Mempool Optimization for Defending Against DDoS Attacks in PoW-based Blockchain Systems

Muhammad Saad
University of Central Florida

Laurent Njilla
Air Force Research Laboratory

Charles Kamhoua
Army Research Laboratory

Joongheon Kim
Chung-Ang University

DaeHun Nyang
Inha University

Aziz Mohaisen
University of Central Florida

***Abstract*—In this paper, we present a new form of attack that can be carried out on the memory pools (mempools) of blockchain-based cryptocurrencies. Towards that end, we study such an attack on Bitcoin mempool and explore its effects on transactions fee paid by legitimate users. We also propose countermeasures to contain such an attack. Our countermeasures include fee-based and age-based designs, which optimize the mempool size and help in countering the effects of DDoS attacks. We further evaluate our designs by simulations and analyze their usefulness in varying attack conditions. Our analyses can be extended to other blockchain-based applications which use memory pools to cache network activities.**

I. Introduction

In blockchain-based cryptocurrencies, DDoS attacks are launched on blockchain, miners, users, and currency exchanges [1]. A classical form of DDoS attack on Bitcoin exploits the block-size limit and the network throughput to prevent legitimate users from getting their transactions mined. In Bitcoin, the block size is limited to 1MB and the average block publishing time is 10 minutes. The size of individual transaction varies from 200–1K Bytes. Under these constraints, Bitcoin can only verify 3-7 transactions per second [2], [3]. Low transaction throughput creates a competitive environment where only selected transactions are mined. This makes Bitcoin vulnerable to flood attacks [4], where malicious users exploit the block size limit in Bitcoin (1MB) to overwhelm the blockchain with low-valued spam transactions, and cause delay in the verification of legitimate transactions. To prevent such attacks, miners apply priority checks on transactions, and prioritize the ones that pay higher mining fee.

In cryptocurrencies, the memory pool (mempool) acts as a repository for unconfirmed transactions. Once a user generates a transaction, it is broadcast to the network, and stored in mempools of nodes. If the rate of incoming transactions at the mempool is less than the network throughput (3-7 transactions/sec), there is no queue of unconfirmed transactions. Once the rate exceeds the throughput, a transaction backlog develops and the transactions that remain unconfirmed for long are eventually rejected. On November 11, 2017, the mempool exceeded 115k transactions, resulting in USD 700 million worth transaction stall [5]. As the mempool size grows, users naturally pay higher mining fee to prioritize their transactions.

In this paper, we identify mempool flooding attack that causes denial-of-service for legitimate users. We establish a relationship between the mempool size and transaction fee, and demonstrate how attackers can use it to make legitimate users pay higher than the normal fee. Although miners discard spam transactions during the mining process, there is no effective mechanism to prevent the mempool DDoS attack.

Contributions. In summary, we make the following contributions. 1) First, we identify the effect of mempool flooding on legitimate users in Bitcoin and the way it shapes into a DoS attack. 2) We present the threat model and the attack procedure that enable an adversary to exploit current protocols of the system. 3) We propose effective countermeasures at the mempool level, including fee-based and age-based designs, that optimize mempool size and prevent attacks. 4) We test the performance of our proposed countermeasures through discrete-event simulations.

Organization. In §II we outline the preliminaries of this work, including the operations of cryptocurrencies, DDoS attack on mempools and data collection for this study. In §III and §III-A we describe the threat model attack procedure that lead to mempool flooding and rise in the mining fee. We propose countermeasures in §V. Experimental results are reported in §VI. In §VII we review the related work. Conclusion and future work are presented in §VIII.

II. Preliminaries

UTXO. In Bitcoin, a user generates a transaction by using spendable balance in his wallet. Spendable balance comprises of confirmed "Unspent Transaction Outputs" (UTXO's) [6] that are previously mined in the blockchain.

Relay Fee and Mining Fee. Relay fee in Bitcoin is the minimum fee paid for a transaction to be included in a mempool. If a transaction does not include the relay fee, peers do not forward the transaction to other peers. Mining fee is the fee paid to a miner as an incentive to include the transaction into a block [7]. Miners tend to prioritize the transactions which pay higher mining fee.

Confirmation. Transaction confirmation means that a transaction has been mined into a block and its parent UTXO's are valid and spendable in receiver's wallet [7]. A confirmation

978-1-7281-1328-9/19/$31.00 2019 IEEE

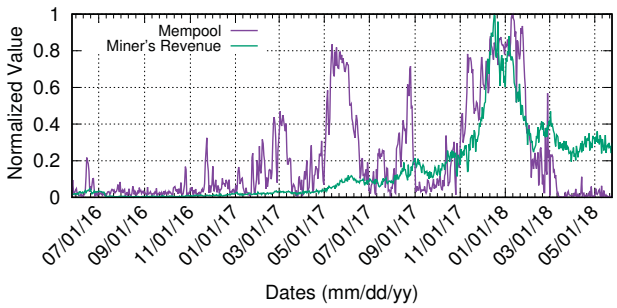


Fig. 1. Temporal study of mempool size and mining fee paid by the users. Notice that as the mempool size grows, the mining fee increases accordingly. The spikes during May, September, and November indicate spam attacks.

score, also known as the age of a transaction, is the difference between the block number in which it was mined and the most recent block. A confirmation score of 0 means that the transaction is in the mempool, and not yet mined. Such a transactions is also called an "unconfirmed transaction."

Memory Pool. In cryptocurrencies, a memory pool (mempool) acts as a cache of unconfirmed transactions [7]. The mempool is a bottleneck in the system, and if the transactions' rate of arrival exceeds their mining rate, the mempool size starts to grow and the verification process gets delayed.

Dust Transactions. In cryptocurrencies, transactions with small input values are known as "dust transactions" [8]. Dust transactions contribute very little to the exchange volume of Bitcoin but consume as much space in the block as a high valued transaction. Spam attacks to exhaust the block space are carried out using these transactions [4].

