Replacement Cycling Attacks on the Lightning Network

Antoine Riard - btc@ariard.me

Abstract. The Lightning Network brings a solution to the Bitcoin's known scalability problem by introducing a network of off-chain payments channels between a restricted set of counterparties. The security of such second layer relies on the ability of participants to confirm on-chain transactions to claim at any time. This ability is critical when off-chain HTLC balances are at stake during a time-bounded period, enforced by Bitcoin Script timelocks. Assuming reasonable interactivity with the counterparty, unilateral closure should happen very rarely.

One of the vulnerability risk identified early on it is a transaction relay jamming of the unilateral broadcast of mentioned time-sensitive transactions by a counterparty with a competing interest.

Such jamming is possible due to malicious interference with the propagation rules of the base-layer network of Bitcoin nodes. In theory, a jamming can be maintained until expiration of the time-bounded safety period guaranteeing than Lightning packets (e.g HTLCs) cannot be claimed by the counterparty with an opposite interest.

One practical class of transaction-relay jammings is known as "pinning attacks, a technique to hinder the propagation of transaction between transaction-relay network nodes by abusing default anti-DoS protections. In this paper, we present a "second" class of transaction-relay jamming attacks, dubbed "replacement cycling". This offensive exploitation relies on the existence of replacement as transaction fee-bumping mechanism itself. Replacement cycling can be considered as more concerning than pinning attacks as it can be achieved whatever the state of network mempools congestion. While the attack success is dependent on the timing of miner block template construction, those algorithms are mostly public and their processing can be predicted by an attacker. We argue this replacement cycling is one of the most practical way to steal Lightning funds.

According to our investigation, it is currently possible to steal the total channel capacity of Lightning routing hops if the capacity can be fully routed as pending HTLCs. A replacement cycling can be also played out to prevent Lightning payment nodes to recover their funds with a lesser liquidity griefing severity only.

In addition, a replacement cycling attack neither requires access to hashrate, nor network-level interference capabilities. An attacker can have only regular lightning channels with the target and a basic Bitcoin full-node.

We present a serie of mitigations either at the Lightning Network level or at the transaction-relay and mempool level of the Bitcoin base-layer. A subset of those mitigations have been implemented and deployed by major Lightning implementations. Unfortunately, we think those mitigations are not robust in face of sophisticated adversaries with low-hashrate capabilities.

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 October 2023, the system operates at least 50,000 nodes simultaneously running Bitcoin protocol software, not including the many more users of custodial services, under diverse forms of trust model.

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 transaction history and keeping the validation ressources low is a network security parameter. The drawback of public auditability is a constraint on the transaction throughput. Second-layers protocol on top of Bitcoin were designed to overcome this limitation. The Lightning Network is an example of a contracting protocol, where the security model is relying on pre committed transaction, shared-ownership of UTXOs and timelocks. It is solving the double-spend problem by shifting it to be a private matter (as opposed to being solved on-chain).

Such second-layers protocols introduce new assumptions when compared to original Bitcoin threat model. In this work, we explore how LN users may be subject to a risk of having their funds stolen, once their honest transaction broadcast are systematically jammed.

At high-level, we exploit the transaction replacement mechanism of Bitcoin full-nodes, that allows an unconfirmed transaction in a mempool to be replaced with a different transaction, spending the same inputs. This policy rule allows further replacement of package ancestor as long as the anti-DoS rules are satisfied.

Per replacement cycling attack, a malicious actors replaces a second-stage claim transaction issued by the victim by its own pre-committed transaction

and then replace an ancestor of this malicious second-stage transaction to stale the spend of a time-sensitive outbound HTLC output. This trick is cycled at each block if there is a rebroadcast from the victim until the safety timelock of the inbound HTLC output expires and HTLC carriage over a routing hop can be double-spent by nehighboring channel counterparties.

