Replacement Cycling Attacks on Bitcoin Miners Block Templates

Antoine Riard - btc@ariard.me - v0.1

Abstract. Bitcoin is a pure version of electronic cash allowing online payments to be sent directly from one party to another without going through an institutional intermediary. The system is secured by a peer-to-peer network where nodes are burning computational resources to find proof-of-work in exchange of a reward priced in the system units themselves. Such peer-to-peer nodes are usually designated as miners. As long as the a majority of CPU power is controlled by miners that are not cooperating to attack the network, they should generate the longest chain and outpace attackers. However, by using novel techniques to jam the computation phase of the updates to the chain i.e the generation of block template, an attacker can disproportionately capture a share of the system units reward and consumes this addition to progressively gain a higher share of CPU power among the unstructured network. In the present work, we present one such jamming technique, namely using the replace-by-fee mechanism to provoke jamming cycles in miners block template.

1 Introduction

Bitcoin is a peer-to-peer electronic cash system, which solves the double-spend problem with a trust-minimized architecture by letting everyone verify all transactions. As of January 2024, the system operates in the range of 80,000 nodes simultaneously running Bitcoin protocol software, not including the many more users of custodial services, under diverse forms of trust models.

Public auditability of the Bitcoin transaction history since the original genesis block in January 2009 is the foundation of removing trusted third parties. All full nodes in the system are verifying by default the ledger replica. This

ledger replica is updated by chaining blocks, i.e a collection of new transactions to be added to the history signed by a proof-of-work. A proof-of-work is the hash of the block data structure satisfying the difficulty threshold adjusted periodically by the network of bitcoin full-nodes.

A full-node engaging in the process of finding a valid block is known as a miner and expends CPU time and electricity in exchange of the block reward, composed of the block subsidy and the sum of the transaction fees.

As every simple full-node, a miner node is connecting with nodes in the network to discover issued transactions, cache them locally in the node's data structure known as a mempool and select them based on the offers in the miner local block template. This transaction collection process is continuously happening in parallel of the process of finding a valid proof-of-work for the current template. A mining node is periodically updating its current block template based on its local(s) mempool(s).

In this work, we explore how Bitcoin miners may be subject to a risk of having the reward for mining blocks significantly downsized, once their mempools are systematically jammed by competing miners.

At high-level, we exploit the transaction replacement and expiration mechanisms of Bitcoin full-nodes, that allows an unconfirmed transaction in the mempool to be replaced with a different transaction, spending a subset of the same inputs. The replacement mechanism allows further replacement of an already in-mempool transaction as long as the anti-DoS rules are satisfied.

Per replacement cycling attacks, a malicious actor replaces third-party transaction in the victim's mempool by a transaction spending one of the actor's UTXO and then replace this jamming transaction by re-spending the same UTXO with a conflicting transaction. This trick is cycled for each block for which there is a rebroadcast by the third-party transaction issuer reaching the victim's mempool.

At the next block finding by the malicious actor, the absolute fee reward collected by the block is increased by the jammed transaction, absent from any block of the jammed victim. Repeatedly jamming other mining nodes can constitute a disproportional advantage in the distribution of the total block reward among all mining nodes. This on the long-term can have consequences on the topological structure of the bitcoin peer-to-peer network, facilitating coins double-spend and other negative externalities.

Timing the sequence of replacement cycles with the series of fetches by the victim's node from its mempool to constitute a local block template is one of the difficulty factor of the attacks, as there is non-null chances than the honest transaction being stuck in the block template. As a local block template is composed of transactions data stored in a local data structure expose to the read / write of its network peers, adaptive replacement attacks could be devised to adapt to a specific miner's block template generation algorithm.

At the time of original publication, there are not yet available line of mitigation that a miner can deploy to mitigate this class of replacement cycling attacks targeting their block templates. Topological restrictions on chain of

bitcoin transactions coming with the new cluster mempool design might reduce the jamming surface exposed by a mempool, while numerous tradeoffs stay unknown.

