Strategic Analysis of Griefing Attack in Lightning Network

Subhra Mazumdar, Prabal Banerjee, Abhinandan Sinha,
Sushmita Ruj, *Senior Member, IEEE*, and Bimal Kumar Roy

***Abstract*—Hashed Timelock Contract (*HTLC*) in Lightning Network is susceptible to a *griefing attack*. An attacker can block several channels and stall payments by mounting this attack. A state-of-the-art countermeasure, Hashed Timelock Contract with Griefing-Penalty (*HTLC-GP*) is found to work under the classical assumption of participants being either honest or malicious but fails for rational participants. To address the gap, we introduce a game-theoretic model for analyzing griefing attacks in *HTLC*. We use this model to analyze griefing attacks in *HTLC-GP* and conjecture that it is impossible to design an efficient protocol that will penalize a malicious participant with the current Bitcoin scripting system. We study the impact of the penalty on the cost of mounting the attack and observe that *HTLC-GP* is *weakly effective* in disincentivizing the attacker in certain conditions. To further increase the cost of attack, we introduce the concept of *guaranteed minimum compensation*, denoted as ζ, and modify *HTLC-GP* into HTLC-GP$^\zeta$. By experimenting on several instances of Lightning Network, we observe that the total coins locked in the network drops to 28% for HTLC-GP$^\zeta$, unlike in *HTLC-GP* where total coins locked does not drop below 40%. These results justify that HTLC-GP$^\zeta$ is better than *HTLC-GP* to counter griefing attacks.**

***Index Terms*—Bitcoin; Lightning Network; Griefing Attack; Game Theory; Hashed Timelock Contract with Griefing-Penalty or *HTLC-GP*; Guaranteed Minimum Compensation.**

I. Introduction

Blockchain has redefined trust in the banking system. Transactions can be executed without relying on any central authority [1]. However, blockchain-based financial transactions cannot match traditional payment systems [2] in terms of throughput. Factors serving as the bottleneck are computation-overhead involved in verifying transactions, and expensive consensus mechanism [3]. Layer 2 solutions [4] have been developed on top of blockchain to address these shortcomings. One of the solutions, *payment channels* [5] are widely deployed and quite simple to implement. Several interconnected payment channels form a payment channel network or PCN. The network is used for executing several transactions between parties not directly connected by a channel, without recording any of them in the blockchain.

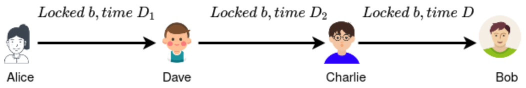


Fig. 1: Formation of contract, forwarding conditional payment from *Alice* to *Bob*

Lightning Network is the most popular Bitcoin-based PCN [6]. A payer can securely transfer funds to an intended recipient in the network by using a Hashed Timelock Contract or *HTLC*. It is a form of conditional payment where a payment succeeds contingent on the knowledge of the preimage of a hash value. For example, in Fig. 1, *Alice* intends to transfer b coins to *Bob* via channels *Alice-Dave, Dave-Charlie*, and *Charlie-Bob*. *Bob* generates a hash $H = \mathcal{H}(x)$ and shares it with *Alice*. The latter forwards a conditional payment to *Dave*, locking b coins for time D_1. *Dave* forwards the payment to *Charlie*, locking b coins for D_2 units of time. Finally, *Charlie* locks b coins with *Bob* for time D, where $D_1 > D_2 > D$. To claim b coins from *Charlie*, *Bob* must release x within the time D. If *Bob* does not respond, then *Charlie* withdraws the coins locked from the contract by going on-chain and closing the channel after the timeout period. However, *Bob* manages to lock b coins in each of the preceding payment channels for the next D units [7]. This is an instance of griefing attack [8]. An empirical analysis [9], [10] shows that a griefing attack reduces the liquidity of Lightning Network, and it has been used for eliminating specific edges in PCN [11].

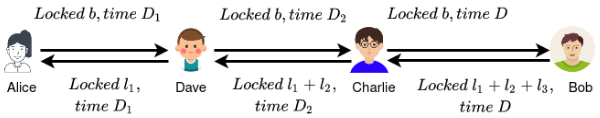


Fig. 2: Off-chain contract formation in *HTLC-GP*

A payment protocol Hashed Timelock Contract with Griefing Penalty or *HTLC-GP* [12] was proposed to counter griefing attack. The off-chain contract formation in *HTLC-GP* is illustrated in Fig. 2. *Dave* has to lock l_1 coins as a guarantee against b coins locked by *Alice*, for a period of D_1 units. If *Dave* responds within D_1, he claims b coins and unlocks l_1 coins. If he fails to respond, *Alice* goes on-chain, closes the channel, unlocks b coins, and gets the compensation from

Subhra Mazumdar was with Cryptology and Security Research Unit, Indian Statistical Institute Kolkata, India. She is now with TU Wien and Christian Doppler Lab Blockchain Technologies for the Internet of Things, Vienna, Austria. Email: subhra.mazumdar@tuwien.ac.at

Prabal Banerjee is with Cryptology and Security Research Unit, Indian Statistical Institute Kolkata, India and Polygon (previously Matic Network). Email: mail.prabal@gmail.com

Abhinandan Sinha was with Economic Research Unit, Indian Statistical Institute Kolkata, India. He is now with Ahmedabad University, India. Email: abhinandan.sinha@ahduni.edu.in

Sushmita Ruj was with CSIRO's Data61, Sydney, Australia. She is now with University of New South Wales, Sydney, Australia. Email: sushmita.ruj@unsw.edu.au

Bimal Kumar Roy is with Applied Statistics Unit, Indian Statistical Insitute Kolkata, India. Email: bimal@isical.ac.in

Dave. Similarly, *Charlie* has to lock $l_1 + l_2$ coins for a period of D_2 units as a guarantee against b coins locked each by *Dave* and *Alice*. *Bob* locks $l_1+l_2+l_3$ coins for a period of D units as a guarantee against b coins locked by *Charlie*, *Dave* and *Alice*. The drawback of *HTLC-GP* is that it does not consider an attacker to be rational. If *Bob* intends to mount griefing attack, he will choose a strategy that avoids paying any penalty. He will resolve the payment off-chain with *Charlie* just before the contract's locktime elapses. In this way, he manages to lock *Alice, Dave* and *Charlie*'s coins without compensating them. We realize that the underlying punishment mechanism in *HTLC-GP* must be argued from a game-theoretic point of view and not from the cryptographic aspect.

A. Contributions

We make the following contributions in our paper:

(i) This is the first attempt to propose a two-player game-theoretic model for analyzing griefing attacks in *HTLC*. The first player receives a conditional payment and makes a decision of forwarding the same to the second player based on the belief that the latter may be *corrupt* or *uncorrupt.*

(ii) We use the same game model to analyze the griefing attacks in *HTLC-GP* [12]. From the analysis, we conjecture that it is impossible to design an effective countermeasure without changing the Bitcoin scripting system.

(iii) We analyze the impact of the penalty on the attacker's behavior and infer that *HTLC-GP* is *weakly effective* in countering the attack in certain conditions.

(iv) We introduce the concept of *guaranteed minimum compensation*, ζ, and propose a protocol, HTLC-GP$^\zeta$, for disincentivizing griefing attack.

(v) Our experimental analysis shows that the total coins locked by the attacker drops to 28% when the guaranteed minimum compensation is 2.5% of the transaction amount and the maximum allowed path length is set to 10 in HTLC-GP$^\zeta$. This quantity is 27% less than the coins locked in *HTLC-GP*, proving that HTLC-GP$^\zeta$ is far more effective than *HTLC-GP* in countering the griefing attack. The code is provided in GitHub[1].

B. Organization

Section II discusses the background and Section III discusses the state-of-the-art. In Section IV, we propose a game-theoretical model of griefing attack in *HTLC*. We discuss an existing countermeasure for griefing attack, *HTLC-GP*, in Section V and use the game model discussed in Section IV to analyze the griefing attacks in *HTLC-GP* in Section VI. In Section VII, we propose the concept of *guaranteed minimum compensation*, denoted as ζ. We modify *HTLC-GP* into HTLC-GP$^\zeta$ by incorporating the concept of minimum compensation, and discuss in Section VIII. Experimental analysis of the effectiveness of *HTLC-GP* and HTLC-GP$^\zeta$ is provided in Section IX. Section X discusses how scalable is HTLC-GP$^\zeta$ compared to the state-of-the-art. Finally, we conclude the paper in Section XI.

II. Background

(i) Lightning Network: It is a *layer 2* solution for scalability issues in Bitcoin blockchain [6]. It is modeled as a bidirected graph $G := (V, E)$, where V is the set of nodes and $E \subseteq V \times V$ is the set of payment channels opened between a pair of nodes. Every node charges a processing fee for processing payment requests, defined by a function f, where $f : \mathbb{R}^+ \rightarrow \mathbb{R}^+$. Each payment channel $(U_i, U_j) \in E$ has an initial capacity $locked(U_i, U_j)$, denoting the amount deposited by U_i in the channel. $t_{i,j}$ is the timestamp at which the channel was opened. In the context of the Bitcoin blockchain, this will be the block height. T is the expiration time of the channel $id_{i,j}$, i.e., once a channel is opened, it remains active till time T units[2]. $remain(U_i, U_j)$ signifies the residual amount of coins U_i can transfer to U_j via off-chain transactions. M denotes the average fee for mining a Bitcoin transaction.