Good propagation of the malicious replacement is one of the difficulty factor of the attacks, as there is a low chance than the honest transaction being stuck in the block template. In addition of monitoring the transaction broadcast of the honest Lightning node, an attacker can use modified transaction-relay software to propagate its own malicious transaction at advantage.

Low-hanging fruit mitigations can be deployed by Lightning implementations to harden the practicatility of replacement cycling attacks. As a first solution a Lightning node can monitor its local mempool to detect the broadcast of a counterparty claim with the expected satisfying witness. A second solution can be to aggressively rebroadcast the second-stage HTLC claim with a defensive fee, therefore deterring the attacker to launch the attack.

However, this line of mitigations might not be robust in face of adversaries with low-hashrate capabilities able to mine their own transactions or no special capabilities adversaries able to place a double-spent of a malicious input in a batched transactions.

The paper is structured as follows:

- We define our threats models, the setup to launch replacement cycling attacks and their execution
- We present the mitigations which have been deployed by Ligthning implementations and additional counter-measures
- We present how a replacement cycling attack can affect package relay
- We suggest a list of open questions for further discussions and research

2 Replacement Cycling Attack

In this section, we demonstrate the threat model we chose, we discuss the nature of replacement cycling, and explain how to attack the per-hop packet delay.

2.1 Threat Model

First of all, we assume that an attacker can open two payment channels with the victim node and the victim node accept HTLC forwarding. We also make the following assumptions:

- Users run unmodified Bitcoin and Lightning node software
- The blockchain provides transaction safety based on confirmations; mining hashrate is stable and blocks are mined reliably
- the network of honest users forms a connected graph, except for a victim eclipsed by an attacker
- channels opened can be inidfferently legacy, anchor outputs or taproot types

For simplicity, we also assume that blocks are reliably relayed across nodes within seconds. We refer to the latest known block as "chain tip".

When it comes to the capabilites of an attacker, we consider:

- An attacker does not control any hashrate
- An attacker can deploy dozens of sybil nodes with modified Bitcoin software to mass-relay its transactions

This threat model allows an attacker to execute the underlying attack (replacement cycling). An attacker then becomes capable of replacement cycling a victim and stealing funds from their payment channel.

2.2 Replacement Cycling Attacks

A replacement cycling exploits the replacement mechanism as allowed by full-nodes mempools. Assuming default settings, the transaction replaced must signals 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. However, this mechanism does not establish a bound on the number of replacement (rbf penalty), neither than the replacement transaction follows the semantic of the overlaying contract protocol.

A replacement cycling attack works in the following way. Assume we have the topology Mallory, Bob, Mallet sharing channels, the link Mallory Bob and Bob Mallet. Mallory and Mallet are the attackers in collusion against Bob a routing hop. Chain tip is at 1000 blocks.

Mallory forwards a 1 BTC HTLC to Mallet by the intermediary of Bob, with a timeout at block 1020 on the Bob Mallet link and a timeout at block 1080 on the Mallory Bob link. All channels are legacy types.

Bob's Lightning implementation is configured to break a channel at timeout + 3 blocks in case of pending offered HTLC to timeout with a second-stage transaction. In the present case, Bob should break the channel on block 1023 (the delta of 3 is a Lightning implement policy grace delay to enable cooperative settlement to encompass asymmetries in block propagation / processing between Lightning counterparties chain validation clients).

At block 1020, Mallet broadcasts a low-feerate chain A of transactions (in anticipation to bypass bip125 rule 2 preventing the addition of new unconfirmed inputs to a replacement transaction). At block 1023, Bob broadcasts his commitment transaction, it confirms at block 1024. Bob broadcasts his 1 sat / vbyte HTLC-timeout transaction at block 1024.

Mallet observers the HTLC-timeout propagation on the peer-to-peer network, and broadcasts a replacement candidate made of a 2 sat / vbyte the HTLC-preimage (double-spending the HTLC-timeout) and a spend of the chain A ancestor (double-spending the child).