The paper is structured as follows:

- We recall the background on the transaction synchronization process and its incentives
- We recall the Gambler's Ruin problem and its implications for the network security
- We define our threats models and the generic transaction-relay jamming of a miner block template processing
- We present 2 replacement cycling techniques to execute a miner block template transaction-relay jamming
- We suggest a list of open questions for further discussions and research

2 Background

In this section, we remind the fundamentals of the transaction synchronization process (i.e mining) among the peer-to-peer network of full-nodes and the economical incentives underlying this process.

Fundamentally, the Bitcoin system is a peer-to-peer distributed timestamp server chronologically ordering transactions by leveraging computational proofs instead of an authoritative stamp. The system is peer-to-peer as there is no privileged node elected by the consensus rules and distributed as one node failing at any moment should not affect operations of the wider system. The computational proofs temporal ordering from the Genesis block among the network of nodes is solved accordingly to the common consensus rules. This chronological sequence of computational proofs is usually called the blockchain.

The computational proofs in the Bitcoin system are known as proof-of-work, i.e for a given block data structure, there is a hash variant of this block satisfying the current network difficulty. This proof-of-work constitutes a form of signature of the block and probabilistically that enough computational resources have been burned to allow an update of the blockchain. Instead of being a signature of the knowledge of a secret, this is a signature of computational resources burned for a given public hash function. This scheme offer the benefit of allowing an open membership of the signatory set of the bitcoin blockchain, enabling any new node to join at any time this set and join the signatory set, without interactive cryptographic blessing by a trusted third party.

For the computational resources burned in the process of finding valid proof-of-work, there is an incentive mechanism devised by Bitcoin consensus rules to reward participating nodes. Those nodes are offered a coinbase static amount of coins, which is divided by 2 at each halvening (approximately every 4 years). This mechanism is the one by which new coins are introduced in the Bitcoin system, until expiration of the fixed supply. Mining nodes are also rewarded by transaction fees, which are used to signal inclusion priority in the blocks.

In this system, the coin transfer mechanism among users happens through transactions, spending already generated coins to new entries in the blockchain ledger, and doing so by offering a fee, the amount difference between inputs and outputs. Along the lifetime of the Bitcoin system, a replace-by-fee policy-only mechanism was introduced in most of Bitcoin full-nodes softwares in 2015 to allow the users to dynamically bid on the scarce blockspace by offering a fee increase on their previously propagated transactions.

3 The Gambler's Ruin problem and Bitcoin Network Security

In the original work formalizing the security of Bitcoin as a peer-to-peer electronic cash system, the game-theory classic Gambler's Ruin problem is introduced as an analogous reasoning device to compare a race between honest miners and a mining attacker in generating an alternate chain to commit double-spend attacks and other arbitrary changes. In the same work, it is assumed that (a) mining nodes are economically rational actors and (b) that temporary block subsidy and transactions fees may incentivize nodes to stay honest and to follow conformably the protocol rules to not undermine the validity of their own wealth.

This set of assumptions, if respected, should ensure that the majority of CPUs (today ASICs) and electricity determining the efficient update to the global network of ledger replicas should stay concentrated in the majority of the economically rational actors. This is demonstrated by an analysis of the Gambler's Ruin problem.

Formally, the Gambler's Ruin problem can be presented in the following fashion. A compulsive player engages in a coin toss with a casino. The coin is biased. If the player wins, he gains one credit. If the casino loss, he losses one credit. If the coin is measured to be biased at the advantage of the casino, whatever the initial wealth of the player, after some period of time this wealth will go to zero. The coin bias measurement and the initial wealth should determine the quantitative period of time to elapse before the gambler is ruined.

Similarly transposed to the analysis of the Bitcoin mining process among a crowd of economically rational miners and an attacker, the economically rational crowd and the attacker competitively engages in a process of finding a valid proof-of-work for the next block. The odds of finding the next valid proof-of-work are biased in favor of the player with the highest concentration of CPUs. Whichever player winning a toss trial i.e a proof-of-work finding, they get a measurable reward in coins. If the highest concentrations of CPUs is already concentrated in the hands of the crowd of economically rational miners, assuming the attacker's stack of CPUs is inferior, the odds of finding the next valid proof-of-work should be in favor of the crowd of economically rational miners. The initial concentration of CPUs in hands of the crowd of economically rational miners and the initial concentration of CPUs in the

hand of the attacker should determine the measurable period of time to elapse necessary for the attacker to catch-up on the longest valid chain.