(ii) Hashed Timelock Contract: It is used for forwarding conditional payments across parties not directly connected by the payment channel [6]. Suppose a payer U_0 wants to transfer funds to a payee U_n via an n-hop route $P = \langle U_0, U_1, U_2, \ldots, U_n \rangle$ in the network. U_n creates a condition Y defined by $Y = \mathcal{H}(x)$ where x is a random string and $\mathcal{H}$ is a cryptographic hash function [15]. U_n shares the condition Y with U_0. The latter uses Y for conditional payment across the whole payment path. Between any pair of adjacent nodes (U_{i-1}, U_i) in P, where $i \in [1, n]$, the hashed timelock contract is defined by $HTLC(U_{i-1}, U_i, Y, b, t_{i-1})$, where $t_{i-1} = t_i + \Delta$, where Δ is the worst-case confirmation time for a transaction to get confirmed on-chain. The contract implies that U_{i-1} locks b coins in the off-chain contract. The coins locked can be claimed by the party U_i only if it releases the correct preimage $x' : Y = \mathcal{H}(x')$ within time t_{i-1}. If $\mathcal{H}$ is a collision-resistant hash function, then $x' \neq x$ with negligible probability. If U_i doesn't release the preimage within t_{i-1}, then U_{i-1} settles the dispute on-chain by broadcasting the transaction. The channel between U_{i-1} and U_i is closed and U_{i-1} unlocks the coins from the contract. If both the parties mutually decide to settle off-chain then U_i either releases the preimage and claims b coins from U_{i-1}. If U_i decides to reject the payment then b coins are refunded to U_{i-1}.

(iii) Dynamic Games of Incomplete Information or Sequential Bayesian Games: In this class of games, players move in sequence, with at least one player being uncertain about another player's payoff. We define a belief system and a player's behavioral strategy to approach these games. A *type space* for a player is the set of all possible types of that player. A *belief system* in a dynamic game describes the uncertainty of that player of the types of the other players. A *behavioral strategy* of a player i is a function that assigns to each of $i's$ information set a probability distribution over the set of actions to the player i at that information set, with the property that each probability distribution is independent of every other distribution. A dynamic game of incomplete information [16] is defined as a tuple that consists of (i)

[1] https://github.com/subhramazumdar/Strategic_Analysis_Griefing

[2] Each channel in Lightning Network has an infinite lifetime. However, we assume an upper bound on the channel lifetime for our analysis. Setting channel expiration time has been used in the literature as well [14]

a set of players $\mathcal{I}$; (ii) a sequence of histories H^m at the m^{th} stage of the game, each history assigned to one of the players (or to Nature/Chance); (iii) an information partition. The partition determines which of histories assigned to a player are in the same information set; (iv) a set of pure strategies for each player i, denoted as S_i; (v) a set of types for each player $i : \theta_i \in \Theta_i$; (vi) a payoff function for each player $i : u_i(s_1, s_2, \ldots, s_l, \theta_1, \theta_2, \ldots, \theta_l)$; (vii) a joint probability distribution $p(\theta_1, \theta_2, \ldots, \theta_l)$ over types. *Perfect Bayesian equilibrium* [17] is used to analyze dynamic games with incomplete information.

III. Related works

Few existing works analyze the payments executed in Lightning Network from a game-theoretic point of view but none of them have analyzed the griefing attack. Zappala et al. [18] proposed a framework for formally characterizing the robustness of blockchain systems in the presence of Byzantine participants. The authors have defined *HTLC* as a game between three participants. However, it is assumed that an *HTLC* can either be accepted or rejected but there is no discussion on griefing. Rain et al. [19] discusses the shortcoming of the game model of multi-hop payment proposed in [18]. They have improved the model and analyzed *wormhole attacks* in the network but the work does not discuss the griefing attack.

In [20], a game-theoretic analysis of atomic cross-chain swaps using *HTLC* has been provided. They have studied the impact of token price volatility on the strategic behavior of the participants initiating the swap and suggested the use of collateral deposits to prevent parties from canceling the swap. Han et al. [21] proposed use of premium for fairness in the atomic cross-chain swap. This is the first paper to introduce penalty as a countermeasure for countering the griefing attack in the context of atomic swap. However, the paper lacks a detailed analysis of how the introduction of premiums might ensure a faster exchange of assets. Also, the premium calculated is not a function of the assets locked in the off-chain contract and its timeout period. This might lead to under-compensation of victims. Our work is the first to propose a game model for analyzing griefing attack in Lightning Network. Several countermeasures like upfront payments [22], proof-of-Closure of channels [23], etc were proposed for mitigating the work. However, they are not effective because the malicious node does not lose anything in the process. Hashed Timelock Contract with Griefing Penalty or *HTLC-GP* [12] claims to address the above shortcomings and disincentivize griefing attacks. We use our proposed game model to study the effectiveness of *HTLC-GP* and suggest appropriate modifications to the protocol.

IV. Analysis of griefing attack in HTLC

A. System Model

Given an instance of payment in G, where a payer S wants to transfer α coins to a payee R where $S, R \in V$. The notations used in defining the model is summarized in Table I. We discuss how the payment is routed in the network:

(i) S finds a path P of length $\kappa : \kappa \in \mathbb{N}$ that connects it to R. The maximum allowed path length for routing is n so $|P| = \kappa \leq n$. We denote $P = \langle U_0 \rightarrow U_1 \rightarrow U_2 \ldots \rightarrow U_\kappa \rangle$, where $U_0 = S$ and $U_\kappa = R$ and $locked(U_{i-1}, U_i) > \alpha, \forall (U_{i-1}, U_i) \in E, i \in [1, \kappa]$.

Notation	Description
$G := (V, E)$	A bidirected graph representing the Lightning Network
V	Set of nodes in Lightning Network
E	Set of payment channels in Lightning Network, $E \subset V \times V$
α	Amount to be transferred from sender S to receiver R
P	Path connecting S to R
κ	Length of the path P in G connecting payer S to payee R.
n	Maximum allowed path length for payment, where $n \in \mathbb{N}, \kappa \leq n$
$U_i \in V, i \in [0, \kappa]$	Nodes in $P, (U_{i-1}, U_i) \in E$, where $U_0 = S$ and $U_\kappa = R$.
$id_{i,j}$	Identifier of channel (U_i, U_j)
T	Lifetime of a channel
$t_{i,j}$	The time at which the channel between U_i and U_j was opened
$locked(U_i, U_j)$	Amount of funds locked by U_i in the payment channel (U_i, U_j) while channel opening.
$remain(U_i, U_j)$	Net balance of U_i that can be transferred to U_j via off-chain transaction
$f(\alpha)$	Processing fee charged by a node for forwarding α coins to its neighbor
λ	Security Parameter
$\mathcal{H}\{0,1\}^* \rightarrow \{0,1\}^\lambda$	Standard Cryptographic Hash function
Δ	Worst-case confirmation time when a transaction is settled on-chain
t_i	HTLC Timeout period in channel $(U_i, U_{i+1}), i \in [0, \kappa - 1]$
D	Least HTLC Timeout period (or $t_{\kappa-1}$)
L	Bribe offered per attack
$I_{t,\alpha}$	Compensation offered for keeping α coins unutilized for the next t units of time
r_U	Rate of payments processed by node U per unit time
$O(r_U, t, val)$	Opportunity cost of a node U for next t units of time, also denoted as $o_U^{t,val}$
$t_{contract\_initiate}$	Timestamp at which off-chain contract got initiated
M	Average fee charged for mining a transaction
Γ_{HTLC}	Extensive form of a 2-party sequential Bayesian game in *HTLC*
γ	Rate of griefing penalty (per minute)
$\Gamma_{HTLC-GP}$	Extensive form of a 2-party sequential Bayesian game in *HTLC-GP*
ζ	Guaranteed Minimum Compensation
k	Ratio of the cumulative penalty locked by payer and the payment value locked by payee
$\gamma^{\zeta,k}$	Rate of griefing-penalty in HTLC-GP$^\zeta$ for a given ζ and k
$\tilde{n}^{\zeta,k}$	Maximum path length in HTLC-GP$^\zeta$ for a given ζ and k, $\tilde{n}^{\zeta,k} \leq n$

TABLE I: Notations used in the paper

(ii) Criteria for a node to route the payment: A node U_{i-1} can lock α_{i-1} coins in an off-chain *HTLC* formed with U_i if $remain(U_{i-1}, U_i) \geq \alpha_{i-1} : \alpha_{i-1} = \alpha_0 - \Sigma_{k=1}^{i-1} f(\alpha_{i-1}), i \in [1, n]$. Node U_{i-1} gets a processing fee $f(\alpha_{i-1})$. If U_i claims the coins, the residual capacity is updated as follows : $remain(U_{i-1}, U_i) = remain(U_{i-1}, U_i) - \alpha_{i-1}$ and $remain(U_i, U_{i-1}) = remain(U_i, U_{i-1}) + \alpha_{i-1}$.