Once this chain of transactions is well-propagated on the transaction-relay network, Mallet replaces the parent of the chain A with a higher-fee / high-feerate transaction. There should not be anymore a pending spend of the offered HTLC output in a network mempools.

Bob's Lightning implementation (before replacement cycling mitigations deployement) should rebroadcast the HTLC-timeout at its rebroadcast frequency (e.g pre-replacement cycling LDK behavior was on height based timer not necessarily at every block). At each rebroadcast of the HTLC-timeout from Bob-side, Mallet should rebroadcast a chain of 2 transactions and then the HTLC-preimage (ackknowledging the fee for bip 125 rule 4 rbf penalty).

The sequence is replayed for 57 blocks until block 1080, at which Alice broadcasts her commitment transaction and a HTLC-timeout. Those two transactions confirm at block 1080 (assuming legacy channel – no 1 CSV).

Once the transactions are confirmed on the Mallory Bob link, Mallet can finally let her HTLC-preimage confirms in block 1081. Mallory and Mallet successfully double-spend Bob of 1 BTC.

The attack holds for anchor output channel type, where the confirmed commitment transaction can be either Bob's or Mallet's one. It should be noted the attack could be played on a payment receiver, where the HTLC-timeout for a payment failure resolution is blocked with a replacement cycling. The severity should be only a timevalue DoS, however it could be a building block for other types of attacks. Different configurations of package topology can be layed out by the attacker optimizing to save on replacement fees.

3 Mitigations

In this section, we introduce five ways replacement cycling attacks can be mitigated or hardened. We also present their limitations or known means of bypass by advanced attackers

3.1 Local-Mempool and Transaction-Relay Traffic Monitoring

The nature of a replacement cycling attack consists in delaying the inclusion of a second-stage HTLC-timeout in miners block templates, until the inbound HTLC can be claimed by another second-stage HTLC timeout of the attack-controlled "left-side" node. However, if the victim Lightning node gains knowledge of the inbound HTLC preimage, an HTLC-preimage transaction can be broadcast immediately, without interactivity with the attacker-controlled "left-side" node.

Indeed, this preimage must be included in the attacker's HTLC preimage of the "right-side" puppet node to be accepted as a valid transaction by network mempools. If the HTLC-preimage propagates inside the local mempool of the victim node, opportunity is given to extract the preimage from the HTLC transaction and construct a satisfying witness to claim the offered output on the inbound channel state. This preimage extraction can be realized independently from the HTLC-preimage transaction confirmation on the "right-side" channel state. As of October 2023, local-mempool watching has been implemented by the LND and Eclair implementations.

This mitigation can be bypass by a mempool-partitioning trick where the local mempool is partitioned from the rest of the network in two steps. Firstly by identifying the Bitcoin full-node associated with the victim Lightning node, then secondly by partially partitioning the victim local mempool.

Identifying a Bitcoin full-node associated with its Lightning node has been explored in the past under the name of a cross-layer mappping technique. Known and successful techniques are e.g mapping Bitcoin and Lightning node operating under the same IP. Another known technique is to open low-value channels with public Lightning nodes, route an HTLC across it and provoke a force-close while parallelly observing the initial broadcast of a HTLC-preimage by a Bitcoin full-node. As of October 2023, one shot initial broadcast of Lightning channel transaction over the transaction relay network is not known to be widely implemented and deployed by Lightning nodes.

Once the Bitcoin full-node and victim Lightning node association has been realized, a partial partitioning can be achieved by brodcast two conflicting transactions A and B between the local mempool of the victim and the rest of the network. Conflicting transactions with the exact same feerate and absolute fee will not suceeed to replace each other as they're not paying for the replacement penalty. The local mempool of the victim sees transaction A and the rest of the majority of network mempools sees transaction B.

Replacement cycling can be tried again where the malicious HTLC-preimage spends transaction B in the rest of network mempools though it won't be accepted in the local mempool due to a missing spent parent. The local mempool watching mitigation is neutralized as the HTLC-preimage is never discovered by the victim Lightning node.