By capturing a higher measured reward in coins for finding a valid proof-of-work, in proportion of the reward captured by the crowd of economically rational miners, an attacker can progressively buy more CPUs and electricity to increase its odds of finding the next valid proof-of-work and reduce the time necessary to catchup on the longest valid chain. Hypothetically on long periods, even obtaining a dominant strategy in selecting blocks for the longest valid chain.

In this work, we present such practical technique for a miner to capture a higher share of the block reward compared to its competing miners by jamming the transaction processing happening within the mempool and block template, concretized by exploiting replacement cycling attacks and other mempool tricks.

In the remainder of the paper, we present the replacement cycling attacks against Bitcoin miners block templates, with 2 practical variants of the attacks, respectively the feerate differential variant and the expiration time based one.

4 Transaction-Relay Jamming of Bitcoin Miner Block Template Order

In this section, we demonstrate the threat model we chose, we discuss the nature of replacement cycling at the miner-level and explain how to attack a batch transaction issued by an exchange. The attack explanation is given for a classic design mempool, while it has been tested to be effective on the cluster design mempool in its early implementation in the bitcoin core software.

4.1 Threat Model

First of all, we assume there are on-going bitcoin transaction traffic issued by a third-party, for example an exchange or another on-ramp service and that the attacker can be paid an output on one of those bitcoin transaction. We also make the following assumptions:

- Users run unmodified Bitcoin and a basic getblocktemplate stack
- Mining hashrate is stable and blocks are mined reliably
- Miners have available inbound connections slots where anyone can connect

For simplicity, we also assume that blocks are reliably relayed across nodes within seconds. We refer to the last known block as "chain tip". When it comes to the capabilities of an attacker, we consider:

- An attacker controls a non-null quantity of hashrate
- An attacker can deploy dozens of sybil nodes with modified Bitcoin software to mass-relay its transactions
- The network of dozens of sybil nodes allows the attacker to get a good visibility of the network

This threat model allows an attacker to execute the underlying attack (replacement cycling at the miner-level). An attacker then becomes capable of replacement cycling a victim miner and significantly downsizing the reward for its mined blocks.

4.2 Replacement Cycling Attacks on a Miner Block Template

A replacement cycling exploits the replacement mechanism as allowed by full-nodes mempools. Assuming default settings, the transaction replaced do not have to signal replaceability and the replacement candidates offer a high absolute fee than the sum paid by the original transaction (including descendants) and a feerate greater than all directly conflicting transactions, among other rules. This mechanism does not establish a bound on the number of replacement (rbf penalty).

A replacement cycling attacks works in the following way. Assume we have the transaction-relay topology Mallet, Mary and Alice where Mallet and Mary are both miners with equal odds of mining a block. Alice is a exchange with a high-volume of batch transactions propagated over the network and she supports fee-bumping to satisfy her service-level agreements towards her payee users. Mallet is the attacker targeting Mary. There is no collusion among Mallet and Mary. Chain tip is at 1000 blocks.

Alice broadcasts a batch transaction with 10 payouts outputs, among those outputs 2 belongs to Mallet and the other ones to unrelated payees of the exchange. Alice's batch transaction is signed with a 1 satoshi per virtual byte for a size of 500 virtual bytes. As soon as Alice's batch transaction enters the mempool, Mallet is blocking the ability to CPFP transaction by generating a chain of 24 junk 100-virtual bytes transactions on his 1st output and a chain of 1 junk 100-virtual bytes transaction on his 2nd output (to block the carve-out). Those junk transactions are signed with a 2 satoshis per virtual byte.

When Alice attempts to fee-bump her batch transaction, she uses a RBF batching with an absolute fee of 4000 satoshis for the same size, while the feerate is superior to the original batch transaction, the absolute fee is under and it is rejected by Mary's mempool.

At this point in time, Mallet conserves Alice's second RBF batching transaction and he proceeds to the following set of actions. He builds a single transaction of 200 virtual bytes with an absolute fee of 5200 satoshis spending the "carpet" UTXOs used to fund the junk branch on Alice's transaction and he inserts Alice's second RBF batching transaction and the "carpet" UTXO spends in his block template.

