces the probability of win of the victim pool. This increases her own chance of winning the Bitcoin mining game. It was reported that on June 13, 2014, a large scale BWH attack was conducted against the mining pool Eligius [9] causing a loss of 300BTC at the expense of honest miners.

In [6], Laszka et al. provided a game theoretic analysis of BWH attack and showed interesting results about the long term viability of a Nash equilibrium between attacking pools in the Bitcoin network. Eyal et al. [7] proposed a mining game where a mining pool tries to infiltrate other pools by sending some of its members to join the other pool, thus lunching a BWH attack on the victim pool. They showed that with two pools or more pools attacking each other, there will be honest members of the pools who will earn less than their expected amount of revenue.

In this paper we have discussed a similar scenario. However, our work differs from that of Eyal et al. [7] in the following ways: (a) We have focused on the amount of revenue generated by the attacker who launches BWH attack on a pool and gets some reward from a different pool for attacking the previous one. Eyal et al. calculated expressions of the revenues generated by the pools when the attacking scenario of the network is in a state of equilibrium. (b) In our model, an attacker can split her computing power and can attack both the pools. Not on this, she can accept reward from each of the pools for attacking the other pool. However, Eyal et al. discuss an attacking scenario where the attackers are pool members and share the revenue obtained from the victim pool with miners of the rogue pool.

A. Motivation and Related Work

Eyal et al. [7] defined and analyzed a game where identical mining pools attack each other. They have showed that in such a scenario where the mining pools attack each other, there exists a Nash equilibrium where all of them earn less than what they should have if none had attacked.

Luu et al. [8] performed detailed study of BWH attack and showed the gain of the attacker under different scenarios. They proved that this attack could indeed increase the gain of the attacker. They derived various expressions for the attackers gain under different settings. The intuition behind the attacker's incentive earned through BWH attack is that she uses part of her computing power in reducing the probability of win of a pool, thereby increasing her own chance of winning.

In this work we propose what we call a "sponsored block withholding attack". We observe that by carrying out a BWH attack on a victim pool the attacker indirectly increases the probability of win of another pool (as well as of her own if she has got additional resources to mine privately). Hence, she can conspire with some other pool to spend a fraction of her computing power to attack one pool and thereby reducing the victim pool's chance of winning. In such case, she may be rewarded by the selfish pool for attacking its rival pool and the amount of reward should be proportional to the increase of gained incentive of the selfish pool that the attacker causes by attacking the victim pool. As such, the expected gain of the attacker will be higher than what was calculated by Luu et al. in [8]. This is the motivation behind our analysis which we provide before section VI of this paper.

BWH attack may have disastrous effect on the long term viability of Bitcoin system. In section VI, we seek mitigation of BWH attack. We propose a concrete measures that could be used to defeat BWH attack and thus will make Bitcoin mining pools safer place for investment of miners' computing power. We first discuss a generalized scheme that could be used along with any cryptographic commitment protocol. Then we propose a variant of the same scheme which uses hash function in stead of commitment scheme. The scheme is so designed to not allow a miner to predict in advance if one of her pool share can be utilized to generate Bitcoins by the pool. She only gets to know that after she has delivered the pool share to the pool administrator. The details on these schemes are discussed later. It is to be noted that our countermeasure applies to all known BWH attacks, and is not just limited to "sponsored BWH attacks".

B. Contribution

The contribution of this paper is two fold. The paper analyzes BWH attack with respect to a specific system model and also proposes two mining schemes as a remedy to this attack.

- In the section IV we analyze the revenue function of an attacker who with the support of another pool launches BWH attack on a target pool. This attack increases the revenue of the second pool and she can share her extra incentive with the attacker which is the main observation based on which we calculate and analyze the revenue function of the attacker. In V we analyze the revenue of the attacker who attacks both the pools.

  Our study is different from [7] in that, [7] studied a scenario where one mining pool sends its members to attack another pool. The miner who acts as a double agent transfers all her earned incentive back to the malicious pool which then distributes all its earned revenue evenly among its members. In our model however the miner acts independently and gets a fixed amount of incentive for every block she withholds from the pool administrator.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TIFS.2016.2623588, IEEE Transactions on Information Forensics and Security

- The main results of this paper are given in Theorem 1, 2, 3 and 5. These results deal with optimal strategies of a BWH attacker who wants to efficiently use her computing power to earn as much as possible. In Theorem 1 we prove a crucial result that the attacker can use to maximize her total amount of earned incentive by devising an appropriate attacking strategy.
  In Theorem 2, we prove that when the computing power of the pools are maintained at a constant value, then the attacker needs to attack both the pools simultaneously for increasing her gain.
  In Theorem 3 we show that the attacker gains more from attacking rather than mining independently when some feasible conditions are met. In Theorem 5 we show that there exists a feasible condition, under which the attacker can maximize her gain by attacking both the pools simultaneously when the computing power of the mining pools are not kept as constant i.e attacking just a single pool would not fetch her maximum gain. This result is more similar to the result of Theorem 2, the only difference is that in Theorem 5, unlike Theorem 2 the computing power of the two pools are not maintained at a constant value.
- Finally, in section VI, we propose an improvement on the existing Bitcoin mining scheme that could make it insusceptible to block withholding attack. We first propose a generalized scheme that uses cryptographic commitment protocol. We then provide a modified scheme that makes use of hash function in stead of commitment protocol. The basic idea behind both the schemes is the same. More clearly, the second scheme is a minor variation of the first one. We show that both the schemes can make Bitcoin mining immune against block withholding attack without compromising on the key properties of the existing mining schemes. Our countermeasure not only holds good for the attacks discussed in this paper, but also holds good for any sort of BWH attacks discussed in [5], [9], [7] etcetera. The add-ons are very simple to implement and they do not affect the overhead of computing a proof of work.

The section VI is independent of the sections IV and V with no inter-link as such. Hence, a reader may skip section IV and V and move directly to section VI, if the reader is only interested in learning about the defense mechanism against BWH attack.

C. Organization

The rest of the paper is organizes as following: in section II we define the 'gain' of the miner and give the reader an idea about cryptographic commitment protocol and mining in Bitcoin system. In section III we describe the attack model and our assumptions in this study. In section IV, we analyze the incentive of the attacker who attacks a single pool only. In section V, we study the incentive of the attacker who victimizes both the pools. Lastly, we propose an improvement on the Bitcoin mining scheme that can defeat the block withholding attack on mining pools. We conclude the paper in section VII.

II. Preliminaries

A. Notations & Definitions

Serial No.	Notation	Description
1	α	Computing power of attacker
2	β	Fraction of computing power used to attack pool P
3	δ	Fraction of computing power used to attack pool P'
4	γ	Fraction of extra gain a pool shares with the attacker
5	p	Computing power of pool P
6	p'	Computing power of pool P'

TABLE I: Table of notations

Definition 1. *We use the same definition of 'gain' as in [8]. Gain of a miner is defined to be the fraction amount of incentive earned by a miner by performing a mining job using her computing power according to a specific mining strategy with respect to the total incentive earned by the entire Bitcoin network. If g_m be the incentive earned by a particular miner and g_B be the incentive earned by the entire Bitcoin network then gain of the miner is g_m/g_B.*

B. Commitment Scheme

A non-interactive commitment scheme [10] C consists of three probabilistic polynomial time algorithms $C.Setup()$, $C.Commit()$ and $C.Open()$.

- $C.Setup(1^\kappa)$: Given the security parameter κ, the $Setup(1^\kappa)$ algorithm outputs the public parameter $\mathcal{CK}$ of the commitment scheme.
- $C.Commit(\mathcal{CK}, x)$: It takes a string $x \in \{0,1\}^\kappa$ and outputs a pair $(com, decom)$. In this paper, for ease of writing, we sometime omit the $decom$ value and write $com \leftarrow C.Commit(\mathcal{CK}, x)$. However, the reader may note that every execution of the $C.Commit(\mathcal{CK}, x)$ algorithm outputs both the commitment and the opening value.
- $C.Open(\mathcal{CK}, com, decom)$: It takes the commitment com, the decommitment $decom$ and and outputs x or the error symbol $\perp$.

Security Property

A cryptographic commitment scheme has these properties

Let C be a commitment scheme.