One step further to alleviate the concern of partial mempool partitioning can be for a Bitcoin full-node to implement the monitoring of received HTLC-preimage before the sequene of acceptance checks of the mempool. We leave for future research if transaction-relay traffic monitoring can be made robust against denial-of-service and advanced mempool partitioning threats.

3.2 Miner-mempool Monitoring

Building on the insights of local-mempool monitoring, an advanced form of mitigations can be the introduction of an overlay transaction-relay network between miners nodes and Lightning nodes.

To be successful, a replacement cycling attack must replace the honest HTLC-timeout out of all the miner's mempool with a sufficient odd of mining a block during the targeted timelock of attacked HTLCs. Every time the honest HTLC-timeout is replaced, the malicious HTLC-preimage shows up temporarily in miners' mempools. This appearance provides an opportunity for preimage detection and extraction to be then further relayed on an overlay network with a pub-sub pattern. This mitigation design has been already proposed in the past to defeat pinning attacks at the commitment-transaction level.

Conceptually, it presents few limitations. Firstly, if there is no privacy-preserving techniques backed up in this overlay and the listening Lightning nodes are directly connecting to the miner's mempool, this constitutes a cross-layer mapping technique, where the miner's mempools can be linked with the announcement of a HTLC-preimage transaction. From then, further mempool partitioning can be launched by attackers to blind miners from receiving either HTLC-preimage or HTLC-timeout by "pinning-in-the-middle" transaction-relay nodes between the miners mempools and the honest Lightning node.

Secondly, overlay network can only have limited processing capabilities (bandwidth / CPU / memory) and denial-of-service protections of those ressources in a "fair" fashion is unclear. Connections slot can be occupied by attacker-controlled puppet nodes and amplification attacks triggered where the honest preimage are not announced on the overlay network, the bandwidth being consumed by "malicious" HTLC-preimage transactions.

Thirdly, an overlay network maintained by miners nodes might introduce an incentive hazard. While originally, the Bitcoin network was envisioned as a peer-to-peer network where all nodes will mine blocks, or at the very least a good majority of them, years of deployment has thrust the mining of new blocks in the hands of specialized nodes and entities. E.g in a future where the block reward offers higher variance, miners nodes operating the overlay might restrain the access on short notice to extract more incomes from the ecosystem of Lightning nodes, jeopardizing their channel funds security.

3.3 Aggressive Rebroadcasting

The efficiency of a replacement cycling attack is dependent on the HTLC-preimage successfully replacing the honest HTLC-timeout and being conflicted out at each block. One way to harden the attack is to aggressively rebroadcast the HTLC-timeout at interval inferior to one block and at random point. This behavior coerces the attacker to monitor network mempools to observe the propagation of the HTLC-timeout and react in consequence to the honest rebroadcast to counteract with a malicious HTLC-preimage and subsequent replacement.

As an implementation detail, it should be noted the HTLC-timeout must be re-signed at every rebroadcast attempt to alter its identifier and avoid interference with p2p anti-DoS filters of the transaction-relay network.

3.4 Defensive fee-rebroadcasting

The efficiency of the defensive-fee rebroadcasting relies on the economic deterrence of the attacker, where more fees will be burnt by the attacker than gained as stolen HTLC value. Indeed, with anchor channels HTLC-timeout transactions, sighashsingle anyonecanpay signatures are exchanged between counterparties. Those signatures can be used to adjust the absolute fee offered by the HTLC-timeout. This absolute fee can be adjusted to a fraction of the

HTLC value where the sum of each fraction should be equal to the HTLC value. Assuming the HTLC-timeout is broadcast at least once per block, the full HTLC value should be paid by the adversary as a deterrence.