At this stage, Mary's block template is composed of Alice's original transaction and the 25 junk children for a total size of 3000 virtual bytes and an absolute fee reward of 5500 satoshis. On the other hand, Mallet's block template is composed of Alice's RBF batching and of his own "carpet" spending transaction for a total size of 700 virtual bytes and 9200 satoshis.

For equal odds of mining a block among Mallet and Mary, the jamming advantage of Mallet over Mary can be evaluated at an indice of 3.5, given with the following formula (with A for Adversary and T for Target):

$$\frac{(ATemplateSats/ATemplateSize)/(TTemplateSats/TTemplateSize)}{AHashrateShare}$$

At each round of Alice rebroadcast of a RBF batching transaction, during the propagation over the transaction-relay network, Mallet can replace a segment of his chain of junk transactions by a higher absolute fee child, or alternatively let the chain of junk transaction to be replaced with Alice's RBF batching transaction before to replace it with his own higher feerate child to be subsequently replaced out.

We observe that for this replacement cycling attack parameters and with the exposed formula, in a Bitcoin network topology of 5 miners with equal hashrate, a jamming attacker among those 5 miners can gain an indice with a 43 percentage advantage in executing a transaction-relay jamming attack.

The attack holds for different types of transaction traffic hijacked, as long as the mining attacker can gain a single output in the broadcast transaction. It should be noted that the attacker do not have to own any UTXO in the hijacked transaction traffic and those ones are spent parties not necessarily being miners themselves. Hijacked transaction traffic leveraged to mount a block template jamming can come from a wide variety of bitcoin transaction issuers: exchanges, lightning nodes, merchants.

5 Another Attack Vector: Transaction Expiration Time

In this section, we layout an additional block template order forgeability vector that can be used to improve transaction-relay jamming attacks on the bitcoin miners block templates, i.e the automatic expiration of old transaction.

Bitcoin full-node softwares implement some mechansims to expire received unconfirmed transaction after a lapse of time. When a transaction has been accepted to the mempool, there will be a screen of transactions that have been received more than the expiration delay. By default, this lapse of time is around 2 weeks.

This expiration mechanism allows a replacement cycling attacker to significantly improve on its jamming of a bitcoin miner block template by propagating a lot of low-fees transactions in the target mempool ordered by time to have one every block or e.g every 2 blocks. Any of this sleeping transaction can be used to re-spend on top a conflict with the the jamming transaction. At the next transaction reception, if the sleeping transaction is expired it will be kicked out of the mempool, even if the descendant transactions offers a high fee among mempool candidates.

This technique can be leveraged to gain a significant advantage in a replacement cycling attack, even for miners with very low-hashrate among the total Bitcoin network, and the satoshi cost for 2 weeks of sleeping transactions can be evaluated at roughly 201,600 satoshis.

6 Discussions

In this section, we suggest a list of open questions around transaction-relay jamming of the bitcoin miners block templates.

6.1 Mempool and Block Template Order Forgeability Vector

This work introduces the notion of computational forgeability of the transaction graphs ordering constituting a full-node mempool. Informally, a target mempool can be computationally forged if the average quantity of fees for a single unit is inferior to the attacker's mempool one for a given measurement unit.

In this work, we introduced 2 computational forgeability vectors relying on replacement and expiration affecting a target mempool average feerate unit quality, respectively the feerate differential among broadcast branches of transactions and the expiration time of a processed transaction. We leave as an open subject of research how to combine computational forgeability vectors in a single exploitation and the existence of other types of computational forgeability vectors.

6.2 Information Propagation and Processing Asynchronicity among Miners

While information propagation in the network of bitcoin full-nodes has been analyzed in previous works, their analysis was mainly centered on blocks propagation, more scarcely on transactions in themselves. One previous work analyzed the distribution of bitcoin transactions in a situation with Sybil miners where miners are preventing each other to collect fees. However, only the transaction relay phase is considered, while the difficulties arising from the block computation for the transaction authorization phase are not analyzed.