1) Hiding Property : For all probabilistic polynomial time adversary $\mathcal{A}$
   $|Pr[\mathcal{CK} \longleftarrow C.Setup(1^\kappa), (m_0, m_1, st) \longleftarrow \mathcal{A}(\mathcal{CK}), b \xleftarrow{U} \{0,1\}, (com_b, decom_b) \longleftarrow C.Commit(\mathcal{CK}, m_b) : b = \mathcal{A}(\mathcal{CK}, st, com_b)] - \frac{1}{2}| \leq negl(\kappa)$
2) Binding Property : For all probabilistic polynomial time adversary $\mathcal{A}$
   $Pr[\mathcal{CK} \leftarrow C.Setup(1^\kappa), (com, decom, decom') \leftarrow \mathcal{A}(\mathcal{CK}), m \leftarrow C.Open(com, decom), m' \leftarrow C.Open(com, decom') : m \neq m' \wedge m' \neq \perp] \leq negl(\kappa)$
3) Relaxed Binding : For all probabilistic polynomial time algorithm $\mathcal{A}$,

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TIFS.2016.2623588, IEEE Transactions on Information Forensics and Security

$Pr[\mathcal{CK} \leftarrow C.Setup(1^\kappa), (m, st) \leftarrow \mathcal{A}(\mathcal{CK}) (com, decom) \leftarrow C.Commit(m, \mathcal{CK}), decom' \leftarrow \mathcal{A}(\mathcal{CK}, st, com, decom'), m \leftarrow C.Open(com, decom') : m \neq m' \wedge m' \neq \perp] \leq negl(\kappa)$

III. System Model for Sponsored BWH Attack

Here we discuss some assumptions that we have used in our analysis of 'sponsored block withholding attack' in the next two sections, that is in section IV and V. Readers are reminded that these assumptions are unrelated to the discussion in section VI, where we propose a counter-measure against block withholding attack. In section IV and V, we consider a model of mining pool with a single pool administrator. The responsibility of the administrator is to co-ordinate the calculation of the PoW. She selects the set of transactions and hence determines the transaction Merkle root and other parameters and the individual miners try to construct the proof of work based on these parameters. The administrator also sets a difficulty level for the partial proofs of work which the miners need to submit to the administrator regularly for scrutiny. The difficulty level of the partial proofs of work is usually lower than that of the Bitcoin system but should not be high enough to exceed the capacity of the administrator. The administrator examines every partial proof of work submitted to her until she finds one that matches the difficulty level of the PoW of the entire Bitcoin network and if she finds any partial PoW that matches the target difficulty level of the Bitcoin network, she submits it to the Bitcoin system and claims the reward. We also assume that the pool administrator is honest and cannot be compromised.

We now describe the attack in detail. We assume the computing power of the entire Bitcoin system be 1. There are two pools P and P'. Let $\mathcal{A}$ be an attacker whose computing power is α. She divides her computing power into two parts; the first part she uses for private mining and the other part is used for launching block withholding attack against a mining pool. The attacker joins a victim pool P pretending to be a loyal miner and keeps submitting her partial proofs regularly to the pool administrator. These shares are nothing but proofs of work with a lower difficulty level. Whenever by chance the attacker finds a real solution to the proof of work having a difficulty level same as that of the entire Bitcoin system, she refrains from submitting it to the pool. Neither can she directly submit the same to the Bitcoin system [8]. This is because the existing pool protocol requires one to submit a block through the pool administrator only. Most of the pools require the miners to submit the block through the pool administrator. Even if the pool protocol allows her to submit the block directly to the Bitcoin network, the coinbase transaction contained in the block will cause the mining reward to go directly to the wallet of the pool administrator. The miner will not be able to obtain anything in addition to her fair share which is distributed evenly by the administrator of the pool.

She instead submits it to the pool P' which is a rival of pool P. P' has a secret pact with $\mathcal{A}$. Whenever $\mathcal{A}$ withholds a block from submitting to P, the pool P' offers some rewards to $\mathcal{A}$. It is easy for P' to check the validity of a block submitted by the attacker assuming that P' knows the set of transactions the pool P is working on including the transaction Merkle root making it possible for P' to verify the proof of work that the attacker withheld from pool P. Since, we assume that P is an open pool so it is plausible for P' to get hold of the set of transactions P is working on. Thus the gain of the attacker increases as she has got a sponsor for launching attack on P. The reward that P' gives to the attacker $\mathcal{A}_1$ is proportional to the expected amount of gain, P' makes from $\mathcal{A}$'s attack on P. Thus, both P' and $\mathcal{A}$ gains at the expense of the computational resources of honest miners of P. The expected gain of the attacker is necessarily higher than normal BWH attack where she does not fetch any incentive from rival pools for carrying out this attack.

IV. Analysis of the incentive earned by the attacker who attacks a single pool

Our main focus is to study of the gain of an attacker who uses the withheld block to earn more incentive from other selfish pools. Let us consider a situation when there are two mining pools P and P' in the entire network and a single attacker $\mathcal{A}$. The computing power of the attacker is α. The attacker($\mathcal{A}$) uses β fraction of its computing strength for mining in the pool P and uses the rest of $1-\beta$ fraction of its computing power for mining solo. The computing power of pool P and P' is p and p' respectively. The attacker $\mathcal{A}$ mines in P with $\alpha\beta$ computing power and it submits its shares to the pool only when the share does not correspond to a valid solution of the actual proof of work. In case the attacker $\mathcal{A}$ indeed finds a valid block, it does not submit it to the pool. Neither can it submit the same to the Bitcoin system and claim the reward. Instead it submits this to the administrator of the pool P'. The pool administrator of P' gives her some incentive for launching attack on the pool P and thus increasing the probability of win of the pool P'. The distinction between our model and the model considered in [7] is that in our model the attacker performs independently, not as a member of the pool P'. Hence, the attacker does retains all her earned incentives. In the model discussed in [7], however the attacker functions as a liaison of another pool and transfers all her ill-gotten incentives to that pool who in turn distributes it among all of its members. Let γ be the fraction of the incentive given to the attacker $\mathcal{A}$ by the pool P' for launching block withholding attack on P. We assume that the network has no other player i.e

$$\alpha(1-\beta) + p + p' = 1. \tag{1}$$

Lemma 1. *The gain of the pool P' when the attacker launches block withholding attack on the pool P is given by $G_{P'} = \frac{\alpha\beta p'}{1-\alpha\beta}$.*

Proof. If the attacker does not use her computing power to launch BWH attack on pool P, the computing work force of the entire network would have been 1. Then the expected gain of P' would be p'. But when the attacker $\mathcal{A}$ uses $\alpha\beta$ fraction of her computing power for attacking P, the effective computing power of the network goes down to

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TIFS.2016.2623588, IEEE Transactions on Information Forensics and Security

α	β	γ	p'	ΔG_h^O
0.200000	0.500000	0.700000	0.340000	0.248016
0.100000	0.900000	0.700000	0.400000	0.417396
0.200000	0.400000	0.700000	0.340000	0.214171
0.160000	0.600000	0.700000	0.330000	0.254052
0.200000	0.800000	0.700000	0.400000	0.394558
0.200000	0.900000	0.700000	0.400000	0.408747
0.200000	0.960000	0.700000	0.400000	0.414236
0.200000	0.990000	0.700000	0.400000	0.416150
0.200000	0.980000	0.700000	0.400000	0.415573

TABLE II: Tabular presentation of values of ΔG_h^O for different values of the corresponding parameters.α is the computing power of the attacker. β is the fraction of computing power that the attacker uses to attack the pool P. γ is the fraction of extra incentive that the pool P' shares with the attacker which P' earns due to the attack on P. p' is the computing power of P'.

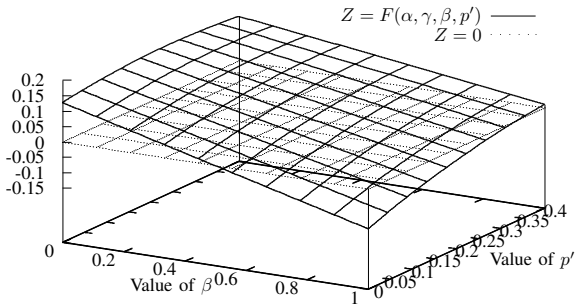

Fig. 1: Graphical depiction of the behavior of $Z = F(\alpha, \gamma, \beta, p')$ when $\alpha = 0.2, \gamma = 0.7$

$1 - \alpha\beta$ and hence the gain of P' increases to $\frac{p'}{1-\alpha\beta}$. Thus, the increase of gain of P' is $\frac{p'}{1-\alpha\beta} - p' = \frac{\alpha\beta p'}{1-\alpha\beta}$. □

Let us now calculate the expected incentive of $\mathcal{A}$. Since, $\mathcal{A}_0$ uses $\alpha\beta$ of its computing power to attack P, the active computing power of the network is $1-\alpha\beta$. Here we assume that all other miners of the pool P and all miners of P' are honest. The gain of $\mathcal{A}$ is threefold. Firstly, the gain of $\mathcal{A}$ from mining privately is equal to $G_1 = \frac{\alpha(1-\beta)}{1-\alpha\beta}$. The expected share of incentive obtained for mining in P is $G_2 = \frac{p-\alpha\beta}{1-\alpha\beta}\frac{\alpha\beta}{p}$. Again, the incentive obtained from P' for carrying out attack on a competitor pool (P) is given by $G_3 = G_{P'} * \gamma = \frac{\alpha\beta p'\gamma}{1-\alpha\beta}$.

So, the total gain of the attacker is

$$G = G_1 + G_2 + G_3 = \frac{\alpha(p - \alpha\beta^2 + p'\beta\gamma)}{p(1-\alpha\beta)}. \quad (2)$$

It can be easily seen from this equation that the incentive of the attacker increases if she attacks the bigger pool and receives incentive from the smaller one.

Note that when the attacker only used classical BWH attack without having a secret agreement with pool P', the expected gain it could have obtained was $G' = G_1 + G_2 = \frac{\alpha(p-\alpha\beta^2)}{p(1-\alpha\beta)}$.