(iii) Once the path is decided, U_κ generates a payment condition $H = \mathcal{H}(x)$ and shares it with U_0. The latter forwards the payment across P by forwarding *HTLC*'s. The *HTLC* timeout period between any pair U_{i-1} and U_i is set to $t_{i-1}, i \in [1, \kappa]$. If U_i chooses to resolve the *HTLC* just before timeout period, it responds at time $t_{i-1} - \delta$, where $\delta \rightarrow 0$. The least *HTLC* timeout period D is assigned for the contract between $U_{\kappa-1}$ and U_κ, i.e., $t_{\kappa-1} = D$.

(iv) Lightning Network uses the Sphinx protocol [24] while forwarding *HTLC*s. It is a form of onion routing where none of the intermediate parties have any information regarding the routing path except their immediate neighbors. Thus, a node U_i upon receving a request, knows that a conditional payment request came from U_{i-1} and it must be forwarded to node U_{i+1} where $i \in [1, \kappa - 1]$.

System Assumption: We discuss some of the assumptions:

(i) All the nodes in the network are rational [25], [26][3]. Rational processes always seek to maximize their expected utility. They may deviate or not choose to participate in a protocol depending on the situation.

(ii) We assume that a channel between U_{i-1} and U_i is unilat-

[3] For our model, we rule out any altruistic and Byzantine behavior and focus on the most typical scenario where participants are rational. However, the Lightning network may have Byzantine as well as altruistic nodes. We leave the analysis of griefing attack in the BAR model [27] as future work.

erally funded by $U_{i-1}, i \in [1, \kappa]$, i.e., $locked(U_i, U_{i-1}) = 0$. (iii) We define a function $O : \mathbb{R}^+ \times \mathbb{W} \times \mathbb{R}^+ \rightarrow \mathbb{R}^+ \cup \{0\}$, where $O(r_U, t, val)$ is the expected revenue a node U would have earned had it utilized the amount val for processing transactions in a period of t units given that r_U is the rate of payments processed by U per unit time. In other words, O defines the *opportunity cost* [28]. We discuss the procedure to calculate the opportunity cost of locked coins in Section I of the Supplemental File.

B. Attacker Model

An attacker with budget $\mathcal{B}_{EX}$ tries to disrupt the network by incentivizing a certain fraction of nodes to mount the griefing attack [4]. We make the following assumptions - (i) if a node has accepted the bribe, then it implies that the expected earning by cooperating with other nodes is lower than the bribe, and hence it has chosen to be a corrupt node. Such nodes act as per the instructions received from the attacker, and (ii) a corrupt node knows whether another node is corrupt or uncorrupt. An uncorrupt nodes lacks this information.

We discuss the bribe offered to a corrupt node and the method for mounting a griefing attack:
(a) *Bribe offered per attack.* Given a payment send across the network is of value α, the attacker fixes the bribe offered to a node to L coins, where $L = \alpha + I_{D,\alpha} + C$. Here C is the auxiliary cost for routing payment and opening new channels, if needed. $I_{D,\alpha}$ coins are used to compensate the node for keeping α coins unutilized for the next D units of time. We assume $I_{D,\alpha} \approx 2O(r_U, D, \alpha), \forall U \in V$ so that a corrupt node gains at least $O(r_U, D, \alpha)$ inspite of locking α coins.
(b) *Method for mounting Griefing Attack.* The attacker instructs the corrupt node to execute a self-payment (i.e., $S = R$) of α coins via a route of maximum allowed path length n in order to maximize the damage. The least HTLC timeout period is $t_{n-1} \approx D$. After the conditional payment reaches the payee U_n, it intentionally stops responding, locking a collateral of $(n-1)\alpha$ for the next D units in the path routing payment.

C. Game Model

(i) *Choice of players*: We assume that all the miners in the underlying blockchain are honest, and only nodes in Lightning Network can be the strategic players. In the path P, a node U_{i-1} locks an amount α_{i-1} with the off-chain contract formed with U_i, hence it will be bothered with $U_i's$ nature and the corresponding action. Except for U_0, none of the intermediaries routing the payment knows the recipient's identity. U_{i-1} makes a decision of whether to forward a payment to U_i based on the belief of the $U_i's$ type (discussed next). We model the griefing attack as an interaction between pair of nodes U_{i-1} and $U_i, i \in [1, \kappa]$ in path P routing payment in the network. Player forms a belief about the type of the other players based on either their position in the network or past interaction.

(ii) *Action Space*: We define the actions of U_{i-1} and U_i:

- *U_{i-1}'s action ($S_{U_{i-1}}$)*: It can either *forward (F)* the conditional payment to U_i or it can choose to *not forward (NF)*. If it chooses to forward the payment, it forms a contract with U_i, locking the designated amount in the channel $id_{i-1,i}$ for time t_{i-1}, which is the *HTLC* timeout period. If $i > 1$, then U_{i-1} gets a fee of $f(\alpha_{i-1})$ from U_{i-2} contingent to the release of solution by U_i. For U_0, the satisfaction level is proportional to success of payment. In this case, we consider $f(\alpha_0)$ as the level of satisfaction. If U_i delays, then opportunity cost of coins locked in the off-chain contract increases, and a loss is incurred. If U_i doesn't respond within t_{i-1}, then U_{i-1} closes the channel and withdraws its coins from the contract.
- *U_i's action (S_{U_i})*: If U_{i-1} has forwarded the payment, then U_i can choose its action from the following: *accept the payment* or *Ac*, *reject the payment* or *Rt*, *wait and then accept* or *W & Ac*, *wait and then reject* or *W & Rt*, and *grief* or *Gr*. If U_{i-1} does not forward the payment, the game aborts.

We define the sequential *Bayesian* game Γ_{HTLC} as the tuple $\Gamma_{HTLC} = \langle N, (\Theta_{U_{i-1}}, \Theta_{U_i}), (S_{U_{i-1}}, S_{U_i}), p_{U_{i-1}}, (u_{U_{i-1}}, u_{U_i}) \rangle$ [29], where $N = \{U_{i-1}, U_i\}$ where $i \in [1, \kappa]$. The type of player U_i is defined as $\Theta_{U_i} = \{\text{Corrupt(co), Uncorrupt (uco)}\}$. The probability function $p_{U_{i-1}}$ is a function from $\Theta_{U_{i-1}}$ into $p(\Theta_{U_i})$, where the $p(\Theta_{U_i})$ denotes the set of probability distribution over Θ_{U_i}, i.e., $p_{U_{i-1}}(Corrupt) = \theta_i$, $p_{U_{i-1}}(Uncorrupt) = 1 - \theta_i$. The payoff function $u_k : \Theta \times S \rightarrow \mathbb{R}$ for any player $k \in \{U_{i-1}, U_i\}$, where $\Theta = \Theta_{U_i}$ and $S = S_{U_{i-1}} \times S_{U_i}$, is such that for any profile of actions and any profile of types $(\hat{\theta}, s) \in \Theta \times S$, specifies the payoff the player k would get, if the player's actual type were all as in $\hat{\theta}$ and the players all chose their action as in s.

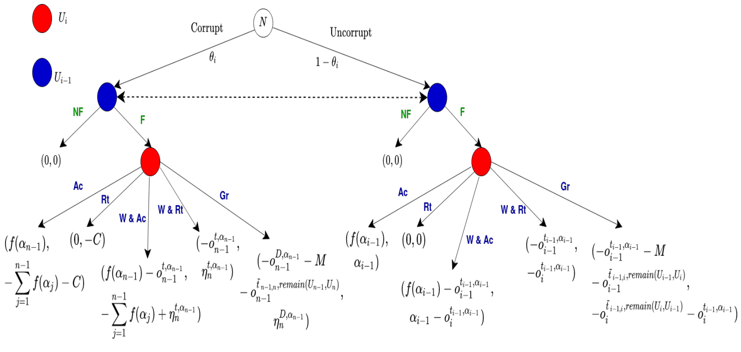

U_i
U_{i-1}
N
Corrupt
Uncorrupt
θ_i
$1-\theta_i$
NF
F
NF
F
$(0,0)$
$(0,0)$
Ac
Rt
W & Ac
W & Rt
Gr
Ac
Rt
W & Ac
W & Rt
Gr
$(f(\alpha_{n-1}), -\sum_{j=1}^{n-1} f(\alpha_j) - C)$
$(0, -C)$
$(f(\alpha_{n-1}) - o_{n-1}^{t,\alpha_{n-1}}, -\sum_{j=1}^{n-1} f(\alpha_j) + \eta_n^{t,\alpha_{n-1}})$
$(-o_{n-1}^{t,\alpha_{n-1}}, \eta_n^{t,\alpha_{n-1}})$
$(-o_{n-1}^{D,\alpha_{n-1}} - M - o_{n-1}^{\tilde{t}_{n-1,n}, remain(U_{n-1},U_n)}, \eta_n^{D,\alpha_{n-1}})$
$(f(\alpha_{i-1}), \alpha_{i-1})$
$(0,0)$
$(f(\alpha_{i-1}) - o_{i-1}^{t_{i-1},\alpha_{i-1}}, \alpha_{i-1} - o_i^{t_{i-1},\alpha_{i-1}})$
$(-o_{i-1}^{t_{i-1},\alpha_{i-1}}, -o_i^{t_{i-1},\alpha_{i-1}})$
$(-o_{i-1}^{t_{i-1},\alpha_{i-1}} - M - o_{i-1}^{\tilde{t}_{i-1,i}, remain(U_{i-1},U_i)}, -o_i^{\tilde{t}_{i-1,i}, remain(U_i,U_{i-1})} - o_i^{t_{i-1},\alpha_{i-1}})$