A. DDoS Attack on Mempools

There are two types of DDoS attacks on blockchain-based cryptocurrencies. In the classical attack, attackers exploit the size limitation of blocks (1MB in Bitcoin) and generate dust transactions to occupy the space and prevent other transactions from mining. This attack has been addressed by the research community [4], and there are countermeasures adopted by the miners to prevent it. The other form of DDoS attack targets mempools by choking them with a flood of unconfirmed transactions. Although, these transactions may eventually be rejected by miners, but their presence in the mempool creates another major problem. The mempool size determines the fee paid to the miners. If the mempool size is big, miners have a limited choice of mining the transactions, and the users try to prioritize their transactions by paying higher mining fee. Therefore, by mempool flooding, the attacker might trap the users into paying high fee. In Figure 1, we show the relationship between the mempool size and the fee paid to the miners. Notice a high correlation between them.

B. Data Collection

To observe the relationship between the mempool size and the mining fee, we used the public dataset provided by the company called "Blockchain" [9]. For this study, we gathered the dataset of mempool size and fee from January 2016 to May 2018. In Figure 1, we plot the results obtained from the dataset and we use the min-max normalization to scale our dataset in the range [0–1]. Our data shows that Bitcoin mempool has been attacked three times in 2017, and each time it resulted in an unfair increase of the mining fee.

III. Threat Model

For our threat model, we assume an attacker with spendable balance in this wallet. The attacker controls a group of sybil accounts, each with multiple public addresses, intended to be used during the attack. Furthermore, the attacker and sybils have client side software and scripts, which enable them to initiate a flood of "raw transactions" [7], higher rate than the network throughput [2]. Additionally, the attacker is also constrained by a fixed "budget." Since each transaction requires a minimum relay fee, it limits the number of transactions that the attacker can generate.

Attacker's Goals. The end goal of the attacker is to flood mempools with unconfirmed *dust transactions*. At mempools, the arrival rate corresponds to the flow of incoming transactions and the departure rate corresponds to the rate of transaction mining. The departure rate is fixed, because the average block computation time and the size of the block are fixed. When the arrival rate increases due to a flood of transactions, it results in transactions backlog. Overwhelming the mempool size alarms the legitimate users, who naturally start paying higher mining fee to prioritize their transactions.

The secondary objective of the attacker will be to reduce the attack cost by getting his transactions rejected. For the attacker, mining will result into losing balance to miners. However, if the transactions get rejected, the attacker will have another chance to repeat the attack. Furthermore, while the attack on block size can be effectively countered by miners, the mempool attack cannot be prevented by them.

A. Attack Procedure

To reduce the attack cost, the attacker will design his transactions in a way that they are less likely to be prioritized by miners. At the same time, the attacker would want his transactions to stay in mempools for as long as possible. To this end, we envision that this attack can be carried out in two phases: the distribution phase and the attack phase.

The Distribution Phase. In the distribution phase, the attacker estimates the minimum relay fee of the network, divides his spendable bitcoins ("UTXO's") into various transactions and sends them to the sybil accounts. The attacker generates a series of outputs to all the addresses of sybil nodes with one or more transactions per address. Transactions made in the distribution phase will have input "UTXO's", which will be previously mined in the blockchain. Hence, these transactions will have greater-than-zero age (confirmation score), and will be capable of paying the relay fee and the mining fee.

The Attack Phase. In the attack phase, sybils will carry out "raw transactions" [7] from the balance received in the distribution phase. Sybils will generate dust transactions and exchange them with each other. The rate of exchange of transactions will be much higher than the network throughput. As a result, the arrival rate of the transactions at mempools will be higher than the departure rate of mined transactions. This will increase the transaction backlog and the mempool size.

Transactions made in the attack phase will have transactions of distribution phase as input "UTXO's". These inputs will still be awaiting confirmation in the blockchain. Due to that, their confirmation factor or age score will be zero.

B. Attack Cost

To reduce the attack cost, the attacker requires transactions to be part of the mempool but not the blockchain. This can be achieved by adding the minimum relay fee (R_f) to each transaction but not the minimum mining fee (M_f). The relay fee is necessary for a transaction to propagate in the network and be accepted by the mempool. If the attacker adds the mining fee, his transactions will attain priority from a miner and might get mined. To avoid that, sybils only pay the relay fee. If a transaction has i inputs, each contributing a size of k Bytes, and o outputs, each contributing a size of l Bytes, then the total size of the transaction S and its associated cost C(BTC) are determined by (1).

$$S(\text{Bytes}) = (i \times k) + (o \times l) + i, C = R_f \times \frac{S}{1024} \quad (1)$$

Assuming that the attacker is limited by a budget B (BTC) and minimum transferable value set by the network as $T_{\min}$, then the total number of transactions T_a that the attacker can generate can be computed in (2).

$$T_a = \frac{B \times 1024}{R_f \times T_{\min} \times [(i \times k) + (o \times l) + i]} \quad (2)$$

On the other hand, a legitimate user who intends to get his transaction mined, pays relay fee for transaction broadcast and mining fee as an incentive to the miner. For such a user, contributing a total T transactions, the cost incurred per transactions and the total cost of all transactions T_l is:

$$C = [R_f + M_f] \times \frac{S}{1024}, T_l = T \times [R_f + M_f] \times \frac{S}{1024} \quad (3)$$

Under these settings, the maximum loss an attacker can incur would happen if all his transactions possibly get mined. The cost in such a case will be the product of the total number of transactions and the relay fee ($T_a \times R_f$). The attacker can re-launch the same attack with a new balance of $B-(T_a \times R_f)$. If a portion of the attacker's total transactions t_a gets mined, where $t_a \leq T_a$, then the attacker would be able to re-launch the attack with new balance of $B - (t_a \times R_f)$.