Hence, the fraction of increase in gain the attacker bestows upon herself with respect to gain of original BWH attack is given by

$$\Delta G_{BWH}^O = \frac{G}{G'} - 1 = \frac{p'\beta\gamma}{p - \alpha\beta^2}. \quad (3)$$

Since, $\alpha\beta < p, \alpha\beta^2 < \alpha\beta < p$. So, $\Delta G_{BWH}^O > 0$. If the attacker was an honest miner, the incentive she would have obtained through mining honestly is $G_h = \alpha$. So the ratio of the incentive she obtains now by third party backed block withholding attack is $\Delta G_h^O = \frac{G}{G_h} - 1 = \frac{\alpha\beta p + p'\beta\gamma - \alpha\beta^2}{p(1-\alpha\beta)}$. Replacing p by $1 - p' - \alpha + \alpha\beta$, we get,

$$\Delta G_h^O = \frac{G}{G_h} - 1 = \frac{\alpha\beta - \alpha\beta p' - \alpha^2\beta + \alpha^2\beta^2 + p'\beta\gamma - \alpha\beta^2}{(1 - p' - \alpha + \alpha\beta)(1 - \alpha\beta)} \quad (4)$$

Theorem 1. *Let $\beta_0 \in (0,1)$ be such that the total gain of the attacker $G|_{\beta=\beta_0} = \max(G|_{\beta=\beta'}) : \forall \beta' \in [0,1]$. Then β_0 should satisfy,*

$$A\beta_0^2 + B\beta_0 + C = 0$$

where, $A = -\alpha^2(1-\gamma)p', B = 4\alpha^2 - 2\alpha^3 - 2\alpha^2 p' + 2\alpha p' - 2\alpha, C = \alpha^3 - 2\alpha^2 + 2\alpha^2 p' - 2\alpha p' - \alpha\gamma p' + \alpha + \alpha p'^2 - \gamma p'^2 + \gamma p'$.

Proof. We have proved that in the proposed setting, the total gain of the attacker $\mathcal{A}$ is $G = \frac{\alpha(p-\alpha\beta^2+p'\beta\gamma)}{p(1-\alpha\beta)}$. After replacing p with $1 - p' - \alpha + \alpha\beta$ in Eq (2), the total gain of the attacker is given by,

$$G = \frac{\alpha(1-p'-\alpha+\alpha\beta-\alpha\beta^2+p'\beta\gamma)}{(1-p'-\alpha+\alpha\beta)(1-\alpha\beta)}.$$

Differentiating with respect to β we get,
$\frac{\partial G}{\partial \beta} = \frac{\alpha(-2\alpha\beta+\gamma p'+\alpha)}{(1-\alpha\beta)(\alpha\beta-p'-\alpha+1)} + \frac{\alpha^2(-\alpha\beta^2+\gamma p'\beta+\alpha\beta-p'-\alpha+1)}{(1-\alpha\beta)^2(\alpha\beta-p'-\alpha+1)} - \frac{\alpha^2(-\alpha\beta^2+\gamma p'\beta+\alpha\beta-p'-\alpha+1)}{(1-\alpha\beta)(\alpha\beta-p'-\alpha+1)^2}$. Simplifying, we get $\frac{\partial G}{\partial \beta} = \frac{Num}{Den}$, where (after simplification),

$Den = (1-\alpha\beta)^2(1-p'-a+\alpha\beta)^2$, and

$Num = -\alpha^3(1-\gamma)p'\beta^2 + (4\alpha^3 - 2\alpha^4 - 2\alpha^3 p' + 2\alpha^2 p' - 2\alpha^2)\beta + (\alpha^4 - 2\alpha^3 + 2\alpha^3 p' - 2\alpha^2 p' - \alpha^2\gamma p' + \alpha^2 + \alpha^2 p'^2 - \alpha\gamma p'^2 + \alpha\gamma p')$.

It is easy to see that $Den \neq 0 \forall \beta \in [0,1]$. If G is to be maximized at $\beta = \beta_0$, $\frac{\partial G}{\partial \beta}|_{\beta=\beta_0} = 0$. So, $-\alpha^3(1-\gamma)p'\beta_0^2 + (4\alpha^3 - 2\alpha^4 - 2\alpha^3 p' + 2\alpha^2 p' - 2\alpha^2)\beta_0 + (\alpha^4 - 2\alpha^3 + 2\alpha^3 p' - 2\alpha^2 p' - \alpha^2\gamma p' + \alpha^2 + \alpha^2 p'^2 - \alpha\gamma p'^2 + \alpha\gamma p') = 0$. Since, $\alpha \neq 0$, we can write the equation as $-\alpha^2(1-\gamma)p'\beta^2 + (4\alpha^2 - 2\alpha^3 - 2\alpha^2 p' + 2\alpha p' - 2\alpha)\beta + (\alpha^3 - 2\alpha^2 + 2\alpha^2 p' - 2\alpha p' - \alpha\gamma p' + \alpha + \alpha p'^2 - \gamma p'^2 + \gamma p') = 0$. Hence, the result. □

Lemma 2. *Let $F(\alpha, \gamma, \beta, p') = A\beta^2 + B\beta + C$. Then $F(\cdot)$ is a decreasing function on β when $\alpha \leq 0.25$.*

Proof. $\frac{\partial F}{\partial \beta} = 2A\beta + B$. Now, $A = -\alpha^2(1-\gamma)p' < 0$. $B = 4\alpha^2 - 2\alpha^3 - 2\alpha^2 p' + 2\alpha p' - 2\alpha = -2\alpha^3 - 2\alpha^2 p' - \alpha(1-2p') - \alpha(1-4\alpha)$. Now, P' is the smaller pool and hence, $p' < 0.5$. So, $-\alpha(1-2p') < 0$. Also, $\alpha(1-4\alpha) < 0$. So, $\frac{\partial F}{\partial \beta} < 0$. So, $F(\cdot)$ is a decreasing function on β. □

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TIFS.2016.2623588, IEEE Transactions on Information Forensics and Security

Figure 1 gives a graphical presentation of the values of $F(\alpha,\gamma,\beta,p') = A\beta^2 + B\beta + C$ where A, B and C are as defines as in Lemma 1. We have plotted the values of $F(\cdot)$ by varying the values of β and p' while keeping α fixed. The figure shows that $F(\cdot)$ takes both positive and negative values when p' is less than a certain limit. Also, we observe that $F(\cdot)$ decreases monotonically, in adherence to the result of Lemma 2. So, we can infer that there exist an β_0 such that $F(\alpha,\gamma,\beta_0,p')$ equals 0 when p' is less than a certain limit. At this point the gain of the attacker $\mathcal{A}$ peaks. Also, since $F(\alpha,\gamma,\beta_0,p') > 0$ for $\beta < \beta_0$, G is an increasing function of β for $\beta < \beta_0$. Similarly as $F(\alpha,\gamma,\beta_0,p') < 0$ for $\beta > \beta_0$, G is a decreasing function of β for $\beta > \beta_0$ when p' is less than the said limit. When p' is beyond the said limit, G is always an increasing function of β, and hence the attacker's best strategy would be to invest all her computing power in attacking the pool P. But when p' is sufficiently low, there is a $\beta_0 \in [0,1]$ that maximizes her gain. This β_0 corresponds to the intersection of the two curves $Z = F(\alpha,\gamma,\beta,p')$ and $Z = 0$ in Figure 1. The value of β_0 can be obtained by solving the quadratic equation $A\beta^2 + B\beta + C = 0$. The value of β_0 that maximizes the gain of the attacker is $\{\frac{-B+4AC}{2A}, \frac{-B-4AC}{2A}\}$.

Note that since $F(\cdot)$ is monotonically decreasing, there exists exactly one value of β_0 that maximizes the gain G. This entails either $0 < \frac{-B-4AC}{2A} < 1$ or $0 < \frac{-B-4AC}{2A} < 1$ but not both.

V. Attacking both the pools simultaneously

Let us again consider another situation where the attacker launches this attack on both the pools P and P'. Let us assume like before the computing power of the attacker is α. She splits her computing power into 3 parts; $\mathcal{A}_1, \mathcal{A}_2, \mathcal{A}_3$. with computing power equal to $\alpha\beta$, $\alpha\delta$ and $1-\alpha\beta-\alpha\delta$. She uses $\alpha\beta$ of her computing power to attack P and get the incentive from P'. Similarly, she uses $\alpha\delta$ of her computing power to attack P' and get an incentive from P. In such case, her expected gain from private mining will be $G_1 = \frac{\alpha(1-\beta-\delta)}{1-\alpha\beta-\delta\gamma}$. The share of reward she obtains from P is given by $G_2 = \frac{p-\alpha\beta}{1-\alpha(\beta+\delta)}\frac{\alpha\beta}{p}$. The reward she obtains by mining in pool P' is given by $G_3 = \frac{p'-\alpha\delta}{1-\alpha\beta-\alpha\delta}\frac{\alpha\delta}{p'}$. The incentive she gets from P' for attacking P is given by $G_4 = \frac{\alpha\beta p'\gamma}{1-\alpha\beta}$. Similarly, the incentive she gets from the pool P for attacking the pool P' is given by $G_5 = \frac{\alpha\delta p\gamma}{1-\alpha\delta}$. Hence, the total gain of the attacker is given by $G = \sum_{i=1}^{5} G_i = \frac{\alpha-\alpha\beta-\alpha\delta}{1-\alpha\beta-\alpha\delta} + \frac{p\alpha\beta-\alpha^2\beta^2}{p(1-\alpha\beta-\alpha\delta)} + \frac{p'\alpha\delta-\alpha^2\delta^2}{p'(1-\alpha\beta-\alpha\delta)} + \frac{\alpha\beta p'\gamma}{1-\alpha\beta} + \frac{\alpha\delta p\gamma}{1-\alpha\delta}$.