Fig. 3: Extensive form of game Γ_{HTLC}

1) Preference Structure: The game begins with Nature (**N**) choosing the type of U_i, either *corrupt* or *uncorrupt*, respectively. U_{i-1} believes that a corrupt U_i will be selected with probability θ_i, whereas an uncorrupt U_i will be selected with probability $1 - \theta_i$. After **N** makes its move, U_{i-1} selects its strategy based on the belief of U_i's type. U_i chooses its strategy after U_{i-1} has forwarded the payment. The extensive form is represented in Fig.3. If U_{i-1} chooses not to forward, then either party receives a payoff 0 since no off-chain contract got established, i.e., $u_{U_{i-1}}(\theta_b, NF, s_b) = u_{U_i}(\theta_b, NF, s_b) =$

[4] It is a standard practice to consider external incentives and several works have adhered to this model [26], [25].

$0, \theta_b \in \Theta_{U_i}$ and $s_b \in S_{U_i}$. We analyze the payoff of U_{i-1} when it has chosen F:

A. If **N** had chosen an *uncorrupt* U_i, then the payoffs are defined as follows:
(a) Instantaneous Response, i.e., $t \rightarrow 0$: If U_i *accepts the payment*, then U_{i-1} gets processing fee $f(\alpha_{i-1})$ from its preceding neighbour U_{i-2} and U_i gets α_{i-1} coins from U_{i-1}. If U_i *rejects the payment* then none of them gains anything, i.e, $u_{U_{i-1}}(uco, F, Rt) = u_{U_i}(uco, F, Rt) = 0$.

(b) Delayed Response, i.e., $0 < t < t_{i-1}$: If U_i *waits and then accepts the payment after t units* then $u_{U_{i-1}}(uco, F, W \ \& \ Ac) = f(\alpha_{i-1}) - o_{i-1}^{t,\alpha_{i-1}}$. U_{i-1} can earn $f(\alpha_{i-1})$ only after U_i resolves the payment but it suffers a loss due to α_{i-1} coins remaining locked in channel $id_{i-1,i}$. $O(r_{U_{i-1}}, t, \alpha_{i-1})$ is the opportunity cost of locked coins, also denoted as $o_{i-1}^{t,\alpha_{i-1}}$. Simultaneoulsy, U_i loses the opportunity to earn profit by utilizing α_{i-1} coins for the next t unit of time. The expected profit that U_i could have made using α_{i-1} within the next t units is $O(r_{U_i}, t, \alpha_{i-1})$, also denoted as $o_i^{t,\alpha_{i-1}}$. Thus the payoff $u_{U_i}(uco, F, W \ \& \ Ac) = \alpha_{i-1} - o_i^{t,\alpha_{i-1}}$.
If U_i *waits and then rejects the payment after time t units* then the payoff of U_{i-1}, $u_{U_{i-1}}(uco, F, W \ \& \ Rt) = -o_{i-1}^{t,\alpha_{i-1}}$, and payoff of U_i, $u_{U_i}(uco, F, W \ \& \ Rt) = -o_i^{t,\alpha_{i-1}}$.

(c) U_i *griefs*: If U_i fails to respond within time t_i, U_{i-1} will close the channel by going on-chain. U_{i-1} cannot utilize α_{i-1} coins locked in the off-chain contract. The opportunity cost is $O(r_{U_{i-1}}, t_{i-1}, \alpha_{i-1})$, also denoted as $o_{i-1}^{t_{i-1},\alpha_{i-1}}$. Additionally, due to closure of channel, U_{i-1} fails to utilize the residual capacity $remain(U_{i-1}, U_i)$ for the next $T - (t_{i-1} + t_{contract\_initiate} - t_{i-1,i})$ unit of time, where $t_{i-1,i}$ is the timestamp at which channel $id_{i-1,i}$ was opened and $t_{contract\_initiate}$ is the current timestamp at which the off-chain contract got initiated in the channel. We use a shorter notation $\tilde{t}_{i-1,i}$ to denote $T - (t_{i-1} + t_{contract\_initiate} - t_{i-1,i})$. The opportunity cost of the remaining balance is $O(r_{U_{i-1}}, T - (t_{i-1} + t_{contract\_initiate} - t_{i-1,i}), remain(U_{i-1}, U_i))$ or $o_{i-1}^{\tilde{t}_{i-1,i}, remain(U_{i-1},U_i)}$. Along with that U_{i-1} has to pay the transaction fee M for settling on-chain. Hence, payoff $u_{U_{i-1}}(uco, F, Gr) = -o_{i-1}^{t_{i-1},\alpha_{i-1}} - o_{n-1}^{\tilde{t}_{i-1,i}, remain(U_{i-1},U_i)} - M$.
If U_{i-1} had previously transferred coins to U_i then $remain(U_i, U_{i-1}) > 0$. In that case, U_i incurs a loss $O(r_{U_i}, T - (t_{i-1} + t_{contract\_initiate} - t_{i-1,i}), remain(U_i, U_i - 1))$, also denoted as $o_i^{\tilde{t}_{i-1,i}, remain(U_i,U_{i-1})}$, due to closure of channel after timeout period t_{i-1}. Additionally, it loses $o_i^{t_{i-1},\alpha_{i-1}}$, as it fails to earn and utilize the coins for other purpose. Hence, payoff $u_{U_i}(uco, F, Gr) = -o_i^{\tilde{t}_{i-1,i}, remain(U_i,U_{i-1})} - o_i^{t_{i-1},\alpha_{i-1}}$. Since U_i doesn't go on-chain for settling the transaction, we do not subtract M from the payoff.

B. If **N** had chosen an *corrupt* node, then the latter executes a self-payment of amount α via a path of length n. Thus we analyze this as a game between U_{n-1} and U_n, where the latter is the corrupt node. The amount forwarded is $\alpha + \sum_{i=1}^{n-1} f(\alpha_i)$, where, $f(\alpha_i)$ is the fee charged by an intermediate node U_i. Since the cost incurred per payment is C and the corrupt node has to keep α coins locked for time $t_{n-1} = D$, the bribe offered must compensate for all these costs. The amount of bribe offered by the attacker is L where $L = \alpha + C + I_{D,\alpha}$, where $I_{D,\alpha} \approx 2o_n^{D,\alpha}$. Since the purpose of U_n is to mount attack, it would not be interested in performing payments like other participants. This implies that U_n has not accepted any payment from U_{n-1} and $remain(U_n, U_{n-1}) = 0$. We analyze each case as follows:
(a) Instantaneous Response, i.e., $t \rightarrow 0$: If U_n *accepts the payment* then it ends up losing approximatey $\sum_{i=1}^{n-1} f(\alpha_i)$, as it needs to pay $(n-1)$ intermediaries. It had already incurred a cost C. U_{n-1} has successfully forwarded the amount. Thus, payoffs are $u_{U_{n-1}}(co, F, Ac) = f(\alpha_{n-1})$ and $u_{U_n}(co, F, Ac) = -C - \sum_{i=1}^{n-1} f(\alpha_i)$. If U_i *or* U_n *rejects the payment* then $u_{U_{n-1}}(co, F, Rt) = 0$ and $u_{U_n}(co, F, Rt) = -C$.