IV. Modelling the Mempool Attack

As mentioned in §II, mempool acts as a buffer for unconfirmed transactions, where the incoming transactions denote the arrival, and transaction mining represents the departure process. As long as the arrival process is within the bounds of system's throughput, the mempool queue remains stable and there is no transaction backlog. However, the attacker overflows the buffer by accelerating the arrival process and increasing the queue size. Therefore, to construct effective countermeasures, it is useful to formulate this abstraction as a queuing theory problem with associated mathematical primitives. To that end, we model this attack as Lyapunov optimization problem that encapsulates the attack procedure and provides a roadmap towards countermeasures.

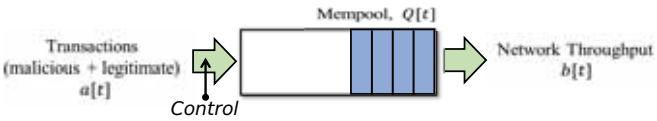


Fig. 2. Mempool as a queue, with arrival and departure processes. The arrow with control is the point where optimization schemes are applied.

A. Lyapunov Optimization

Lyapunov optimization is a popular scheme applied to the field of dynamic control systems for time-average optimization [10]. In queuing networks, Lyapunov drift is used to stabilize queues by optimizing time-average performance objectives. Subject to the queue stability, i.e., $\lim_{t\to\infty} \frac{1}{t} \sum_{\tau=1}^{t-1} Q[\tau] < \infty$, the time-average Lyapunov optimization (*i.e.,* drift-plus-penalty (DPP) [11], [12]) is defined as:

$$\textbf{Minimize}: \lim_{t\to\infty} \frac{1}{t} \sum_{\tau=1}^{t-1} y_o[\tau] \quad (4)$$

where $y_o[t]$ is the objective function at t and $Q[t]$ is the queue backlog at t, respectively. Based on DPP, this time-average optimization subject to queue stability can be formulated as the following decision-making framework:

$$\alpha^*[t] \leftarrow \arg\min_{\alpha[t]\in\mathcal{A}} [V \cdot y_o[\alpha[t]] + Q[t]\{a(\alpha[t]) - b(\alpha[t])\}] \quad (5)$$

where $\alpha[t]$ is possible decision at t, $\alpha^*[t]$ is time-average optimal decision at t, $\mathcal{A}$ is a set of possible decisions, V is tradeoff coefficient between optimization criteria and stability, $y_o[\alpha[t]]$ is objective value with decision $\alpha[t]$ at t, $a(\alpha[t])$ is an arrival with decision $\alpha[t]$ at t, $b(\alpha[t])$ is a departure with decision $\alpha[t]$ at t, respectively.

We model the mempool as a queue and show the arrival and departure processes. We present our design model in Figure 2, where the arrival process is denoted by $a[t]$, the departure process is denoted by $b[t]$, and the mempool is denoted by $Q[t]$. Since the mining rate eventually determines the network throughput, therefore, we use throughput as the departure process. The control allows us to apply countermeasures and modify the queue size. Our objective is to minimize the time-average size, subject to the queue stability. Queue stability can be achieved by removing malicious transactions or limiting the total number of transactions in the queue. The cost is the unwanted removal of legitimate transactions. Therefore, we have two cases that represent the state of the mempool. In the first case, when the mempool is idle, the applied control policy should only remove a small number of transactions to fulfill the objective function of time-average cost minimization. In the second case, when mempool is flooded, the applied control policy should remove a large number of transactions to guarantee mempool size stability. With these objectives, the Lyapunov-based DPP changes to (5) is updated to (6), where $C(\alpha[t])$ is power consumption when our decision is $\alpha[t]$. If the mempool is empty ($Q[t] = 0$), we have to minimize our cost by permitting more arrivals. If the mempool is flooded ($Q[t] \approx \infty$), we have to minimize $(\alpha[t])$ by removing a majority of malicious transactions, and stabilizing the mempool size.

$$\alpha^*[t] \leftarrow \arg\min_{\alpha[t]\in\mathcal{A}} \{V \cdot C[\alpha[t]] + Q[t]a(\alpha[t])\} \quad (6)$$

V. Countering The Mempool Attack

To counter DDoS on Bitcoin's mempool, we propose fee-based and age-based countermeasures.

A. Fee-based Mempool Design

For fee-based mempool design shown in the algorithm 1, we assume that the mempool is initially empty ($Q[t] = 0$) when transactions begin to arrive at $\alpha[t]$. We fix a baseline threshold beyond which the mempool starts spam filtering. Initially, when the transactions arrive at the mempool, for each transaction, the mempool checks if the transaction pays a minimum relay fee. If the transaction pays the minimum relay fee, it is accepted and the mempool size is updated. When the mempool size reaches the threshold, it starts applying the fee-based policy. Now, if the incoming transaction pays both the relay fee and the mining fee, only then it is accepted. The key idea is that only those transactions should be accepted, which aim to be eventually mined into the blockchain. As a result, we filter spam transactions to optimize the queue size. If the new size is less than the baseline threshold then the mempool proceeds its operation from relay fee check. Otherwise, it continues with the fee-based design.

Algorithm 1: Fee-based Mempool Design

```
   State: Mempool Empty
1  foreach transaction ∈ incoming transactions do
2      while (Mempool Size < Threshold Size) do
3          if (transaction relay fee > minimum relay fee)
             then
4              Mempool ← transaction
5              UPDATE (mempool);
6          else
7              (transaction relay fee < minimum relay fee)
                transaction rejected;
       State: Mempool Size Exceeds Threshold Size
8      while (Mempool Size > Threshold Size) do
9          while (transaction relay fee > minimum relay
            fee) do
10             if (transaction mining fee > minimum
                mining fee) then
11                 Mempool ← transaction;
12                 UPDATE (mempool);
13             else
14                 transaction rejected
15     return Mempool Size
   Result: Spam Transactions Rejected
```

1) Analysis of Fee-based Mempool Design: In the following, we analyze the workings of fee-based design and its utility in the light of our threat model. We will limit the number of transactions an attacker can generate within his budget by increasing the mining fee threshold. We also observe how this design affects legitimate users in the network.