A. When the computing power of the pools are constant

Next, we study the gain of the attacker who attacks both the pools P and P'. In the first place, we assume that the computing power of P and P' are constant for the entire period. That is when a miner invests in any of the pools, some of her extra computing power that had been kept idle, the pool administrator shuts down some of its own systems and maintains the computing power of the system at the same level as before. Alternately, when some miner exits the pool, the administrator uses its back up computing resources to compensate the loss. Thus, the computing powers of P and P' remain the same always. The rationale behind this is to keep the overhead on the pool administrator constant. As we have discussed before, pool members regularly submit partial proof of work to the pool administrator which are then verified by the administrator. Thus, the pool administrator checks that all its members are indeed investing their declared amount of computing power for mining in the said pool and any deviation from that would be reflected in the number of shares a member submits to the administrator or in the frequency of submission. This incurs an additional overhead on the pool administrator. So, the administrator may want to keep this overhead constant. So, she might want to keep the mining power of the entire pool unaltered against such small deviations of computing caused due to the exit of one or two miners. Someone can argue the rationale behind such an assumption as it might appear unrealistic since mining pools may not have idle hash power. The authors emphasize that the assumption of constant computing power made here is primarily intended for making the analysis simpler and not for presenting an analysis on a overly realistic instance. In section V-B, we analyze the scenario where the mining power of the pool is not constant and varies as different miner pour in or exit the pool. Hence, the reader may consider this section as a stepping stone for section V-B.

Theorem 2. *Let $\mathcal{A}$ be an attacker who uses e fraction of her computing power in attacking both the pools P and P' i.e. $\beta+\delta = e$. Then there exists a unique value $\beta_0 \in (0,e)$ such that the attacker's gain G is maximized for $\beta = \beta_0$ given the following condition:*

$$\gamma p'^2 + \frac{2\alpha e}{1-\alpha e} > \frac{\gamma p p'}{(1-\alpha e)^2}.$$

Proof. The total gain of the attacker $G = \frac{\alpha-\alpha\beta-\alpha\delta}{1-\alpha\beta-\alpha\delta} + \frac{p\alpha\beta-\alpha^2\beta^2}{p(1-\alpha\beta-\alpha\delta)} + \frac{p'\alpha\delta-\alpha^2\delta^2}{p'(1-\alpha\beta-\alpha\delta)} + \frac{\alpha\beta p'\gamma}{1-\alpha\beta} + \frac{\alpha\delta p\gamma}{1-\alpha\delta}$. Replacing δ by $e-\alpha$ we get, $G = \frac{\alpha(1-e)}{1-\alpha e} + \frac{p\alpha\beta-\alpha^2\beta^2}{p(1-\alpha e)} + \frac{p'\alpha(e-\beta)-\alpha^2(e-\beta)^2}{p'(1-\alpha e)} + \frac{\alpha\beta p'\gamma}{1-\alpha\beta} + \frac{\alpha(e-\beta)p\gamma}{1-\alpha(e-\beta)}$. Taking partial derivative w.r.t. β, $\frac{\partial G}{\partial\beta} = \frac{p\alpha-2\alpha^2\beta}{p(1-\alpha e)} + \frac{-p'\alpha+2\alpha^2(e-\beta)}{p'(1-\alpha e)} + \frac{\alpha\gamma p'}{(1-\alpha\beta)^2} - \frac{\alpha\gamma p}{(1-\alpha(e-\beta))^2}$

$$\left.\frac{\partial G}{\partial\beta}\right|_{\beta=0} = \frac{2\alpha^2 e}{p'(1-\alpha e)} + \alpha\gamma p' - \frac{\alpha\gamma p}{(1-\alpha e)^2}.$$

$$\left.\frac{\partial G}{\partial\beta}\right|_{\beta=e} = -\frac{2\alpha^2 e}{p(1-\alpha e)} + \frac{\alpha\gamma p'}{(1-\alpha e)^2} - \alpha\gamma p.$$

Now, since, $\gamma p'^2 + \frac{2\alpha e}{1-\alpha e} - \frac{\gamma}{(1-\alpha e)^2} p p' > 0$

$$\alpha\gamma p' + \frac{2\alpha^2 e}{p'(1-\alpha e)} - \frac{\alpha\gamma p}{(1-\alpha e)^2} > 0. \tag{5}$$

Since, P is the bigger pool, $p > p'$. So, $\gamma p^2 + \frac{2\alpha e}{1-\alpha e} - \frac{\gamma}{(1-\alpha e)^2} p p' > 0$
Hence,

$$\alpha\gamma p + \frac{2\alpha^2 e}{p(1-\alpha e)} - \frac{\alpha\gamma p'}{(1-\alpha e)^2} > 0. \tag{6}$$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TIFS.2016.2623588, IEEE Transactions on Information Forensics and Security

From equation 5 and 6, we get $\frac{\partial G}{\partial \beta}\big|_{\beta=0} > 0$ and $\frac{\partial G}{\partial \beta}\big|_{\beta=e} < 0$. So, there must be a $\beta_0 \in (0, e)$ such that G is maximum at $\beta = \beta_0$. Now, what is left to be proven is the fact that there is a unique $\beta_0 \in (0, e)$ such that $\frac{\partial G}{\partial \beta}\big|_{\beta=\beta_0} = 0$. Now, $\frac{\partial G}{\partial \beta} = \frac{p\alpha - 2\alpha^2\beta}{p(1-\alpha e)} + \frac{-p'\alpha + 2\alpha^2(e-\beta)}{p'(1-\alpha e)} + \frac{\alpha\gamma p'}{(1-\alpha\beta)^2} - \frac{\alpha\gamma p}{(1-\alpha(e-\beta))^2}$. So, $\frac{\partial^2 G}{\partial \beta^2} = -\frac{2\alpha^2}{p(1-\alpha e)} - \frac{2\alpha^2}{p'(1-\alpha e)} + \frac{2\alpha^2\gamma p'}{(1-\alpha\beta)^3} - \frac{2\alpha^2\gamma p}{(1-\alpha(e-\beta))^3}$. Now, $1 - \alpha\beta > p > p'$. So, $(1-\alpha\beta)^2 > pp' > \gamma pp'$. Also $(1-\alpha\beta) > (1-\alpha e)$. Hence, $(1-\alpha\beta)^3 > \gamma pp'(1-\alpha e)$. So, $\frac{2\alpha^2}{p(1-\alpha e)} > \frac{2\alpha^2\gamma p'}{(1-\alpha\beta)^3}$.

From this we conclude that $\frac{\partial^2 G}{\partial \beta^2} < 0$. Hence, $\frac{\partial G}{\partial \beta}$ is monotonically decreasing function on $\beta \in (0, e)$. So, there cannot be more than one $\beta_0 \in (0, e)$ such that $\frac{\partial G}{\partial \beta}\big|_{\beta=\beta_0} = 0$.

□

In order to find β_0 that maximizes the gain G of the attacker one can solve $\frac{\partial G}{\partial \beta} = 0$. So, $\frac{p\alpha - 2\alpha^2\beta_0}{p(1-\alpha e)} + \frac{-p'\alpha + 2\alpha^2(e-\beta_0)}{p'(1-\alpha e)} + \frac{\alpha\gamma p'}{(1-\alpha\beta_0)^2} - \frac{\alpha\gamma p}{(1-\alpha(e-\beta_0))^2}$

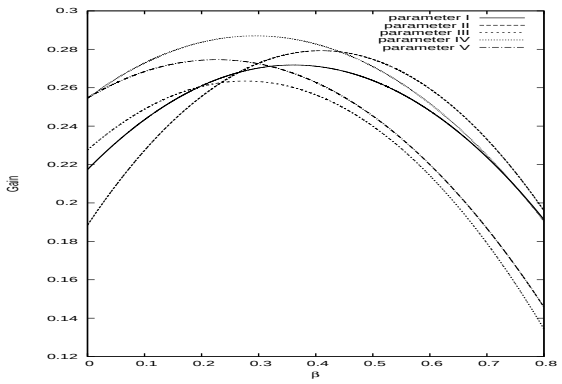

Fig. 2: Graphical presentation of the gain of the attacker who attacks both the pool and receives extra reward from both the pool for attacking each of them. The parameters corresponding to each of the graph can be found in Table III.

Lemma 2 proves that when the attacker has a fixed amount of power invested for carrying out sponsored BWH attack on both the pool, she can split her power intelligently for attacking both the pool such that her expected gain from mining is optimized. In other words **if the attacker does private mining with a fixed fraction of her mining power and uses the rest of her power in attacking the pools P and P', she could split her attacking power such that her gain from attacking both the pools is maximized.** We have plotted the curves corresponding to the attacker's gain in Figure 2, when her attacking power $\alpha(\beta + \delta)$ is fixed. It is easy to see that her gain peaks at some value of β for every values of other parameters. The curve of gain starts bending downward at the peak value. So, if β is further increased beyond a particular value the gain reduces. Thus, it is evident that the Figure 2 corroborates to the result of Theorem 2.