(b) Delayed Response, i.e., $0 < t \leq D - \delta$: If U_n *waits and then accepts the payment after t units* then payoff of U_{n-1} is same as the payoff it had obtained when U_n is not corrupt and chooses to wait and accept the payment. Thus $u_{U_{n-1}}(co, F, W \ \& \ Ac) = f(\alpha_{n-1}) - o_{n-1}^{t,\alpha_{n-1}}$. $\eta_n^{t,\alpha_{n-1}}$ defines the net profit received by U_n for keeping α_{n-1} coins unutilized till time t, where:

$$\eta_n^{t,\alpha_{n-1}} = \begin{cases} -C - O(r_{U_n}, t, \alpha_{n-1}), 0 < t < D - \delta \\ L - C - O(r_{U_n}, D - \delta, \alpha_{n-1}), \\ \text{otherwise} \end{cases} \tag{1}$$

If U_n delays till time $t < D - \delta$, it loses the setup cost and the revenue had it utilized α_{n-1} for t units of time. If U_n delays for at least $D - \delta$, it gets paid for the work done, i.e. $\eta_n^{D-\delta,\alpha_{n-1}}$. Since $\delta \rightarrow 0$, $\eta_n^{D-\delta,\alpha_{n-1}} \approx \eta_n^{D,\alpha_{n-1}} = L - C - O(r_{U_n}, D, \alpha_{n-1})$. But $I_{D,\alpha_{n-1}} \approx 2o_n^{D,\alpha_{n-1}}$, which implies $L - C - o_n^{D,\alpha_{n-1}} = \alpha + C + I_{D,\alpha_{n-1}} - C - o_n^{D,\alpha_{n-1}} \approx \alpha + o_n^{D,\alpha_{n-1}}$. Upon accepting a self-payment, it ends up paying a processing fee to $n-1$ intermediaries. Thus, the payoff of U_n, $u_{U_n}(co, F, W \ \& \ Ac) = -\sum_{i=1}^{n-1} f(\alpha_i) + \eta_n^{t,\alpha_{n-1}}$.
If U_n *waits and then rejects the payment after t units* then $u_{U_{n-1}}(co, F, W \ \& \ Rt) = -o_{n-1}^{t,\alpha_{n-1}}$ and $u_{U_n}(co, F, W \ \& \ Rt) = \eta_n^{t,\alpha_{n-1}}$.

(c) U_n *griefs*: It gets an incentive L coins from attacker. The loss is summation of C, which is the cost for mounting the attack, and the opportunity cost $o_n^{D,\alpha_{n-1}}$. Since the channel is unilaterally funded by U_{n-1},

$remain(U_n, U_{n-1}) = 0$. Thus there is no loss associated due to closure of channel. The payoff of U_{n-1} is the same as the payoff it had obtained when U_n is uncorrupt and chooses to grief. Thus, $u_{U_{n-1}}(co, F, Gr) = -o_{n-1}^{D,\alpha_{n-1}} - o_{n-1}^{\tilde{t}_{n-1,n}, remain(U_{n-1},U_n)} - M$ and $u_{U_n}(co, F, Gr) = L - C - o_n^{D,\alpha_{n-1}} = \eta_n^{D,\alpha_{n-1}}$.

2) Game Analysis: We infer from the payoff model that the corrupt node can select either of the strategies for mounting the attack - (i) *Reject the payment just before lock time D elapses*: U_n rejects the conditional payment forwarded by U_{n-1} off-chain just before the contract's lock time elapses. (ii) *U_n does not respond*: This is as per the conventional definition of griefing. U_{n-1} closes the channel unilaterally after the contract's lock time expires.

The expected payoff of U_{i-1} is calculated by applying backward induction on the game tree Γ_{HTLC}. If U_{i-1} plays *F*; an *uncorrupt* U_i chooses *Ac* as its best response since $u_{U_i}(uco, F, Ac) \geq u_{U_i}(uco, F, s'),\ \forall s' \in S_{U_i}$; a corrupt node (also U_n) can choose either to *grief* or *Wait & Reject* at $D - \delta$ as its best response since $u_{U_n}(co, F, Gr) = u_{U_n}(co, F, W\ \&\ Rt$ at time $D - \delta) \geq u_{U_n}(co, F, s''), \forall s'' \in S_{U_n}$. A corrupt node applies mixed strategy, either choosing to *Grief* with probability $1 - q$ or it can *Wait and Reject* at time $D - \delta$ with probability q. The expected payoff of U_{i-1} for selecting *F*, denoted as $\mathbb{E}_{U_{i-1}}(F)$, is $\theta_i\Big(-qo_{n-1}^{D-\delta,\alpha_{n-1}} + (1-q)(-o_{n-1}^{D,\alpha_{n-1}} - o_{n-1}^{\tilde{t}_{n-1,n}, remain(U_{n-1},U_n)} - M)\Big) + (1 - \theta_i)f(\alpha_{i-1})$ and expected payoff for selecting *NF*, denoted as $\mathbb{E}_{U_{i-1}}(NF)$, is 0.

Since $\delta \rightarrow 0$, we consider $o_{n-1}^{D-\delta,\alpha_{n-1}} \approx o_{n-1}^{D,\alpha_{n-1}}$. If $\mathbb{E}_{U_{i-1}}(F) > \mathbb{E}_{U_{i-1}}(NF)$ then $U_{i-1}'s$ best response is *F* else it chooses *NF*. We derive that U_{i-1} chooses *F* if $\theta_i < \frac{f(\alpha_{i-1})}{o_{n-1}^{D,\alpha_{n-1}} + f(\alpha_{i-1}) + (1-q)\Big(o_{n-1}^{\tilde{t}_{n-1,n}, remain(U_{n-1},U_n)} + M\Big)}$, else it chooses *NF*; *corrupt* U_i can either choose *Grief* or *Wait & Reject* at time $D - \delta$; *uncorrupt* U_i chooses *Accept*; is a perfect Bayesian equilibrium.

V. Hashed Timelock Contract with Griefing Penalty or HTLC-GP

The protocol Hashed Timelock Contract with Griefing Penalty or *HTLC-GP* [12] has been developed on top of *HTLC*. The concept of the griefing penalty is used in the protocol for countering the griefing attack. If the party griefs, it gets penalized, and the amount locked is distributed as compensation amongst the affected parties. The total penalty charged is proportional to the summation of the collateral locked in each off-chain contract instantiated on the channels routing payment. *Collateral locked* in an off-chain contract is the product of coins locked in the off-chain contract and timeout period.

HTLC-GP is a two-round protocol where the first round, or *Cancellation round*, involves locking of penalty. The round is initiated by the payee and proceeds in the reverse direction. The penalty locked by a party serves as a guarantee against a payment forwarded. The second round, or the *Payment round*, involves locking the payment value in off-chain contracts from payer to payee. We explain the protocol with the help of an example shown in Fig. 4. *Alice* wants to transfer b units to *Bob*. Each party that forwards a payment must be guaranteed by its counterparty to receive compensation if there is an incidence of a griefing attack. The compensation charged must be proportional to the collateral locked in the path. We define this proportionality constant as the *rate of griefing-penalty per unit time*, denoted as γ. The first round termed as *Cancellation round* proceeds in the following way: *Bob* has to lock $\gamma bD_1 + \gamma bD_2 + \gamma bD$ coins for duration D. This amount is the cumulative penalty to be distributed among *Alice, Dave* and *Charlie*, if Bob griefs. After *Charlie* receives the cancellation contract, he locks $\gamma bD_1 + \gamma bD_2$ in the contract formed with *Dave* for D_2 units. The latter locks γbD_1 coins in the contract formed with *Alice* for D_1 units of time. The second round, termed as *Payment round* proceeds similarly as in *HTLC*. Payment value b is forwarded from *Alice* to *Bob* via the intermediaries. Since the least timeout period is D, one might question why *Bob* must take into account the lock time of the other contracts while locking penalty. If *Bob* locks $3\gamma bD$ coins, *Charlie* locks $2\gamma bD_2$ and *Dave* locks γbD_1 as penalty, and *Bob* griefs, then *Charlie* keeps the compensation γbD and forwards $2\gamma bD$ to *Dave*. *Dave* is greedy and refuses to cancel the off-chain contract with *Charlie*. After elapse of D_2, he goes on-chain and claims $2\gamma D_2$. Since $D_2 > D$, *Charlie* incurs a loss of $2\gamma b(D_2 - D)$. Thus, we account for the lock time of each contract while calculating compensation to prevent the loss of coins of uncorrupt parties.

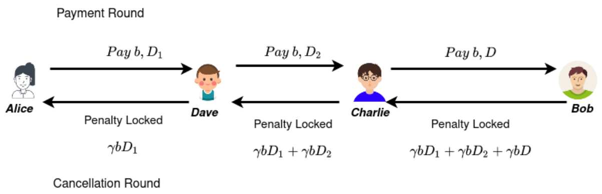


Fig. 4: Formation of contract in *HTLC-CP*

Suppose *Bob* griefs. He has to pay a compensation of $\gamma bD_1 + \gamma bD_2 + \gamma bD$ units to *Charlie*, as per the terms of the contract. After D expires, *Charlie* goes on-chain. He closes the channel, unlocks b coins, and claims compensation. He requests *Dave* to cancel the off-chain contract, offering a compensation of $\gamma bD_1 + \gamma bD_2$. *Dave* cancels the contract off-chain, unlocks b units from the contract, and claims the compensation from *Charlie*. If *Charlie* decides to grief, *Dave* can claim the compensation by going on-chain and closing the channel. *Dave* requests *Alice* to cancel the contract by offering a compensation of γbD_1. Except for *Bob*, none of the parties lose coins. In the next section, we formulate a game model for griefing attacks in *HTLC-GP* and study its effectiveness.