In the current settings, if mempools employ the fee-based design, all spam transactions will be rejected. The mempool will only accept transactions that pay both the relay fee and the mining fee. Legitimate users will benefit from this design, since they will always pay the relay and the mining fee. Once the attacker becomes aware of this design, the only way he can attack is by adapting to the new settings. The attacker can do that by adding mining fee to each transaction. Given a budget B, adding mining fee to each transaction will reduce the total number of transactions T_a the attacker can generate, and the equation (2) will change to:

$$T_a = \frac{1024 \times B}{[(i \times k) + (o \times l) + i] \times [R_f + M_f] \times T_{\min}} \tag{7}$$

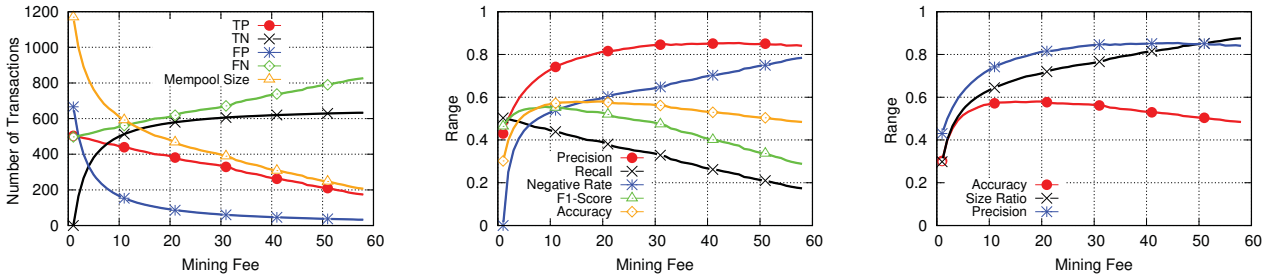
From (7), we can observe that the number of transactions the attacker can generate has an inverse relationship with the total fee paid per transactions. Using that relationship, we can adjust the fee parameter and investigate how it limits the attacker's capabilities. To do that, we simulate the affect of increasing the mining fee on the volume of transactions that the mempool accepts. We allocate a fixed budget to the attacker and select thresholds of minimum mining fee and maximum mining fee. Using (2), we select a suitable budget for attacker that results into 1,000 transactions with a minimum mining fee. Then, we generate 1000 legitimate transactions, each with a mining fee normally distributed over the range of the minimum and maximum mining fee. Using a discrete-event time simulation, we increase the mining fee and monitor its affect on transactions of the attacker and the legitimate users.

TABLE I
Confusion Matrix

		Actual Transaction	
		Legitimate	Malicious
Mempool Transaction	Legitimate	TP	FP
	Malicious	FN	TN

2) Evaluation Results: We plot the results in Figure 3, and use the confusion matrix in Table I, and evaluation parameters Table II to observe the effect of the fee-based design on the mempool. The results in Figure 3(a) show that with the increase in the mining fee threshold, the mempool size (TP+FP), malicious transactions (FP) and legitimate transactions (TP) decrease. The trend of (FP) is explained by (7). With a fixed budget, increasing the mining fee decreases the total transactions T_a. Accordingly, the size of the mempool also decreases due to fewer spam transactions (FP). However, increasing mining fee also limits the budget of legitimate users, which explains the trend of decreasing (TP). Figure 3(b), shows that the accuracy increases with the mining fee to a maximum value and then decreases. Using that, we found a minimum fee cutoff corresponding to the maximum accuracy. In Figure 3(c), we plot accuracy and size ratio; the size ratio is the fraction of mempool transactions out of the total number of incoming transactions. Lower size ratio indicates higher size optimization. The results show that at a fee threshold of 13, we achieve 60% accuracy, 70% size optimization, and 78% precision. Increasing the fee parameter further, increases the size optimization but decreases the accuracy. Therefore, the fee-based design presents a trade-off between the size efficiency and the accurate detection of malicious transactions.

3) Limitations of Fee-based Mempool Design: To understand limitations of "Fee-based Mempool Design" we highlight the nature of some transactions in Bitcoin. Suppose Alice sends 10 BTC to Bob in a transaction. That transaction is yet to be verified and mined, but Bob spends them by sending 5 BTC to Charlie. For Bob's transaction to be successfully mined, its parent transaction by Alice needs to be mined first. This



(a) Confusion Matrix (b) Evaluation Parameters (c) Accuracy, Size Ratio and Precision

Fig. 3. Analysis of Fee-based Design. As the mining fee increases, the mempool size reduces. However increasing fee also affects legitimate transactions which reduces detection accuracy. An optimum cut-off fee can be selected from Figure 3(c) based on the trade-off between accuracy and size ratio.

TABLE II
EVALUATION PARAMETERS

Precision	$\frac{TP}{TP+FP}$
Recall	$\frac{TP}{TP+FN}$
F_1	$\frac{2\times precsion\times recall}{precision+recall}$
Accuracy	$\frac{TP+TN}{TP+TN+FP+FN}$
Negative Rate	$\frac{TN}{TN+FN}$

sequence of transactions is known as parent-child transaction [7]. For a child transaction to be mined, its parent transaction needs to be mined first. Often when priority factor of a parent transaction is low, the child transaction increases the mining fee to increase the overall priority factor. This process is called "Child Pays For Parent" (CPFP) [7]. For legitimate users, this situation might be undesirable, since more child transactions can lead to transactions getting stuck. However, the attacker can exploit this to circumvent the fee-based design.