The efficiency of this mitigation is uncertain if face of network mempools spikes, where the HTLC's ancestor score is too low to be included in a block template and get stuck in few blocks in network mempools. Those blocks of delay can be opportunistically leveraged by an attacker, which doesn't have to replace the HTLC-timeout until in its in the top mempool space of the block template.

3.5 Per-Hop Packet Delay Bumping

With every block to jam, there is an odd of the HTLC-timeout of the target Lightning node being included in a miner's block template, and as such the attack to fail. Indeed, block variance is uncertain and it be can hardly predicted by an attacker. Increasing the per-hop packet delay bumping is a practical mitigation for Lightning nodes. As of October 2023, all major Lightning implementations have 'cltv expiry delta' settings between 34 to 144 blocks.

4 Replacement Cycling Attacks in a Package Relay World

4.1 Solving Pinning Attacks with Package Relay and nVersion=3

An extension of the current transaction-relay protocol has been proposed since years and as of October 2023 is under implementation in the Bitcoin Core client. Package relay introduces new peer-to-peer messages enabling nodes to request and relay the unconfirmed ancestor package of a known transaction. It comes with the additional advantage of allowing a package feerates to be evaluated as a whole, therefore ensuring the good propagations odds of high-fees CPFP attached to low-fees parent transactions.

Coupled with a new set of replacement policy rules (nversion=3), package relay should present a solution to the known pinning attacks affecting the Lightning Network, as much as the other Bitcoin second-layers sharing the same set of characteristics. Under current consensus and policy rules, low feerate and low-fee parent propagation can be jammed by attaching a malicious pinning child with a high-fee yet low-feerate. Under enough network mempools congestion, the pinned transaction can stuck in the mempool until the expiration of a Lightning channel timelock.

Indeed the new set of replacement policy rules constrains the total size of the package and as such the malicious child even if presents a high fee will be forced to yield a high feerate incentivizing miners node to include the whole malicious pinning package in their block template for fast confirmation. Additionally, the new peer-to-peer messages allow a honest high-feerate CPFP

to be attached to a low-feerate parent to compete with any concurrent package spending the same UTXO.

4.2 Breaking Package Relay with Replacement Cycling Attacks

Replacement cycling attacks introduce a novel fashion to break the security hardening introduced by package relay at the benefit of Bitcoin second-layers such as Lightning channels.

This attack is based on evicting of the network mempools a time-sensitive Lightning commitment transaction until pending HTLCs are mature and their resolution can be double-spent. The attack starts by two lightning channels being opened Mallory Bob and Bob Mallet. Bob enforces a cltv delta of M blocks on incoming HTLCs. When an outgoing HTLC expires at T, Bob should broadcast the parent commitment transaction and a HTLC-timeout transaction to avoid an asymmetric resolution between the outgoing and incoming HTLC. This broadcast of a chain of transaction has M blocks to confirm in the chain before Bob is at risk of an asymmetric resolution. Assuming package relay deployment over the base-layer transaction-relay network, channels transactions are presigned with the mempool minimum relay feerate.

Mallory forwards an HTLC over her link with Bob to be routed on Mallet link. Once the HTLC is committed on the links, Mallet withholds the resolution. When block T is reached, Bob broadcasts the channel commitment transaction and a high-fee child in a single package to be relayed over the network. As soon as the package propagation is under way and its feerate is known, Mallet can broadcast a higher-fee package with its own commitment transaction and high-fee child maliciously replacing the Bob's package. The high-fee child of this parent can be double-spent by a subsequent 3rd broadcast, letting in the network mempools Mallory's low-feerate commitment transaction.

Assuming there is sufficient network mempools congestion where the top mempool with weight-depth equals 1 block is superior to the low-feerate commitment transaction, the low-feerate commitment transaction should be stuck in the network mempools. At each new block, Bob should rebroadcast a package of its honest commitment transaction and high-fee child, unsuccessfully being replaced by Mallet's package at each attempt.