VI. Analysis of griefing attack in HTLC-GP

A. System Model

For payment of α from S to R via $(\kappa - 1)$ intermediaries, where $\kappa \in \mathbb{N}, \kappa \leq n$, we denote the cumulative compensation to be locked by U_i if U_{i-1} forwards α_{i-1} as $Z_{\alpha_{i-1},i}, i \in [1, \kappa]$ where $Z_{\alpha_{i-1},i} = Z_{\alpha_{i-2},i-1} + c_{\alpha_{i-1},i-1}$;

$c_{\alpha_{i-1},i-1}$ is the compensation charged by node U_{i-1} and $Z_{\alpha_{i-2},i-1}$ is used for compensating other nodes $U_j, j < i$. Note that $Z_{val,0} = 0, \forall val \in \mathbb{R}^+, \alpha_{\kappa-1} = \alpha$. $c_{\alpha_{i-1},i-1}$ charged by a node U_{i-1}, must be proportional to the collateral it has locked in the off-chain contract formed with node U_i for timeout period $t_{i-1}, i \in [1,\kappa]$. $t_{i-1} = t_i + \Delta$, $t_{\kappa-1} = D$ and $c_{\alpha_{i-1},i-1} = \gamma\alpha_{i-1}t_{i-1}$, γ being the rate of griefing-penalty per unit time. An off-chain contract between node U_{i-1} and U_i requires U_{i-1} locking an α_{i-1} coins and U_i must lock $Z_{\alpha_{i-1},i}$ coins. Here $Z_{\alpha_{\kappa-1},\kappa}$ denoted as $Z_\alpha = \sum\limits_{i=0}^{\kappa-1} c_{\alpha_i,i}$ is locked by U_κ as guaranteed compensation against the amount locked by party $U_{\kappa-1}$. Again, $U_{\kappa-1}$ has locked $Z_{\alpha_{\kappa-2},\kappa-1}$ with the contract formed with $U_{\kappa-2}$ for time $t_{\kappa-1}$.

Change in System Assumption: The assumptions taken here is same as in Section IV-A, except that the payment channel is considered to be dual-funded. This implies that in channel $id_{i-1,i}, i \in [1,\kappa]$, both the parties U_{i-1} and U_i lock coins i.e., $locked(U_{i-1}, U_i) > 0$ and $locked(U_i, U_{i-1}) > 0$.

B. Attacker Model

If a corrupt node routes a self-payment via maximum allowed path length n, then the recipient (i.e., U_n) has to lock extra coins as guarantee. Cost of the attack increases. However, the attacker does not increase the incentive offered per attack. Thus, U_n is forced to distribute α coins between the cumulative penalty locked in the contract formed with U_{n-1} and the amount to be forwarded for payment. This implies α is the summation of transaction value $v + \sum\limits_{i=1}^{n-1} f(v_i)$ and the cumulative penalty $Z_v = \sum\limits_{i=0}^{n-1} c_{v_i,i} = \gamma \sum\limits_{j=0}^{n-1} v_j t_j$ where $v_j = v_0 - \sum\limits_{k=1}^{j}, j \in [0, n-1]$. Since $\sum\limits_{i=1}^{n-1} f(v_i) << v$, we consider $v_0 + Z_v \approx v + Z_v = \alpha$ or, $v = \frac{\alpha}{1+\gamma \sum\limits_{j=0}^{n-1} t_j}$. U_n executes a self-payment of v coins.

C. Game Model

Given a payment is routed via a path of length κ using *HTLC-GP*, U_κ locks penalty in the contract formed with $U_{\kappa-1}$ in the first round locking phase. However, the former has the power to unlock the coins anytime by releasing the preimage of the cancellation hash. $U_{\kappa-1}$ will accept the contract if it thinks that the expected payoff upon forwarding the payment contract in the second round will be greater than the expected payoff on not forwarding the same. Only then it will lock the penalty in the off-chain contract with $U_{\kappa-2}$. This holds for any pair U_{i-1} and U_i in path P. If U_{i-1} accepts to form a contract with U_i in the first round of *HTLC-GP*, it implies that it will forward the conditional payment to U_i in the second round as well. Thus we merge both the first and second round while studying the interaction between any two parties U_{i-1} and U_i. We analyze the payoff assuming both parties lock their coins in a single off-chain contract instead of two separate contracts. We adapt the model Γ_{HTLC} and propose the two-party game model $\Gamma_{HTLC-GP}$ for griefing attack in *HTLC-GP*. The extensive form of the game $\Gamma_{HTLC-GP}$ is shown in Fig. 1 in Section II of Supplemental File.

1) Preference Structure: When U_{i-1} chooses not to forward, both U_{i-1} and U_i receives a payoff 0 since no off-chain contract got established, i.e., $u_{U_{i-1}}(\theta_b, NF, s_b) = u_{U_i}(\theta_b, NF, s_b) = 0, \theta_b \in \Theta_{U_i}$ and $s_b \in S_{U_i}$. We analyze the payoff of each case when U_{i-1} chooses to forward:

A. If **N** had chosen an *uncorrupt* U_i, then the payoffs are defined as follows:
(a) Instantaneous Response, i.e., $t \rightarrow 0$: Upon instant acceptance or rejection of payment, the payoffs are the same as that observed in Γ_{HTLC}.

(b) Delayed Response, i.e., $0 < t < t_i$: If U_i *waits and then accepts the payment after* t *units* then $u_{U_{i-1}}(uco, F, W \ \& \ Ac) = f(\alpha_{i-1}) - o_{i-1}^{t,\alpha_{i-1}}$. U_i has to keep $Z_{\alpha_{i-1},i}$ coins locked in the contract established in channel $id_{i-1,i}$. Thus, it faces loss not only due to delay in claiming α_{i-1} coins from U_{i-1} but also due to unutilization of $Z_{\alpha_{i-1},i}$. The expected profit that could have been made using α_{i-1} and $Z_{\alpha_{i-1},i}$ within the next t units is $O(r_{U_i}, t, \alpha_{i-1})$, also denoted as $o_i^{t,\alpha_{i-1}}$, and $O(r_{U_i}, t, Z_{\alpha_{i-1},i})$, denoted as $o_i^{t,Z_{\alpha_{i-1},i}}$. Thus, $u_{U_i}(uco, F, W \ \& \ Ac) = \alpha_{i-1} - o_i^{t,\alpha_{i-1}} - o_i^{t,Z_{\alpha_{i-1},i}}$. If U_i *waits and then rejects the payment after* t *units* then the payoff of U_{i-1} is $u_{U_{i-1}}(uco, F, W \ \& \ Rt) = -o_{i-1}^{t,\alpha_{i-1}}$ and payoff of U_i is $u_{U_i}(uco, F, W \ \& \ Rt) = -o_i^{t,\alpha_{i-1}} - o_i^{t,Z_{\alpha_{i-1},i}}$.

(c) U_i *griefs*: Since U_{i-1} can claim a compensation of $Z_{\alpha_{i-1},i}$ by going on-chain and closing the channel. Payoff $u_{U_{i-1}}(uco, F, Gr) = -(o_{i-1}^{t_{i-1},\alpha_{i-1}} + o_{i-1}^{\tilde{t}_{i-1,i},remain(U_{i-1},U_i)} + M) + Z_{\alpha_{i-1},i}$.
U_i incurs loss of $Z_{\alpha_{i-1},i}$ coins to compensate U_{i-1}. It fails to earn revenue due to non-utilization of $Z_{\alpha_{i-1},i}$ coins in channel $id_{i-1,i}$ for period t_{i-1}. The additional loses suffered are due to inability to claim α_{i-1} coins and using it within time t_{i-1} and failure in utilizing the residual capacity $remain(U_i, U_{i-1})$ for the next $T - (t_{i-1} + t_{contract\_initiate} - t_{i-1,i})$ units. Payoff of U_i is $u_{U_i}(uco, F, Gr) = -o_i^{\tilde{t}_{i-1,i},remain(U_i,U_{i-1})} - o_i^{t_{i-1},\alpha_{i-1}} - o_i^{t_{i-1},Z_{\alpha_{i-1},i}} - Z_{\alpha_{i-1},i}$.

B. If **N** had chosen an *corrupt* node, then the payoffs are defined as follows:
(a) Instantaneous Response, i.e., $t \rightarrow 0$: Upon instant acceptance or rejection of payment, the payoffs are the same as that observed in Γ_{HTLC}.