For transactions generated in the attack phase, their parent transactions in the distribution phase need to be verified and mined. The attacker can minimize the probability of transaction acceptance in the first phase by reducing their priority factor; e.g., by paying a minimum relay fee and no mining fee. Once the parent transactions have lower probability of acceptance in the first phase, the child transactions can increase their priority factor by adding higher relay fee and mining fee. In such a situation, and when the mempools apply the countermeasures, spam transactions of the attack phase will get into the mempool.

Countermeasure. One way to address this problem is to prioritize the incoming transactions on the basis of the mining fee. Mempool can sort the incoming transactions for the fee value and accept the ones which pay higher fee. As we increase the mining fee, the capability of attacker to produce transactions reduces (7). The attacker is constrained by the budget and increasing mining fee reduces the number of transactions he can produced. We can observe this trend in Figure 3(a). Although this reduces the number of spam transactions in the mempool and optimizes its size, it also reduces accuracy and the number of legitimate transactions that get accepted. As the fee parameter is increased, the capability of all the legitimate users to pay higher fee also decreases. To this end, the fee-based countermeasures limit the attacker from flooding the mempool, but also limit the number of legitimate transactions that successfully pass the fee threshold. To address these limitations we propose the age-based countermeasures.

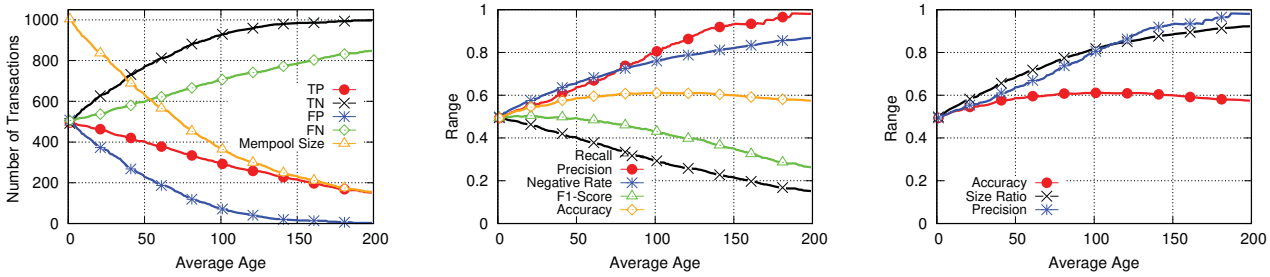
Algorithm 2: Age-based Mempool Design

State: Mempool Size Exceeds Threshold Size
foreach *transaction* $\in$ *incoming transactions* **do**
  initialize;
  $average\ age$ = 0;
  N $\leftarrow$ number of parent transactions of current transaction;
  while *(transaction relay fee* $>$ *minimum relay fee)* **do**
    while *(transaction mining fee* $>$ *minimum mining fee)* **do**
      $average\ age = \frac{(\sum_{i=1}^{N} parent_i)}{N}$;
      /* apply age filter */
      if *(average age* $>$ *minimum age limit)* **then**
        $Mempool \leftarrow transaction$;
        UPDATE (mempool);
      else
        $transaction\ rejected$
  return *Mempool Size*;
Result: Spam Transactions Rejected

B. Age-based Mempool Design

To limit attacker's chances, we propose the "Age-based Mempool Design" that addresses the limitations of our previous model. We leverage the confirmation factor or "age" of a transaction to distinguish between legitimate and malicious transactions. The age of a transaction determines how many confirmations it has achieved over time (§II).

For this design shown in algorithm 2, we assume that the baseline size threshold of the mempool has been reached, and the mempool is only accepting transactions which are paying the relay fee as well as the mining fee. Now, for each incoming transaction at $\alpha[t]$, we count the number of inputs or parent transactions. We initialize a variable "average age" and set its value to 0. Next, we calculate the average age of the transaction by adding the age of each parent transaction and dividing by the total number of parent transactions. This gives an estimate of confirmation score of the incoming transaction. Then, we apply a "minimum age limit" filter on the mempool. The "minimum age limit" can take any arbitrary value greater than 0. Only if the transaction's mean age value fulfills the criteria of age limit, then the mempool accepts the transaction.



(a) Confusion Matrix (b) Evaluation Parameters (c) Accuracy, Size Ratio and Precision

Fig. 4. Analysis of Age-based Design. Notice that with age-based design, the accuracy, precision, and size ratio are comparatively higher than the fee-based design. This policy is effective in rejecting the unconfirmed transactions.

A transaction in Bitcoin has an input pointer pointing to the spendable transaction that it has previously received. In this design, we apply the check on the age of the incoming transactions. In the attack phase (§III-A), the spam transactions will have input pointers of a parent transaction that will not be confirmed in any block. The age of of all those parent transactions, made in the distribution phase (§III-A), will be 0. Using this knowledge, we compute the average age of all the input pointers (parent transactions); minimum age value of 1 means that all transactions coming into the pool are confirmed in at least the most recent block. Once this design is implemented, if a user tries to spend his coins, he needs to have at least one valid confirmation backing up his transaction. This gives advantage to the legitimate user who can make a normal transaction with confirmed parent transaction of significant age. On the other hand, transactions of the attacker will be rejected due to low confirmation factor despite high fee.

1) Analysis of Age-based Mempool Design: Now, we analyze the working of "Age-based Mempool Design" and how it helps in countering DDoS attack. For this design, we establish that the attacker has the capability of circumventing the "Fee-based Design" and is willing to pay the relay fee and the mining fee for all transactions. Also, the attacker knows that its transactions will not be verified, so it pays higher relay and mining fee than the legitimate users.