At block height T + M, the incoming HTLC on the Mallory-Bob link expires and Mallory can broadcast and confirm a commitment and a HTLC-timeout transaction. On the other link, Mallet broadcasts and confirms a HTLC preimage transaction, Bob has been double-spent.

While this attack assumes network mempools congestion, in the lack of sufficient congestion, the attacker can continuously buy the top mempool space to prevent the inclusion of its own commitment transaction pre-signed with a minimum relay feerate. Assuming a minimum relay feerate of 1 sat/vbyte and an effective block size of 2 MB, the bounded cost is equal to 0.02 BTC. This attacker cost can be compensated by the scaling effect of targeting multiple Lightning channels at the same time.

5 Discussions

5.1 Low-Hashrate Capabilities Attackers Advantage

As presented in the Section mitigation, one of the most efficient way to mitigate a replacement cycling attack against pending HTLCs is to ensure at each rebroadcast to attach an absolute fee equals to a fraction of the HTLC value, where the fraction should be equal to the HTLC value divided by the number of blocks timelocked. The total replacement cost of at least V+1 paid by the adversary should deter the launch of a replacement cycling attack.

However if the attacker has the capability to mine blocks during this period and the transactions in the mempool has a lower absolute fee than the fractional fee offered by the replaced HTLC, the deterrence effect is lost. With an increase in the hashrate capabilities of the attacker there is an increase in the gain retained in a successful replacement cycling attack. With an increase in the max HTLC timelock accepted by a Lightning routing hop, there is an increase in the set of low-hashrate attacker that can benefit from this advantage. As of October 2023, the max HTLC timelock is of 2016 blocks from the chain tip.

5.2 High-Frequency and High-Fee Collaboration Transactions Advantage

Assuming the defensive-fee mitigation is deployed, a fractioncal absolute fee is attached to each rebroadcast of the second-stage HTLC-timeout transaction. This absolute fee must be overpaid by the attacker rebroadcasting its own malicious HTLC-preimage, majored by the RBF penalty if a naive rebroadcast is done. If the fees is paid at each rebroadcast, even if the targeted HTLC is successfully double-spent, the adversary is at zero-sum.

However, an attack has the ability to inject one of the input of the ancestor or the HTLC-preimage in a high-fee collaborative transaction (e.g a multi-party dual-funding) where the fractional absolute fee burden is divided between all the participants of the high-fee collaborative transaction. Those participants might be honest and unaware than one of the input is used to jam a Lightning HTLC resolution. Once this high-fee collaborative transaction is confirmed, it will double-spent the ancestor of the HTLC-preimage, therefore the fractional fee cost is not fully on the burden of the attacker. Replacement cycling attacks might have an incentive to run large-scale collaborative transactions makers to dimish the cost of their attacks.

5.3 Miners Harvesting Attacks

If dynamic fee-bumping is unsecure, Lightning channels nodes might rely on pre-signed high absolute fees scaled on worst historical level of network mempools congestion. If there is a necessity to go to chain a commitment or HTLC-transaction can be broacast, without anchor output or malleability, therefore preventing replacement cycling or pinning attacks.

However, this introduce a risk of miners harvesting Lightning channels, where coalition of miners might have an interest to open channels with Lightning nodes during periods of low-fees, when the block template feerate can be inferior to the pre-signed transactions. At soon as the channels have been opened, miners can provoke mass force-close to collect the high absolute fee pre-signed transactions. Game-theory equilibrium of Lightning fee-bumping allocation and miners incentives is unclear.

6 Conclusion

In this work we present a new class of transaciton-relay jamming attacks affecting the current and future versions of the protocol. As a novel fact, this practical attacks allows to steal funds from Lightning channels even in the lack of network mempools congestion, an easing of the environment conditions defining state-of-the-art Lightning attack. All the funds up to what is allowed as in-flight HTLC value can be targeted. A variant of the attack can affect the future p2p extension package relay. This is an open question how privileged capabilities of the attacker might enhance attack success and efficiency.

7 Acknowledgements

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