(b) Delayed Response, i.e., $0 < t \leq D - \delta$: If U_n waits and then accepts the payment after t units then payoff of U_{n-1} is $u_{U_{n-1}}(co, F, W \ \& \ Ac) = f(v_{n-1}) - o_{n-1}^{t,v_{n-1}}$. The loss observed is due to delay in claiming of v_{n-1} coins as U_n delays in resolving payment. U_n keeps $v + Z_v \approx \alpha$ coins locked for mounting the attack

and receives a bribe L. The value $\eta_n^{t,\alpha}$ is the same as defined in Eq.1. Upon accepting a self-payment of amount v, the corrupt node ends up paying a processing fee to $n-1$ intermediaries. Thus, the payoff of U_n is $u_{U_n}(co, F, W\ \&\ Ac) = -\sum\limits_{i=1}^{n-1} f(v_i) + \eta_n^{t,\alpha}$.
If U_n waits and then rejects the payment after t units, then the payoff of U_{n-1}, $u_{U_{n-1}}(co, F, W\ \&\ Rt) = -o_{n-1}^{t,v_{n-1}}$ and $u_{U_n}(co, F, W\ \&\ Rt) = \eta_n^{t,\alpha}$. When $t \approx D - \delta$ where $\delta \to 0, \eta_n^{t,\alpha}$ attains the maximum value.

(c) U_n *griefs*: It gets an incentive L from the attacker, but at the same time loses Z_v in order to compensate the affected parties. U_{n-1} loses channel and the expected revenue due to coins remaining locked but gets the compensation Z_v. Thus, the payoffs of U_{n-1} and U_n are $u_{U_{n-1}}(co, F, Gr) = -o_{n-1}^{D,v_{n-1}} - o_{n-1}^{\tilde{t}_{n-1,n}, remain(U_{n-1},U_n)} - M + Z_v$ and $u_{U_n}(co, F, Gr) = L - C - o_n^{D,\alpha} - Z_v - o_{n-1}^{\tilde{t}_{n-1,n}, remain(U_n,U_{n-1})} = \eta_n^{D,\alpha} - Z_v - o_n^{\tilde{t}_{n-1,n}, remain(U_n,U_{n-1})}$ respectively.

2) Game Analysis: If U_{i-1} plays F; an *uncorrupt* U_i chooses Ac as its best response but a corrupt U_i will choose *Wait & Reject* at $D-\delta$ as its best response since $u_{U_i}(co, W\ \&\ Rt$ at time $D-\delta) \geq u_{U_i}(co, F, s''), \forall s'' \in S_{U_i}$. The expected payoff of U_{i-1} for selecting F, denoted as $\mathbb{E}_{U_{i-1}}(F)$, is $\theta_i(-o_{n-1}^{D-\delta,v_{n-1}}) + (1-\theta_i) f(\alpha_{i-1})$, and expected payoff for selecting *NF*, denoted as $\mathbb{E}_{U_{i-1}}(NF)$, is 0.

Since $\delta \to 0$, we consider $o_{n-1}^{D-\delta,v_{n-1}} \approx o_{n-1}^{D,v_{n-1}}$. If $\mathbb{E}_{U_{i-1}}(F) > \mathbb{E}_{U_i}(NF)$, then U_{i-1} chooses F else it aborts. We derive that U_{i-1} chooses F if $\theta_i < \frac{f(\alpha_{i-1})}{f(\alpha_{i-1}) + o_{n-1}^{D,v_{n-1}}}$, else it chooses *NF*; *corrupt* U_i chooses *Wait & Reject* at time $D-\delta$; *uncorrupt* U_i chooses *Accept*; is a perfect Bayesian equilibrium.

Comparing θ_i for which U_{i-1} chooses F *in Γ_{HTLC} and $\Gamma_{HTLC-GP}$:* Since $f(\alpha_{i-1}) + o_{n-1}^{D,v_{n-1}} < o_{n-1}^{D,\alpha_{n-1}} + f(\alpha_{i-1}) + (1-q)(o_{n-1}^{\tilde{t}_{n-1,n}, remain(U_{n-1},U_n)} + M)$, the cut-off of θ_i for which U_{i-1} will choose to forward a payment is higher in $\Gamma_{HTLC-GP}$ than in Γ_{HTLC} even if $q \to 0$. The corrupt player has to invest some amount as penalty and as a consequence, the payment amount reduces from α_{n-1} to v_{n-1}. Additionally, the corrupt player chooses to cancel the payment with U_{n-1} off-chain just before elapse of locktime. This results in less risk compared to *HTLC*. We simulate the games Γ_{HTLC} and $\Gamma_{HTLC-GP}$ in Section III of Supplemental File and the experimental results supports the mathematical deduction.

D. Effectiveness of HTLC-GP

The analysis in Section VI-C2 shows that a rational corrupt node will cancel the payment off-chain just before the contract lock time elapses i.e., at time $D-\delta$, where $\delta \to 0$. The corrupt node avoids paying any penalty but still manages to mount the attack. We cannot protect uncorrupt parties from griefing attacks unless we do not account for the intermediate delay in resolving payments. Since Bitcoin scripting language is not Turing-complete, we cannot have a single off-chain contract where we can define penalty as a function of time. There is no way to execute a transaction like this: *If t' time units have elapsed, pay amount p. If* $t'+1$ *time units have elapsed, pay amount* $p+\delta$. CheckSequenceVerify [30] imposed on the first condition of elapse of t' time unit makes the transaction eligible for broadcasting on-chain after elapse of time $t'+1$. This might lead to a race condition, and the victim might not receive proper compensation. Instead, it is desirable to construct t' off-chain contracts, each accounting for a delay after every t' interval. The timeout period of the i^{th} contract is $i\frac{D}{t'}, i \in [1, t']$. However, multiple off-chain contracts for a single payment reduce the network throughput. Additionally, it is risky to have off-chain contracts with a shorter timeout period as it will lead to the abrupt closure of the payment channel and compromise the security of the protocol [31]. *We conjecture that it is impossible to design an efficient protocol that will penalize the attacker and compensate the victims of a griefing attack with the current Bitcoin scripting system.*

Instead of focussing on the victims, we analyze the protocol from the attacker's point of view. The latter has an objective of maximizing the damage by locking as much network liquidity possible for the given budget $\mathcal{B}_{EX}$. The attacker will continue to invest in the network if the return on investment is good enough. If the return on investment diminishes, the attacker will refrain from mounting the attack and instead prefer to invest in another activity. In *HTLC-GP*, the introduction of penalty led to locking extra coins, increasing the cost of the attack. The attacker will be able to corrupt fewer nodes compared to *HTLC*. We define a metric, *capacity locked in a path routing payment*, that indirectly determines the success rate of the attack [9], [10]. It is the summation of the coins locked in the off-chain contract instantiated on the channel forming the path. Ignoring the processing fee (negligible quantity), assuming all the payments executed are of value α and the bribe offered per instance is L, the attacker can corrupt $\frac{\mathcal{B}_{EX}}{L}$ nodes in the networks. We assume that for any node $U \in V$, $\mathbb{E}_U(\text{F}) > \mathbb{E}_U(\text{NF})$. So each self-payment gets routed and reaches the payee.

***Claim** 1: Given the total budget of the attack is $\mathcal{B}_{EX}$, incentive per attack being L, transaction value per payment being α, HTLC timeout period is D, time taken to settle a transaction on-chain being Δ, n is the maximum allowed path length and a corrupt recipient rejects the payment at time $t' = D - \delta$, where $\delta \to 0$, the capacity locked upon using HTLC-GP is less than the capacity locked in HTLC, the loss percent being* $\frac{\gamma n(\frac{D}{2} + \frac{\Delta(n-2)}{6})}{1 + \gamma n(D + \frac{(n-1)\Delta}{2})}$

Proof: Discussed in Section IV of Supplemental File. ■

We observe that the loss percent $\frac{\gamma n(\frac{D}{2} + \frac{\Delta(n-2)}{6})}{1 + \gamma n(D + \frac{(n-1)\Delta}{2})}$ is dependent on γ [5]. If γ is too low, the loss percent is not substantial and the attacker can still consider stalling the network. Hence the payment protocol *HTLC-GP* is *weakly effective* in disincentivizing the attacker.

[5] It may not be always possible for a corrupt node to find a path of maximum allowed length for mounting the attack. In that case, the attacker might ask the corrupt node to get a the longest feasible path for routing payment. In our analysis, we consider the worst-case scenario where the corrupt node is able to route all its transaction via paths of length n.

VII. Guaranteed minimum compensation

If the incidence of griefing attack increases in the network, γ can be increased. However, the disadvantage of increasing the rate of griefing penalty means uncorrupt nodes have to put a higher amount at stake for routing small-valued payments. The success rate of payments decreases due to a lack of liquidity in the channels. Our objective is to increase the cost of the attack without forcing uncorrupt parties to lock high penalties. Since corrupt parties are asked to route self-payment via the longest available path, cost of the attack increases if the maximum path length allowed for routing payments decreases [31]. This would lead to locking of less coins in the network by the corrupt parties. However, an abrupt reduction in the maximum allowed path length may lead to higher failure in executing transactions. Thus we must design a mechanism by which one can adjust the maximum allowed path length based on the rate of incidence of griefing attacks in the network.