To analyze the effectiveness of age-based countermeasures, we set a minimum age limit and a maximum age limit as thresholds for the incoming transactions. For the attacker, the only set of transactions with age value greater than 1 are generated in the distribution phase. Child transactions made in the attack phase were assigned 0 age value due to unconfirmed parent transactions. To capture that, we normally distribute the average age value of all malicious transactions from 0 to the minimum age limit. The average age value of all legitimate transactions was set from 0 to the maximum age limit. A total of 2000 transactions were generated with half of them being malicious and half being legitimate. Then we applied the age-based design on all the incoming transactions at the mempool. We increased the age requirement for the incoming transactions and evaluated the accuracy of detection and the state of mempool for each transaction.

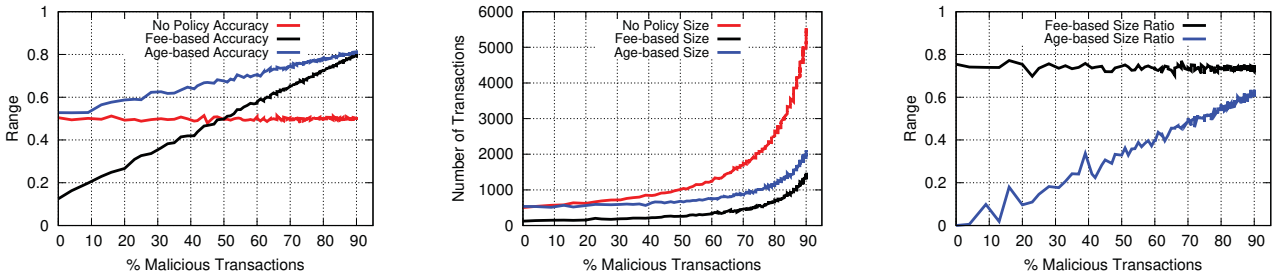
2) Evaluation Results: For evaluation of this design, we used the same confusion matrix in Table I and evaluation parameters in Table II. The results in Figure 4 show that upon increasing the average age the malicious transactions, (FP) decreases sharply. The mempool size decreases to a point where there are only legitimate transactions left in the mempool. Due to low (FP) and higher (TP), the precision reaches a close to 1 in Figure 4(b). In Figure 4(c), it can be observed that at an average age value of 100 we achieve 60% accuracy, 80% size optimization and 80% precision. As we increase the age parameter to 200, the accuracy does not decrease as of the fee-based design, while the size ratio increases up to 90% and the precision increases up to 98%. This shows that the age-based prevents a majority of malicious transactions from entering into the pool and favors the legitimate users in the system.

In these settings, if the attacker intends to spam the network, he needs to have majority of his transactions confirmed in the blockchain. However, in our attack model, we have described that confirmation is undesirable for the attacker since it results in losing budget in mining and relay fee. In Bitcoin, recall that the average block mining time is 10 minutes. For a single confirmation of all of the transactions, the attacker has to wait on average for 10 minutes. Using the results from Figure 4(c), the attacker will have to wait a minimum of 100 blocks to relaunch the attack. With average block computation time of 10 minutes, 100 blocks lead to 16 hours of delay. Even if the attacker still plans to carry out the attack after waiting and paying all the fee, he will not be able to flood the mempool. The best the attacker might achieve will be occasional network stressing with series of transactions. Higher attack cost and low incentive might discourage the attacker. Therefore, the age-based design offers more security against the DDoS attack while ensuring regular service for the legitimate users.

3) Limitations of Age-based Countermeasures: Although the age-based countermeasures are more effective in preventing DDoS attack, they may also have some limitation. In fast transactions where users cannot wait for verification [13], their transactions will be rejected by the mempool. An illustration of the fast transaction is bitcoin vending machine [14]. However, we do not see Bitcoin evolving into such an applications in near future, therefore it is not a significant problem.

VI. Experiment and Results

In this section, we describe the experiments we performed to compare our designs. We try to present the scenario similar



(a) Accuracy (b) Mempool Size (c) Size Ratio

Fig. 5. Performance of both designs under mempool DDoS attack. When the percentage of malicious transactions is low, indicating less severe attack, the fee-based policy is effective in terms of accuracy and size optimization. As the attack rate increases, the age-based policy also becomes more effective.

to [5], where a group of attackers attacked Bitcoin with unconfirmed transactions. We construct the attack scenario in our simulations and analyze our countermeasures for detection accuracy and mempool size optimization.

For simulations, we generate a series of legitimate and spam transactions. For both types of transactions, we normally distribute the relay fee and mining fee over a selected range of fee values. As per our attack procedure (§III-A), we generate the spam transactions with low age factor. Although, we observed in the distribution phase (§III-A), that the attacker may possess some transactions which can have some age value. To capture that, we normally distribute the age of spam transactions from 0 to a lower age limit, and for legitimate transactions, we normally distribute the age factor from 0 to an upper age limit. To mimic the attack scenario in [5], we fixed the size of legitimate transactions and sequentially increased the number of malicious transactions from $0 - 90\%$ of the total transactions. We found (TN) and (FP) from confusion matrix as suitable measures to determine efficiency of fee and age-based models in detecting spam.

In Figure 5(a), we plot the accuracy. It can be seen that the accuracy of the age-based design exceeds 80% as malicious transactions increase. The accuracy of fee-based design is low at the beginning, but eventually grows as the malicious percentage grows. The reason for the low accuracy at the start is due to the low detection of (TP). Although, the age-based design appears to be suitable for detecting malicious transactions, Figure 5(b) shows that the fee-based model achieves better size efficiency.