Now, we prove another property of the gain of the attacker. Here, we prove that the attacker can gain more by using her computing power than mining independently when the parameters meet certain constraint. In such cases the attacker can forget about private mining and use all her mining power for launching attack.

Theorem 3. *Let, $\frac{\gamma p'}{(1-\alpha+\alpha\delta)^2} + \frac{\alpha - k\alpha^2}{(1-\alpha)^2} > \frac{2\alpha(1-\delta)}{p(1-\alpha)}$ where $p' \geq \frac{1}{k}$. Then the attacker's best strategy will be to invest all her computing power for attacking rather than mining privately.*

Proof. The total gain of the attacker $G = \frac{\alpha - \alpha\beta - \alpha\delta}{1-\alpha\beta-\alpha\delta} + \frac{p\alpha\beta - \alpha^2\beta^2}{p(1-\alpha\beta-\alpha\delta)} + \frac{p'\alpha\delta - \alpha^2\delta^2}{p'(1-\alpha\beta-\alpha\delta)} + \frac{\alpha\beta p'\gamma}{1-\alpha\beta} + \frac{\alpha\delta p\gamma}{1-\alpha\delta}$. $\frac{\partial G}{\partial \beta}\big|_{\beta=0} = \frac{\alpha^2(1-\delta)}{(1-\alpha\delta)^2} + \frac{\alpha^2\delta(p'-\alpha\delta)}{p'(1-\alpha\delta)^2} + \alpha p'\gamma > 0$. Hence, G is increasing function of β at $\beta = 0$.

Now, the maximum value β could take is $1 - \delta$.

So, $\frac{\partial G}{\partial \beta}\big|_{\beta=1-\delta} = \frac{\alpha^2(1-\delta)(p-\alpha(1-\delta))}{p(1-\alpha)^2} + \frac{\alpha^2\delta(p'-\alpha\delta)}{p'(1-\alpha)^2} + \frac{\alpha p'\gamma}{(1-\alpha+\alpha\delta)^2} - \frac{2\alpha^2(1-\delta)}{p(1-\alpha)} = \frac{\alpha^2}{(1-\alpha)^2} - \frac{\alpha^3}{(1-\alpha)^2}(\frac{(1-\delta)^2}{p} + \frac{\delta^2}{p'}) + \frac{\alpha p'\gamma}{(1-\alpha+\alpha\delta)^2} - \frac{2\alpha^2(1-\delta)}{p(1-\alpha)}$.

Now, if $\frac{\gamma p'}{(1-\alpha+\alpha\delta)^2} + \frac{\alpha - k\alpha^2}{(1-\alpha)^2} > \frac{2\alpha(1-\delta)}{p(1-\alpha)}$, $\frac{\alpha\gamma p'}{(1-\alpha+\alpha\delta)^2} + \frac{\alpha^2}{(1-\alpha)^2} - \frac{k\alpha^3}{(1-\alpha)^2} - \frac{2\alpha^2(1-\delta)}{p(1-\alpha)} > 0$.

Now, $\frac{(1-\delta)^2}{p} + \frac{\delta^2}{p'} < \frac{(1-\delta)^2}{p'} + \frac{\delta^2}{p'} < \frac{(1-\delta)^2 + \delta^2}{p'} \leq \frac{1}{p'} \leq k$, since $p' \geq \frac{1}{k}$.

Thus, $\frac{\partial G}{\partial \beta}\big|_{\beta=1-\delta} > 0$.

Now, in order to prove that G is always increasing on $\beta \in (0, 1-\delta)$, we need to prove that G does not reach a maxima somewhere in the interval $(0, 1-\delta)$. To prove this fact we first assume that G reaches a maxima in the interval $(0, 1-\delta)$. Note that if G indeed reaches a maxima at $\beta = \beta_1, \beta_1 \in (0, 1-\delta)$ and as $\frac{\partial G}{\partial \beta}\big|_{\beta=\beta_0} > 0$, there will be a distinct value β_2 in $(0, 1-\delta)$ such that $\frac{\partial G}{\partial \beta}\big|_{\beta=\beta_2} = 0$ and $\frac{\partial G}{\partial \beta}\big|_{\beta\in(\beta_2, 1-\delta)} > 0$. So, $\frac{\partial^2 G}{\partial \beta^2}\big|_{\beta=\beta_2}$ must be greater than 0.

Therefore, $\frac{\partial^2 G}{\partial \beta^2} = \frac{2\alpha}{1-\alpha\beta-\alpha\delta}\frac{\partial G}{\partial \beta} - \frac{2\alpha^2}{p(1-\alpha\beta-\alpha\delta)} - \frac{2\alpha^3\gamma p'\delta}{(1-\alpha\beta)^3(1-\alpha\beta-\alpha\delta)}$.

Now, $\frac{\partial^2 G}{\partial \beta^2}\big|_{\beta=\beta_2} = \frac{2\alpha}{1-\alpha\beta-\alpha\delta}\frac{\partial G}{\partial \beta}\big|_{\beta=\beta_2} - \frac{2\alpha^2}{p(1-\alpha\beta-\alpha\delta)} - \frac{2\alpha^3\gamma p'\delta}{(1-\alpha\beta)^3(1-\alpha\beta-\alpha\delta)}$. Since, $\frac{\partial G}{\partial \beta}\big|_{\beta=\beta_2} = 0$, $\frac{\partial^2 G}{\partial \beta^2}\big|_{\beta=\beta_2} < 0$. This is a contradiction. So, we conclude that our assumption was wrong. Thus, G does not reach a maxima in the interval $(0, 1-\delta)$. So, G is always increasing on β as δ remains constant. Hence, the result. □

Corollary 4. *If $\frac{\gamma p}{(1-\alpha+\alpha\beta)^2} + \frac{\alpha - k\alpha^2}{(1-\alpha)^2} > \frac{2\alpha(1-\beta)}{p'(1-\alpha)}$, where $p \geq \frac{1}{k}$, G increases as δ increases while β remains constant.*

Remark 1. *If the condition of Theorem 3 is satisfied, the attacker's best strategy would be to utilize all her computing power for attacking the two pools. Figure 3 shows a graphical plot of the attacker's gain when β and δ varies and for various values of α, p and p'. It is easy to see that the gain maximizes when $\beta + \delta = 1$. This is a visual conformity of the result of lemma 3.*

Table IV shows that the condition $\frac{\gamma p'}{(1-\alpha+\alpha\delta)^2} + \frac{\alpha - k\alpha^2}{(1-\alpha)^2} > \frac{2\alpha(1-\delta)}{p(1-\alpha)}$ where $p' \geq \frac{1}{k}$ is indeed achievable. We have showed this for few values of the set of parameters that the condition is feasible when the parameters take practical values.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TIFS.2016.2623588, IEEE Transactions on Information Forensics and Security

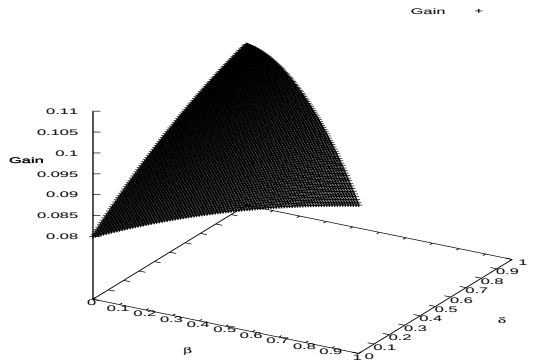

Fig. 3: Graphical presentation of the gain of the attacker who attacks both the pool and receives extra reward from both the pool for attacking each of them for as β and δ varies.

B. When the computing power of the pools are not constant

Here, we study the the characteristic of the gain-function of the attacker when the computing power varies with the computing power invested by the attacker into the pools. Let, p and p' be the computing powers of the pool P and P' excluding the contribution of the attacker. Since, the attacker expends $\alpha\beta$ computing power for attacking pool P, the total computing power of P becomes equal to $p + \alpha\beta$ where $\alpha\beta$ is the computing power intended for sponsored BWH attack. Similarly, the total computing power of P' becomes $p' + \alpha\delta$ where $\alpha\delta$ is the contribution of the attacker. So, now the gain of the attacker is given by,

$$G = \frac{\alpha - \alpha\beta - \alpha\delta}{1 - \alpha\beta - \alpha\delta} + \frac{p\alpha\beta}{(p + \alpha\beta)(1 - \alpha\beta - \alpha\delta)} + \frac{p'\alpha\delta}{(p' + \alpha\delta)(1 - \alpha\beta - \alpha\delta)} + \frac{\alpha\beta(p' + \alpha\delta)\gamma}{1 - \alpha\beta} + \frac{\alpha\delta(p + \alpha\beta)\gamma}{1 - \alpha\delta} \quad (7)$$

Theorem 5. *If the attacker expends a constant amount of her computing power to attack both the pools i.e. if $\beta + \delta = e$, the attacker's gain is maximized for some $\beta \in \{0, e\}$ provided $\frac{\alpha}{1-\alpha e} > \frac{\alpha p^2}{(p+\alpha e)^2(1-\alpha e)} + \frac{\alpha\gamma(p+\alpha^2e^2-\alpha e)}{(1-\alpha e)^2} - \alpha\gamma(p' + \alpha e)$.*

Proof. Since, $\beta + \delta = e$, the gain of the attacker is given by

$G = \frac{\alpha-\alpha e}{1-\alpha e} + \frac{p\alpha\beta}{(p+\alpha\beta)(1-\alpha e)} + \frac{p'\alpha\delta}{(p'+\alpha\delta)(1-\alpha e)} + \frac{\alpha\beta(p'+\alpha\delta)\gamma}{1-\alpha\beta} + \frac{\alpha\delta(p+\alpha\beta)\gamma}{1-\alpha\delta}$.