In this work, we center the analysis on the transaction computation or processing phase, the one being the object of a jamming by competing miners leveraging replacement cycling tricks. The competing miner is assumed to have a direct connection with the target mempool (e.g thought an inbound slot) and there are not considerations on the spontaneous latency among graphs of full-nodes. In an unstructured peer-to-peer network, the propagation latency of transactions can spontaneously affect the synchronous reception of transactions. By design, Bitcoin transaction-relay peer-to-peer network makes no assumptions on any synchronicity among full-nodes on the state of their mempool, until the next block reception.

This asynchronicity in the transaction processing arising from the lack of structure of the transaction-relay peer-to-peer network underlights why the transaction-relay jamming of bitcoin miners block templates presented in this work cannot be simply solved by removing the replace-by-fee mechanism as a fundamental mempool mechanism. By design, the transaction-relay protocol has always allowed a conflicting transaction in 2 distinct full-node mempools to be originated from a single unique UTXO, with no guarantee the 2 conflicting transaction offers an equivalent feerate. We leave as an open subject of research what in-depth mitigations can be devised in face of transaction-relay jamming attacks affecting Bitcoin miners block templates.

6.3 Competing Miner Cost Function

In this work, the evaluation of a competing miner average gain / cost is summarily, among all the miners are considered. Future analysis on the efficiency of transaction-relay jamming attacks on bitcoin miners block templates can integrate 3 parameters in its gain and cost estimation.

Firstly, a high hashrate adversary might have an advantage in diminishing the loss coming from the carpet transaction used to replace out a jamming transaction. The UTXO used to generate the carpet transaction can be covertly spent in the adversary's block template with a zero feerate, as a 0-fee transaction is a consensus valid bitcoin transaction.

Secondly, an adversary can silently piggyback on high-fee collaborative transactions to diminish the loss coming from the carpet transaction. The UTXO used to generate the carpet transaction can be silently contributed to a high-fee collaborative transactions, without the other collaborators of the transactions noticing the previous broadcast spend coming from this UTXO. An adversary can profit from the asymmetries in the distribution of the fees burden to the collaborative transaction.

Thirdly, a transaction-relay jamming adversary can realize a higher average gain by leveraging the eviction mechanism of full-nodes mempools. Full-nodes mempools have a maximum size limit on the number of in-memory transactions, when this limit is reached, the mempool starts to kick-out the transaction at the bottom ordered by feerate. This mempool mechanism can be triggered by the adequate timing of the replace out of the jamming transaction and the sudden influx of third-party transactions in the target mempool, evicting the jammed parent transaction, or another of the component of the package.

We highlight that those 3 parameters are non-exclusive of each other and we leave as an open subject of research coming up with fine-grained cost functions for each replacement cycling attack layout.

6.4 Transaction Traffic Hijack and Policy Exposure Surface

Fundamentally, transaction-relay jamming of a bitcoin miner block templates relies on the hijack of third-party issued bitcoin transactions, where on those

transactions the adversary has included a spending output. Analyzing the categories of transaction traffic propagating in bitcoin network mempools as of January 2024, numerous categories of transactions are exposed to be traffic hijacked to mount furher transaction-relay replacement cycling attacks, among them: exchange batch transactions, lightning commitment transaction, coinjoin, colored coins transactions. Even simple payment one input / one output transaction to a merchant can be exploited, if this merchant is the attacker itself, which is a more restrained scenario.

Among those categories of transactions, usage of transaction-relay policy mechanism that increases the malleability surface of a chain of transactions provides more quantitative traffic to be exploited by a transaction-relay jamming adversary. The malleability surface increase can come from removing the necessity of a signature to authenticate that a fee-bumping child is issued by the same signer than the parent. We leave as an open subject of research the mempool policy mechanisms increasing the exposure surface of their beneficent transactions towards traffic hijack.

7 Conclusion

In this work we present a new class of transaction-relay jamming attacks affecting Bitcoin miners block templates. As a novel fact, this attack allows to gain a dominant strategy in exfiltrating fee reward for mining blocks among a set of miners with equivalent hashrate distribution. This attack works by exploiting the replace-by-fee mechanism present in full-nodes mempools. A variant of this attack leverages the expiration mechanism of full-nodes mempools to gain higher efficiency. This is an open question how privileged capabilities of the attacker might enhance attack success and efficiency.

8 Acknowledgements

We would like to thanks the Bitcoin Protocol Development Community for many insights and fruitful observations.