Mempool size is determined by (TP+FP). For both the designs, we plot mempool size in Figure 5(b). The figure shows that below 80% malicious transactions, fee-based design has lower mempool size. After that, age-based design becomes more size-efficient due to low (FP). To understand the mempool size optimization ratio, we use (8). The equation gives a ratio of size difference of mempool when countermeasures are applied. The results in Figure 5(c) show that fee-based design achieves a consistent size optimization of 78% irrespective of malicious percentage. The size ratio for age-based design increases with the percentage increase in malicious transactions. At 88% malicious transactions, age-based design achieves a size ratio of 60%. We have already shown in §V-A, that there is a trade-off between the accuracy and size optimization.

$$size\ ratio = 1 - \frac{mempool\ size\ under\ design}{mempool\ size\ no\ design} \tag{8}$$

We can use the size and accuracy trade-off to select appropriate countermeasures during attack. If the attack is less severe but the pending transaction backlog is high, the fee-based design will limit the incoming malicious transactions and optimize the mempool size until the backlog is cleared. If the attack is severe [5], and majority of incoming transactions are unconfirmed then age-based design will be more useful in detecting malicious transactions, reducing FP and optimizing mempool size. Using this knowledge and the results obtained from our experiments, we derive (9) that reduces the spam transactions under varying attack conditions.

$$\underset{f,a}{\text{minimize}} \quad \boldsymbol{R}_{spam}(f,a) = \alpha\frac{\boldsymbol{\Omega}(f)}{N} + (1-\alpha)\frac{\boldsymbol{\Phi}(a)}{N}, \tag{9}$$

In (9), f and a are the mining fee and average age cutoffs that we learn from Figure 4(c) and Figure 3(c). The cutoffs f and a are used to minimize the accepted spam ratio $\boldsymbol{R}_{spam}$. $\boldsymbol{\Omega}(f)$ and $\boldsymbol{\Phi}(a)$ are two functions of mining fee f and the average age a learned from the simulations to show the numbers of accepted spams under the two polices. $0 \leq \alpha \leq 1$ is a hyperparameter for balancing the weights of the two policies, and N is the total number of transactions. During the attack, when N increases, the mempools will check the transaction backlog and the nature of incoming transactions. If the backlog is high and incoming transactions are mostly legitimate, α is increased. If the incoming transactions are mostly unconfirmed and the arrival rate is high, α will be decreased.

VII. Related Work

In this section, we review notable work covering security aspects of blockchains [15], [16], [17].

Selfish mining is a form of attack where miners choose not to publish their block after computation, hoping to mine subsequent blocks and get more reward. The problem of selfish mining has been addressed by Eyal and Sirer [18], Sapirshtein *et al.* [19], Solat and Potop-Butucaru [20] and Heilman [21]. Eyal and Sirer [18] proposed defense strategies to deter selfish mining attacks on Blockchains. Block Withholding Attack (BWH), introduced in [22], is an attack in which miners in a pool choose to submit partial proof of work, instead of the full proof. As a result, they get rewarded for participating in the pool although the pool suffers a loss

due to partial solutions. Kwon *et al.* [23] studied a new attack on blockchains called "Fork After Withholding" (FAW) attack that guarantees higher rewards than block withholding attacks.

The 51% attack can be launched if a mining pool in the network gains more than 50% of the network's hashing power. With more than half the hashing power of network, the attacker can prevent transactions from verification and other miners from computing a block. Double-spending or equivocation happens when a user generates two transactions from the same inputs and sends them to two recipients [13].

Distributed denial-of-service (DDoS) attacks have been quite prevalent [24]. DDoS attacks are repeatedly launched against the mining pools, the legitimate users, and the currency exchanges. Johnson *et al.* [25] performed a game-theoretic analysis of DDoS attacks against Bitcoin mining pools. Vasek *et al.* [24] empirically illustrated the denial-of-service attacks on the Bitcoin system. Prior to its release on November 12, 2017, Bitcoin Gold suffered from a massive DDoS attack [5]. Cryptocurrency exchanges have also been frequently targeted to prevent coin tradings, and no clear nor specific mitigation techniques to those attacks have been proposed.

Another form of DDoS attack on blockchain includes spamming the network with low valued dust transactions. This attack is also called the *penny-flooding* attack. Baqer *et al.* [4] performed Bitcoin stress testing to analyze the limitations of the Bitcoin network, and how attackers exploit them.

VIII. Conclusion

In this paper, we identify a DDoS attack on Bitcoin mempools that traps the users into paying higher mining fee. Attacks on Bitcoin mempools have not been addressed previously, so we propose two countermeasures to the problem: fee-based and age-based designs. From our simulations, we conclude that when the attack is not severe, the fee-based design is more effective in mempool size optimization. However, it does so by affecting both the attacker and the legitimate users. In contrast, when the attack is severe, the age-based design is more useful in helping legitimate users while discarding maximum spam transactions. Although the size optimization achieved by the age-based design is less compared to the fee-based design, its accuracy of spam detection is much higher. In the future, we aim to extend our analysis towards a hybrid model that leverages the benefits of both designs and achieves maximum spam detection and size optimization.

Acknowledgement:. This work is supported by Air Force Material Command award FA8750-16-0301, Global Research Lab program of the National Research Foundation NRF-2016K1A1A2912757, and NSF grant CNS-1809000.

References

[1] M. Saad, L. Njilla, C. A. Kamhoua, and A. Mohaisen, "Countering selfish mining in blockchains," *CoRR*, vol. abs/1811.09943, 2018. [Online]. Available: http://arxiv.org/abs/1811.09943

[2] K. Croman, C. Decker, I. Eyal, A. E. Gencer, A. Juels, A. Kosba, A. Miller, P. Saxena, E. Shi, E. G. Sirer *et al.*, "On scaling decentralized blockchains," in *International Conference on Financial Cryptography and Data Security*. Springer, 2016, pp. 106–125.