$\frac{\partial G}{\partial \beta}\big|_{\beta=0} = \frac{\alpha}{1-\alpha e} - \frac{\alpha p'^2}{(\alpha e+p')^2(1-\alpha e)} + \alpha\gamma(p' + \alpha e) - \frac{\alpha\gamma(p+\alpha^2e^2-\alpha e)}{(1-\alpha e)^2}$.

$\frac{\partial G}{\partial \beta}\big|_{\beta=e} = \frac{\alpha p^2}{(p+\alpha e)^2(1-\alpha e)} - \frac{\alpha}{1-\alpha e} + \frac{\alpha\gamma(p'+\alpha^2e^2-\alpha e)}{(1-\alpha e)^2} - \alpha\gamma(p + \alpha e)$.

We can rewrite these equations as:

$\frac{\partial G}{\partial \beta}\big|_{\beta=0} = \frac{\alpha}{1-\alpha e} - \Big(\frac{\alpha p'^2}{(p'+\alpha e)^2(1-\alpha e)} + \frac{\alpha\gamma(p+\alpha^2e^2-\alpha e)}{(1-\alpha e)^2} - \alpha\gamma(p' + \alpha e)\Big)$.

$\frac{\partial G}{\partial \beta}\big|_{\beta=e} = \Big(\frac{\alpha p^2}{(p+\alpha e)^2(1-\alpha e)} + \frac{\alpha\gamma(p'+\alpha^2e^2-\alpha e)}{(1-\alpha e)^2} - \alpha\gamma(p + \alpha e)\Big) - \frac{\alpha}{1-\alpha e}$.

Since, $\frac{\alpha}{1-\alpha e} > \frac{\alpha p^2}{(p+\alpha e)^2(1-\alpha e)} + \frac{\alpha\gamma(p+\alpha^2e^2-\alpha e)}{(1-\alpha e)^2} - \alpha\gamma(p' + \alpha e)$ and $p > p'$, $\frac{\alpha}{1-\alpha e} - \Big(\frac{\alpha p'^2}{(p'+\alpha e)^2(1-\alpha e)} + \frac{\alpha\gamma(p+\alpha^2e^2-\alpha e)}{(1-\alpha e)^2} - \alpha\gamma(p' + \alpha e)\Big) > 0$.

So, $\frac{\partial G}{\partial \beta}\big|_{\beta=0} > 0$.

Again, $\frac{\alpha}{1-\alpha e} > \frac{\alpha p^2}{(p+\alpha e)^2(1-\alpha e)} + \frac{\alpha\gamma(p'+\alpha^2e^2-\alpha e)}{(1-\alpha e)^2} - \alpha\gamma(p + \alpha e)$.

Hence, $\frac{\partial G}{\partial \beta}\big|_{\beta=e} < 0$.

Since, $\frac{\partial G}{\partial \beta}\big|_{\beta=0} > 0 > \frac{\partial G}{\partial \beta}\big|_{\beta=e}$, we conclude that there is a $\beta_{max} \in (0, e)$, such that $\frac{\partial G}{\partial \beta}\big|_{\beta=\beta_{max}} = 0$. Now in order to prove that β_{max} maximizes G, we need to show that there is a unique $\beta_{max} \in (0, e)$ such that $\frac{\partial G}{\partial \beta}\big|_{\beta=\beta_{max}} = 0$ and $\frac{\partial G}{\partial \beta}\big|_{\beta=\beta_{max}} < 0$, $\forall \beta \in (\beta_{max}, e)$.

In order to show this we take partial derivative of $\frac{\partial G}{\partial \beta}$ with respect to β and see

$\frac{\partial^2 G}{\partial \beta^2} = -\frac{2\alpha^2p^2}{(1-\alpha e)(p+\alpha\beta)^3} - \frac{2\alpha^2p'^2}{(p'+\alpha(e-\beta))^3(1-\alpha e)} - \frac{2\alpha^2\gamma(1-p-\alpha e)}{(1-\alpha\beta)^3} - \frac{2\alpha^2\gamma(1-p-\alpha e)}{(1-\alpha(e-\beta))^3} < 0$.

Therefore, $\frac{\partial G}{\partial \beta}$ always decreases.

Hence, there could be at most one $\beta_{max} \in (0, e)$ such that $\frac{\partial G}{\partial \beta}\big|_{\beta=\beta_{max}} = 0$ and $\frac{\partial G}{\partial \beta} < 0$, $\forall \beta \in (\beta_{max}, e)$.

□

Remark 2. *Table V shows that the condition $\frac{\alpha}{1-\alpha e} > \frac{\alpha p^2}{(p+\alpha e)^2(1-\alpha e)} + \frac{\alpha\gamma(p+\alpha^2e^2-\alpha e)}{(1-\alpha e)^2} - \alpha\gamma(p' + \alpha e)$ is indeed achievable. We have showed this for few values of the set of parameters that the condition is feasible when the parameters take practical values.*

VI. Countering Block Withholding Attack

As we discussed earlier, block withholding attack could be disastrous for any open mining pool that allows untrusted miners to join pool anytime. As of now, there has been no strong defense against this type of attack. Rosenfeld [5] discussed a technique to generate miner tasks that would lead to a block solution. This technique uses Honeypots for luring rogue miners into a trap. An attacker who withholds such a block solution would be caught by the pool. However, in this scheme, the pool needs to ask its miners to expend their computing power for doing useless computation wasting their valuable resources. Schrijvers et al. [11] proposed to counter BWH attack by making the reward strategy incentive compatible. Bag et al. [12] proposed to give extra reward to the discoverer of the winning block. The difference between the Bag et al. scheme and the Schrijvers et al. scheme is that in Bag et al. scheme the discoverer of the winning block always gets extra incentives as opposed to the later one where the discoverer gets extra incentive only when the miners delay the submission of the pool shares. That is, in Schrijvers et al. scheme the discoverer of the winning block gets more reward only if some pool shares are not submitted to the administrator. In this section we propose a simple tweak to the existing mining scheme that would make block withholding attack implausible for both a regular miner and the pool administrator. We in section VI-A, propose a generic scheme that makes use of cryptographic commitment scheme. Then in section VI-B show how the

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TIFS.2016.2623588, IEEE Transactions on Information Forensics and Security

same scheme can be implemented using hash function which is itself a commitment scheme in a sense. However, we need to perform minor variation to the original scheme for making this to work. Our scheme counter BWH attack by precluding the miners from being able to distinguish between a partial proof of work and a full proof of work.

A. Scheme I

In the existing Bitcoin mining scheme, a miner needs to compute a nonce n and a Bitcoin block containing the nonce and transactions such that the hash of the block header contains some z leading '0' bits. We call it the 'target' of the proof of work. So, the target is a bit string of length z. In the current Bitcoin mining protocol target is a string of z number of 0 bits. So, a Bitcoin block which is correctly constructed and whose SHA-256 hash outputs a string starting with z number of '0' bits is a winning block that can be used to earn 25BTC from the Bitcoin system. The partial proofs of work submitted regularly by the pool members to the pool administrator for evaluation of work have a target bit string of length z' where $z' \leq z$. The value of z' is decided by the pool administrator and should be such that the pool members can regularly find partial proofs but the verification of these partial proofs does not incur an overwhelming computational overhead on the pool administrator. We modify the current scheme and propose a new one where the pool itself can partially choose the target. Our proposed scheme works as follows:

- In our mining scheme, we, like the existing scheme let the pool administrator decide the number of '0's required in the partial proof of work to be submitted by the individual miners of the pool to the administrator. Let this be z' where $z' \leq z$. A partial proof of work is a block whose hash starts with z' zero bits.
- The mining pool also selects the $z - z'$ of least significant bits of the target of the full proof of work. Let, this be $r, r \in \{0,1\}^{z-z'}$. The pool also chooses a random string $s \in \{0,1\}^*$, where $|s| + |r| = \kappa$, where κ is the security parameter of the commitment scheme that is defined by the Bitcoin system and not by the mining pool itself. Here the role of s is to make the length of the combined nonce $r||s$ to be equal to κ, which is the prescribed length for the commitment scheme. The value of the security parameter should be such that breaking the commitment scheme should be harder than finding a full proof of work. In general we can use 256 as the security parameter for our scheme. The main difference between our scheme and the current mining scheme is that in our scheme, the pool decides the trailing $z-z'$ bits of the 'target' and it can be of the form $\{0,1\}^{z-z'}$.
- The final target of the full proof of work for this pool thus becomes $0^{z'}||r$ which is of length z and is same as the length of target of the Bitcoin mining scheme at present.
- The mining pool generates a commitment $(com, decom) \longleftarrow Commit(r||s, \mathcal{CK})$. The pool sends the commitment to all the miners and keeps the opening value $decom$ secret for the time being.
- All the miners of that pool include the commitment com and z' in the proofs of work they compute. So, when the miner constructs a block, the commitment com and the length of the target of the partial proof is also included in the block. The partial proofs include all those blocks whose hash start with $0^{z'}$. So, a miner submits all blocks having hashes equal to $0^{z'}||\{0,1\}^*$.
- The pool administrator receives as partial proofs all those blocks whose hash start with $0^{z'}$. Now, the pool administrator scrutinizes these partial proofs until she finds a full proof that has a hash value starting with $0^{z'}||r$. If she finds such a block then she submits the block to the Bitcoin system along with the opening value of the commitment $decom$.
- When the nodes of the Bitcoin network receive the block and the opening value $decom$, they extract the commitment value com and z' contained in the block and they extract $r||s \longleftarrow Open(com, decom)$. The block is accepted if the first z bits of the hash of the block comes out to be $0^{z'}||r$, where r is the leading $z - z'$ bits of $Open(com, decom)$. If there is any discrepancy, the block is rejected. Note that, the value of z' may be different for separate pools. However, a Bitcoin node can find the value of the same for a particular mining pool in the block submitted by the pool. If the hash of the block starts with a '1', then the value of z' for the block must be 0.

Security Property of the proposed scheme: Here, we show that our scheme is secure against block withholding attack. We also show that this scheme does not provide leeway to any mining pool for solving the mining puzzle by performing less amount of computation that what they are expected to i.e. the mining scheme does not make it easy for the mining pool to win the mining game. The hardness of our scheme is same as the existing Bitcoin mining scheme and the miners don't get any advantage over the existing mining scheme under the proposed scheme.

It is easy to see that proposed scheme is immune against BWH attack launched by miners as the miners only know the target of the partial proof of work ($0^{z'}$) and are not allowed to know the rest of the $z-z'$ bits (r) that the full proof of work consists until the mining job is refreshed due to the emergence of a Bitcoin block constructed by another mining pool that contains one or more transactions from the set of transactions this pool has been mining on. The miners only learn the commitment com to the combined nonce $r||s$ and the first few bits (r) cannot be known from the commitment com due to the hiding property of the commitment scheme. So, the miner cannot distinguish between a partial proof of work and a full proof of work. Thus, she cannot withhold a valid solution. We prove the security of this scheme in Lemma 3 and 4.

Lemma 3. *The proposed mining ensures that a mining pool cannot find a proof of work by doing less amount of computation than what it should have.*

Proof. If the mining pool intends to deceive the proposed mining scheme, it should do the following; Let us assume that the mining pool chooses two bit strings r and s such that $|r| + |s| = \kappa$ and then calculates the commitment

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TIFS.2016.2623588, IEEE Transactions on Information Forensics and Security

$com \leftarrow Commit(r||s, \mathcal{CK})$. The mining pool then needs to find a block whose hash starts with the bit string $0^{z'}||r$. Since $|r| = z - z'$, the probability of getting such a block in every random attempt is $\frac{1}{2^z}$ which is exactly the same as the current Bitcoin mining scheme.

Alternatively, the pool can choose an arbitrary p as the commitment. Create a block B having a hash value whose first z bits are given by $0^{z'}||r'$. Then the mining pool's only job will be to find a string $s' \in \{0,1\}^{\kappa-|r'|}$, such that $p = Commit(r'||s', \mathcal{CK})$. So, if a mining pool can do so, it would break the binding property of the commitment protocol as there is only a single value $r||s$ such that $p \leftarrow Commit(r||s, \mathcal{CK})$ and the probability $Pr\left[r'||s' = r||s\right] = \frac{1}{2^\kappa}$. We can choose a security parameter κ that can ward off the possibility of breaking the binding property of the commitment protocol.

Again, the mining pool can select two nonce r and s, such that $|r| + |s| = \kappa$. Now, it can compute $p = Commit(r||s, \mathcal{CK})$. Then the pool can create a block B such that the first z bits of the hash of B happen to be $\{0\}^{z'}||r', r' \neq r$. Then all the pool needs to do is to compute a nonce s', $|s'| = \kappa - |r'|$ such that $p = Commit(r'||s', \mathcal{CK})$. But this would contradict the binding property of the commitment scheme.

□

Lemma 4. *The proposed mining scheme is secure against BWH attack lodged by the pool administrator.*

Proof. We assume the pool administrator publicly reveals the decommitment d (say) to the combined nonce $r||s$ every time it changes the transaction set. Let p be the commitment to the combined nonce $r||s$. All the miners include p in the proof of work they compute. Whenever the pool administrator changes the transaction set, she reveals the decommitment d to all the miners. Thereafter, the miners can compute $r||s = Open(p, d)$ and can compute the target $0^{z'}||r$, where $|r| = z - z'$. Thus, a miner can detect if she had indeed computed a full proof of work or not. □

B. Scheme II

We now provide an alternate implementation of the above scheme. Here, we shall be using hash function instead of the commitment scheme proposed above. In fact the usage of hash function is the only difference between scheme I and scheme II, the main idea being the same in both the schemes. In this scheme too we shall be dedicating a special field in the block that would hold a string directly linking to the nonce that specify the rightmost few bits of the target. This scheme works as follows:

- The mining pool P selects a random bitstring $r, |r| = z - z'$. The pool also selects another random string $s \in \{0,1\}^*$. Let, $p = H(r||s)$. Note, that if we don not have s, it would be easy to find r using brute force. Here the purpose of using s is to make it hard for the attackers to find the value of r using brute force search.
- The pool makes p and z' public. Now, all constituent members of the pool include p and z' into the header of a block while mining it exactly like scheme I. That is they try to find a block that includes p and z' as a new field of the header of the block(Just like Figure 4). We term this field as 'blockpad'.
- The pool receives pool shares (partial proofs) from its members such that the block submitted by the member contains some z' zero bits. The pool computes hashes of all these blocks until it finds a block with a hash value $0^{z'}||r||\{0,1\}^*$. If the pool finds such a block, she broadcasts it along with $r||s$.
- After a block is sent by a mining pool along with the combined nonce $r||s$, every Bitcoin node checks if the hash of the pad ($H(r||s)$) is included in the block and that the first $z - z'$ bits of the combined nonce are same as r, where the hash of the block is $0^{z'}||r||\{0,1\}^*$. If this check is successful, then the block is accepted and added to the block chain.

Thus, we make the Bitcoin mining scheme immune to BWH attack by letting the pool administrator decide the last $z - z'$ bits in the challenge. Since every partial pool share needs to have z' leading zero bits, no Bitcoin miner can distinguish between a block with a partial PoW and the same with a full PoW. Thus she cannot initiate block withholding attack against the pool. Also, since the pool has to include p in the block header, there is no way it can diminish the amount of computation needed to solve the puzzle. The amount of computation needed to solve the puzzle in our scheme is same as that of the original Bitcoin mining scheme. We formally show it in Lemma 5.

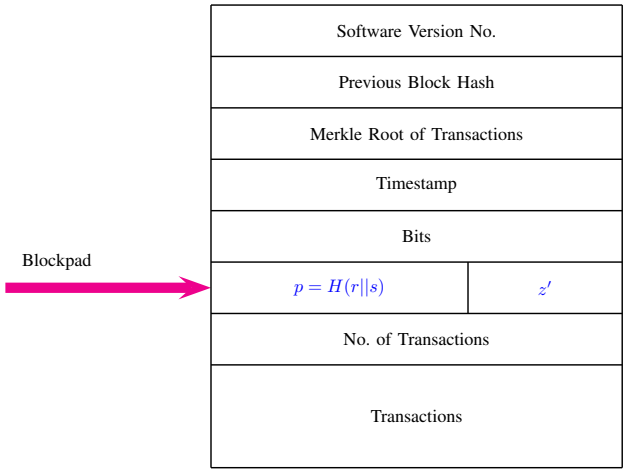

Fig. 4: New scheme for mining

Lemma 5. *The proposed scheme II makes the Bitcoin mining in pool immune against BWH attack as long as the combined nonce $r||s$ is kept secret.*

Proof. Our scheme prevents block withholding attack if the pool administrator does not reveal r and s to the miners of the pool. The miners only get to know the value of $H(r||s)$. So, they cannot guess the value of r as long as s is kept secret. However if s is revealed to the miners, they can guess r by doing a brute force search over all possible values of r. Since $|r| = z - z'$ any miner can find r by performing computation of at most $2^{z-z'}$ hashes which is trivial for small values of $z - z'$. Hence, the pool

administrator must keep both r and s secret until a PoW solution is found and submitted. □

Lemma 6. *The scheme II does not reduce the amount of computation needed by a mining pool to compute a PoW.*

Proof. If the mining pool can cheat the scheme then it will have to compute nonce r, s such that $|r| = z - z', 0 \leq z' \leq z, |r| + |s| = \kappa$ and $p = H(r||s)$, where p is included in the block. Suppose, the mining pool has found a block B such that hash of B starts with z' zeros, where $0 \leq z' \leq z$. The probability of getting such a block in every attempt is $\frac{1}{2^{z'}}$. Let, the first $z - z'$ bits after $0^{z'}$ in the hash of B be r'. So, all the miners needs to do is to find a nonce s', $|r'| + |s'| = \kappa$ and $p = H(r'||s')$. We can choose a hash function having output length $|p|$ as large so that finding such a s' would require enormous amount of computation making it impractical for the miner. □

Just like the scheme I, scheme II too can be used to defeat BWH attack launched by the pool administrator. If the pool protocol makes it mandatory for the pool administrator to reveal the combined nonce $r||s$ every time the blockpad is updated. Thus, all miners will be able to verify whether or not they had indeed created a valid block with a full PoW. Note that whenever the transaction set is updated the blockpad has to be updated as well as the nonce are revealed to the constituent miners of the pool.