[3] M. Saad, M. T. Thai, and A. Mohaisen, "POSTER: deterring ddos attacks on blockchain-based cryptocurrencies through mempool optimization," in *Asia Conference on Computer and Communications Security, AsiaCCS, Incheon, Republic of Korea*, June 2018, pp. 809–811. [Online]. Available: https://doi.org/10.1145/3196494.3201584

[4] K. Baqer, D. Y. Huang, D. McCoy, and N. Weaver, "Stressing out: Bitcoin "stress testing"," in *International Conference on Financial Cryptography and Data Security*. Springer, 2016, pp. 3–18.

[5] F. Memoria, "700 million stuck in 115,000 unconfirmed bitcoin transactions," 2017. [Online]. Available: https://goo.gl/mYX14V

[6] M. Saad and A. Mohaisen, "Towards characterizing blockchain-based cryptocurrencies for highly-accurate predictions," in *IEEE Conference on Computer Communications INFOCOM Workshops, Honolulu, USA*, Apr 2018, pp. 704–709. [Online]. Available: https://doi.org/10.1109/INFCOMW.2018.8406859

[7] B. Community, "Developer's Guide, Confirmation Score, Transaction Fee and Miner Fee, Minimum Relay Fee, UTXO, Memory Pool, Child Pays for Parent, Raw Transactions," 2018. [Online]. Available: https://bitcoin.org/en/developer-reference#rpc-quick-reference

[8] J. A. Kroll, I. C. Davey, and E. W. Felten, "The economics of bitcoin mining," in *Proceedings of WEIS*, 2013.

[9] B. Community, "Bitcoin Data from Blockchain.info," 2017. [Online]. Available: https://blockchain.info/charts/transaction-fees-usd

[10] C. Qiu, Y. Hu, Y. Chen, and B. Zeng, "Lyapunov optimization for energy harvesting wireless sensor communications," *IEEE Internet of Things Journal*, vol. 5, no. 3, pp. 1947–1956, 2018. [Online]. Available: https://doi.org/10.1109/JIOT.2018.2817590

[11] M. J. Neely, "Energy optimal control for time-varying wireless networks," *IEEE Transactions on Information Theory*, vol. 52, no. 7, pp. 2915–2934, Jul. 2006.

[12] J. Kim, G. Caire, and A. F. Molisch, "Quality-aware streaming and scheduling for device-to-device video delivery," *IEEE/ACM Transactions on Networking*, vol. 24, no. 4, pp. 2319–2331, Aug. 2016.

[13] G. Karame, E. Androulaki, and S. Capkun, "Double-spending fast payments in bitcoin," in *ACM Conference on Computer and Communications Security, CCS, Raleigh, USA*, Oct 2012, pp. 906–917. [Online]. Available: http://doi.acm.org/10.1145/2382196.2382292

[14] T. Bamert, C. Decker, L. Elsen, R. Wattenhofer, and S. Welten, "Have a snack, pay with bitcoins," in *Peer-to-Peer Computing (P2P), 2013 IEEE Thirteenth International Conference on*. IEEE, 2013, pp. 1–5.

[15] J. Chen, S. Yao, Q. Yuan, K. He, S. Ji, and R. Du, "Certchain: Public and efficient certificate audit based on blockchain for tls connections," in *IEEE Conference on Computer Communications, INFOCOM, Honolulu, USA*, Apr 2018. [Online]. Available: https://goo.gl/jzH71h

[16] C. Cai, X. Yuan, and C. Wang, "Hardening distributed and encrypted keyword search via blockchain," in *IEEE Symposium on Privacy-Aware Computing, PAC, Washington*, 2017. [Online]. Available: http://doi.ieeecomputersociety.org/10.1109/PAC.2017.36

[17] S. Hu, C. Cai, Q. Wang, C. Wang, X. Luo, and K. Ren, "Searching an encrypted cloud meets blockchain: A decentralized, reliable and fair realization," in *IEEE Conference on Computer Communications, INFOCOM, Honolulu, USA*, Apr 2018. [Online]. Available: http://nisplab.whu.edu.cn/paper/infocom_2018_1.pdf

[18] I. Eyal and E. G. Sirer, "Majority is not enough: Bitcoin mining is vulnerable," in *Financial Cryptography and Data Security*. Springer, 2014, pp. 436–454.

[19] A. Sapirshtein, Y. Sompolinsky, and A. Zohar, "Optimal selfish mining strategies in bitcoin," in *20th International Conference on Financial Cryptography and Data Security FC, Christ Church, Barbados*, Feb 2016, pp. 515–532. [Online]. Available: https://doi.org/10.1007/978-3-662-54970-4_30

[20] S. Solat and M. Potop-Butucaru, "Zeroblock: Preventing selfish mining in bitcoin." *arXiv preprint arXiv:1605.02435*, 2016.

[21] E. Heilman, "One weird trick to stop selfish miners: Fresh bitcoins, a solution for the honest miner," in *Financial Cryptography and Data Security*. Springer, 2014, pp. 161–162.

[22] M. Rosenfeld, "Analysis of bitcoin pooled mining reward systems," *arXiv preprint arXiv:1112.4980*, 2011.

[23] Y. Kwon, D. Kim, Y. Son, E. Y. Vasserman, and Y. Kim, "Be selfish and avoid dilemmas: Fork after withholding (FAW) attacks on bitcoin," in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, CCS, Dallas, USA*, Oct 2017, pp. 195–209. [Online]. Available: http://doi.acm.org/10.1145/3133956.3134019

[24] M. Vasek, M. Thornton, and T. Moore, "Empirical analysis of denial-of-service attacks in the bitcoin ecosystem," in *Financial Cryptography and Data Security*. Springer, 2014, pp. 57–71.

[25] B. Johnson, A. Laszka, J. Grossklags, M. Vasek, and T. Moore, "Game-theoretic analysis of DDoS attacks against bitcoin mining pools," in *Financial Cryptography and Data Security FC, Christ Church, Barbados*, Mar 2014, pp. 72–86. [Online]. Available: https://doi.org/10.1007/978-3-662-44774-1_6