VII. Conclusion

In this paper we showed how a selfish miner could earn some extra incentive for launching BWH attack on a mining pool. This extra incentive comes from some other like minded mining pool that wants to benefit from this BWH attack. The latter mining pool shares a part of the incentive it indirectly earns from the attack. We quantitatively measured the the total gain that a BWH attacker could expect by following different attacking strategies. We also showed some interesting results that an attacker may use for increasing her total gain. We showed that in a scenario where there are only two mining pools, the attacker can divide her computing power to attack both the pool to maximize her gain. We have also presented a strategy to effectively deter BWH attack in any mining pool and have shown how the strategy can be implemented using hash function. The strategy is shown to counter BWH attack launched by both a constituent miner and the pool administrator. This strategy prevent BWH attack by blinding miners' ability to distinguish between the partial and the full proofs of work and at the same time binding the pool administrator to a commitment that would eventually allow all miners to verify if they had indeed found a full proof of work. The strategy is easily implementable using hash function or cryptographic commitment protocols without requiring to compromise with the amount of computation needed to find a PoW.

References

[1] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," 2008.

[2] A. Miller, A. Juels, E. Shi, B. Parno, and J. Katz, "Permacoin: Repurposing bitcoin work for data preservation," in *2014 IEEE Symposium on Security and Privacy, SP 2014, Berkeley, CA, USA, May 18-21, 2014*. IEEE Computer Society, 2014, pp. 475–490.

[3] B. Sengupta, S. Bag, S. Ruj, and K. Sakurai, "Retricoin: Bitcoin based on compact proofs of retrievability," in *Proceedings of the 17th International Conference on Distributed Computing and Networking*, ser. ICDCN '16, 2016, pp. 14:1–14:10.

[4] J. A. Kroll, I. C. Davey, and E. W. Felten, "The economics of bitcoin mining, or bitcoin in the presence of adversaries," *Proceedings of WEIS*, vol. 2013, 2013.

[5] M. Rosenfeld, "Analysis of bitcoin pooled mining reward systems," *CoRR*, vol. abs/1112.4980, 2011. [Online]. Available: http://arxiv.org/abs/1112.4980

[6] A. Laszka, B. Johnson, and J. Grossklags, *Financial Cryptography and Data Security: FC 2015 International Workshops, BITCOIN, WAHC, and Wearable, San Juan, Puerto Rico, January 30, 2015, Revised Selected Papers*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2015, ch. When Bitcoin Mining Pools Run Dry, pp. 63–77.

[7] I. Eyal, "The miner's dilemma," in *2015 IEEE Symposium on Security and Privacy, SP 2015, San Jose, CA, USA, May 17-21, 2015*. IEEE Computer Society, 2015, pp. 89–103.

[8] L. Luu, R. Saha, I. Parameshwaran, P. Saxena, and A. Hobor, "On power splitting games in distributed computation: The case of bitcoin pooled mining," in *IEEE 28th Computer Security Foundations Symposium, CSF 2015, Verona, Italy, 13-17 July, 2015*, C. Fournet, M. W. Hicks, and L. Viganò, Eds. IEEE Computer Society, 2015, pp. 397–411.

[9] N. T. Courtois and L. Bahack, "On subversive miner strategies and block withholding attack in bitcoin digital currency," *arXiv preprint arXiv:1402.1718*, 2014.

[10] I. Damgård, *Lectures on Data Security: Modern Cryptology in Theory and Practice*. Berlin, Heidelberg: Springer Berlin Heidelberg, 1999, ch. Commitment Schemes and Zero-Knowledge Protocols, pp. 63–86.

[11] D. B. Okke Schrijvers, Joseph Bonneau and T. Roughgarde, "Incentive compatibility of bitcoin mining pool reward functions," in *Financial Cryptography and Data Security: FC 2016 International Workshops*.

[12] S. Bag and K. Sakurai, "Yet another note on block withholding attack on bitcoin mining pools," in *Information Security - 19th International Conference, ISC 2016, Honolulu, HI, USA, September 3-6, 2016, Proceedings*, ser. Lecture Notes in Computer Science, M. Bishop and A. C. A. Nascimento, Eds., vol. 9866. Springer, 2016, pp. 167–180.

Appendix

Serial No.	α	$\beta + \delta$	γ	$\alpha\gamma p'^2 + \frac{2\alpha^2 e}{(1-\alpha e)} - \frac{\alpha\gamma p p'}{(1-\alpha e)^2}$
I	0.100	0.80	0.70	0.012051
II	0.110	0.90	0.70	0.014145
III	0.090	0.80	0.70	0.005832
IV	0.084	0.85	0.70	0.005860
V	0.080	0.80	0.70	0.003805

TABLE III: Table of parameters corresponding to different curves of figure 2

α	p	p'	γ	δ	k	$\frac{\gamma p'}{(1-\alpha+\alpha\delta)^2} + \frac{\alpha - k\alpha^2}{(1-\alpha)^2} - \frac{2\alpha(1-\delta)}{p(1-\alpha)}$
0.10	0.50	0.40	0.70	0.600	2.0	0.224807
0.12	0.60	0.28	0.70	0.600	3.0	0.133618
0.10	0.55	0.35	0.80	0.650	2.5	0.251858
0.13	0.45	0.42	0.50	0.600	3.0	0.072794
0.15	0.50	0.35	0.80	0.400	3.0	0.028781
0.13	0.50	0.37	0.77	0.770	4.0	0.247703
0.09	0.55	0.36	0.70	0.550	3.0	0.191223

TABLE IV: Table showing the feasibility of the conditions of Theorem 3

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TIFS.2016.2623588, IEEE Transactions on Information Forensics and Security

α	p	p'	γ	e	$\frac{\alpha}{1-\alpha e} - (\frac{\alpha p^2}{(p+\alpha e)^2(1-\alpha e)} + \frac{\alpha\gamma(p+\alpha^2e^2-\alpha e)}{(1-\alpha e)^2} - \alpha\gamma(p'+\alpha e))$
0.10	0.50	0.40	0.80	0.60	0.018212
0.12	0.55	0.33	0.70	0.70	0.019832
0.13	0.60	0.27	0.70	0.80	0.016287
0.14	0.50	0.36	0.77	0.60	0.034315
0.10	0.55	0.35	0.75	0.66	0.010928
0.13	0.60	0.27	0.60	0.70	0.014515
0.13	0.55	0.32	0.74	0.90	0.034046
0.09	0.50	0.41	0.70	0.90	0.024591

TABLE V: Table showing the feasibility of the conditions of Theorem 5

Samiran Bag earned a Master of Technology degree in Computer Science from the Indian Statistical Institute. He also did his Ph. D. from the Indian Statistical Institute. He has worked as a post doctoral researcher at the Kyushu University in Japan. Presently he is a research associate at the School of Computing Science, Newcastle University. His primary research areas are Electronic Voting, Cryptocurrency and Secure multi-party computation.

Sushmita Ruj received her B.E. degree in Computer Science from Bengal Engineering and Science University, Shibpur, India and Masters and Ph D in Computer Science from Indian Statistical Institute. She was an Erasmus Mundus Post Doctoral Fellow at Lund University, Sweden and Post Doctoral Fellow at University of Ottawa, Canada. She is currently an Assistant Professor at Indian Statistical Institute, India. Prior to this, she was an Assistant Professor at IIT, Indore. She was a visiting researcher at INRIA, France, University of Wollongong, Australia, Kyushu University, Japan. KDDI labs, Japan and Microsoft Research Labs, India.

"Her research interests are in applied cryptography, security, combinatorics and complex network analysis. She works actively in blockchain and cryptocurrencies, , mobile ad hoc networks, vehicular networks, cloud security, security in smart grids. She has served as program co-chair of IEEE ICCC (P&S Track), IEEE ICDCS, IEEE ICC, etc and served on many TPCs. She won best paper awards at ISPA07 and IEEE PIMRC11. Sushmita is a Senior Member of ACM and IEEE.

Kouichi Sakurai received the B.S. degree in mathematics from the Faculty of Science, Kyushu University in 1986. He received the M.S. degree in applied science in 1988, and the Doctorate in engineering in 1993 from the Faculty of Engineering, Kyushu University. He was engaged in research and development on cryptography and information security at the Computer and Information Systems Laboratory at Mitsubishi Electric Corporation from 1988 to 1994. From 1994, he worked for the Dept. of Computer Science of Kyushu University in the capacity of associate professor, and became a full professor there in 2002.