The major source of earning for a node is the processing fee by routing transactions. If there is a griefing attack, then the affected parties fail to earn due to locked collateral. We introduce a new parameter ζ, termed as *Guaranteed Minimum Compensation*. Based on the data provided [32], the fee earned by each node on a single day is quite low compared to the amount routed. Thus, we set ζ in the range $[0, 1)$ to avoid over-compensation.

Adjusting the parameters based on ζ: Let the maximum allowed path length in the new model be denoted as $\tilde{n}^{\zeta,k}$. If payment forwarded by U_{i-1} is α, U_i must lock a minimum cumulative penalty $i\zeta\alpha, i \in [1, \kappa]$ where $\kappa \leq \tilde{n}^{\zeta,k}$. Each party $U_j, j \in [0, i)$ is entitled to receive a compensation $\zeta\alpha$. For a given rate of penalty, let the maximum cumulative penalty an uncorrupt recipient has to lock when payment is routed via path of length $\tilde{n}^{\zeta,k}$ be $Z_{\alpha,max} = k\alpha$, where $k \in \mathbb{R}^+$. We discuss a method for fixing the value k in Section V.A of Supplemental File. Given the conditions, we discuss how to calculate the maximum allowed path length for a payment and its corresponding rate of griefing-penalty.

a. If all the nodes from U_0 to $U_{\tilde{n}^{\zeta,k}-1}$ charge a minimum compensation of $\zeta\alpha$, then the minimum value of $Z_{\alpha,max}$ is $\tilde{n}^{\zeta,k}\zeta\alpha$. We use this relation to calculate the maximum allowed path length $\tilde{n}^{\zeta,k}$ for routing payment.

   ***Proposition** 1: Given the maximum cumulative griefing-penalty for a payment α is $k\alpha$, and the guaranteed minimum compensation ζ, the maximum allowed path length $\tilde{n}^{\zeta,k}$ is $\frac{k}{\zeta}$.*

   Proof: Discussed in Section V.B of Supplemental File ■

b. Once the maximum path length is adjusted, the rate of griefing-penalty γ ceases to be an independent variable. We call this new rate of griefing-penalty $\gamma^{\zeta,k}$ and provide an expression for calculating the same.

   ***Proposition** 2: Given the guaranteed minimum compensation ζ, ratio of maximum cumulative griefing penalty and the transaction amount is k, and* HTLC-GP *timeout period D, the rate of griefing-penalty $\gamma^{\zeta,k}$ is $\frac{2\zeta^2}{2\zeta D+\Delta(k-\zeta)}$.*

   Proof: Discussed in Section V.C of Supplemental File ■

VIII. Modifying HTLC-GP to HTLC-GP$^\zeta$

Given a payment request (S, R, α), R decides on the maximum cumulative penalty $k\alpha$ for a given ζ. S find a path of length κ where $\kappa \leq \tilde{n}^{\zeta,\kappa} = \frac{k}{\zeta}$ for routing the payment where $U_0 = S$ and $U_\kappa = R$. $t_{\kappa-1}$ is set to D and the rate of griefing-penalty $\gamma^{\zeta,k}$ is calculated accordingly. The total amount that the payer needs to transfer is $\tilde{\alpha} = \alpha + \sum_{j=1}^{\kappa-1} f(\alpha_j)$. We denote each $\alpha_i = \tilde{\alpha} - \sum_{k=1}^{i} f(\alpha_k), i \in [0, \kappa-1], \alpha_0 = \tilde{\alpha}$. Each node U_i samples a pair of secret key and public key (sk_i, pk_i), the public key of each node is used to encrypt the information of establishing contract with the neighboring node.

A. Protocol Description

The phases of HTLC-GP$^\zeta$ is similar to *HTLC-GP* [12].
`(a)Pre-processing Phase`: If the exact path length κ is used for routing payment, U_1 locks a penalty $\gamma^{\zeta,k}\alpha_0 t_0$ with U_0 and the former can easily figure out the identity of the sender. To prevent violation of privacy, U_0 randomizes the exact path length using a random function ϕ, and shares $\phi(\kappa)$ with U_κ. The latter calculates the cumulative penalty $\gamma^{\zeta,k}\phi(\kappa)\alpha D$ used for establishing the *cancellation contract*. A routing attempt cost ψ is added such that $\gamma^{\zeta,k}\phi(\kappa)\alpha D \approx \gamma^{\zeta,k}((\psi+\alpha_0)t_0 + \Sigma_{j=1}^{\kappa-1}\alpha_j t_j)$. This acts like a blinding factor and U_1 cannot infer the identity of the sender from the penalty it locks in the off-chain contract formed with U_0. The following steps of the protocols are executed:
(i) U_κ samples two random numbers x and r where $x \neq r$. It constructs the payment hash $H = \mathcal{H}(x)$ and the cancellation hash $Y = \mathcal{H}(r)$.
(ii) The payee shares both the hashes H and Y with the U_0. The cumulative griefing-penalty to be locked by U_1 is $\text{cgp}_0 = \gamma^{\zeta,k}(\psi + \tilde{\alpha})t_0$. The cumulative griefing-penalty to be locked by any other node $U_{i+1}, i \in [1, \kappa-1]$ is $\text{cgp}_i = \gamma^{\zeta,k}.(\Sigma_{j=1}^{i}(\alpha_j t_j) + (\alpha_0 + \psi)t_0)$.
(iii) The payer uses standard onion routing [33] for propagating the information needed by each node $U_i, i \in [1, \kappa]$, across the path P. U_0 sends $M_0 = E(\dots E(E(E(\phi, Z_\kappa, pk_\kappa), Z_{\kappa-1}, pk_{\kappa-1}), Z_{\kappa-2}, pk_{\kappa-2}), Z_1, pk_1)$ to U_1, where $Z_i = (H, Y, \alpha_i, t_{i-1}, \text{cgp}_{i-1}, U_{i+1}), i \in [1, \kappa-1]$ and $Z_\kappa = (H, Y, \alpha_{\kappa-1}, t_{\kappa-1}, \text{cgp}_{\kappa-1}, null)$. Here $M_{i-1} = E(M_i, Z_i, pk_i)$ is the encryption of the message M_i and Z_i using public key pk_i, $M_\kappa = \phi$.
(iv) U_1 decrypts M_0, gets Z_1 and M_1. $M_1 = E(\dots E(E(E(\phi, Z_\kappa, pk_\kappa), Z_{\kappa-1}, pk_{\kappa-1}), Z_{\kappa-2}, pk_{n-2}), \dots, Z_2, pk_2)$ is forwarded to the next destination U_2. This continues till party U_κ gets $E(\phi, Z_\kappa, pk_\kappa)$.

`(b) Two-Round Locking Phase`: It involves the following two rounds:

- *Establishing Cancellation Contract*: U_κ initiates this round and each player $U_i, i \in [1, \kappa]$ locks their respective cumulative griefing-penalty cgp_{i-1}.
  (i) U_κ decrypts and gets Z_κ. It checks $\gamma^{\zeta,k}\phi(\kappa)\alpha D \overset{?}{=} cgp_{\kappa-1}$ and $\alpha_{\kappa-1} \overset{?}{=} \alpha$. If this holds, the payer sends a

contract formation request to $U_{\kappa-1}$. The latter knows that it has to lock $\alpha_{\kappa-1}$ with U_κ in the second round, so it checks that given the belief θ_κ regarding U_κ's type, the expected payoff on forwarding the contract in second round is greater than 0. If so, it accepts the terms of the off-chain contract from U_κ, with the latter locking $cgp_{\kappa-1}$ coins.

(ii) For any other party $U_i, i \in [1, \kappa-1]$, it first checks $cgp_i - \gamma^{\zeta,k}\alpha_i t_i \overset{?}{=} cgp_{i-1}$. This ensures that there is sufficient coins to be locked as penalty in the contract to be formed with U_{i-1}. Next, it checks $\gamma^{\zeta,k}\alpha_i t_i \geq \zeta\alpha_i$. This check ensures that U_i is guaranteed a minimum compensation upon being affected by griefing attack. If both the condition satifies, U_i sends a request to form off-chain contract with U_{i-1}. If the latter accepts based on the belief θ_i regarding $U_i's$ nature, U_i locks cgp_{i-1}.

(iii) The off-chain contract for locking penalty in layman terms: *'U_{i+1} can withdraw the amount $cgp_i = \gamma^{\zeta,k}.(\Sigma_{j=1}^{i}(\alpha_j t_j) + (\alpha_0 + \psi)t_0)$ from the contract contingent to the release of either $x : H = \mathcal{H}(x)$ or $r : Y = \mathcal{H}(r)$ within time t_i. If the locktime elapses and U_{i+1} does not respond, U_i claims cgp_i after the locktime elapses.'*.

The pseudocode of the first round of Locking Phase for U_κ, any intermediate party $U_i, i \in [1, \kappa-1]$ and payer U_0 is stated in Procedure 1, Procedure 2 and Procedure 3 respectively.

Procedure 1: Establishing Cancellation Contract: First Round of Locking Phase for U_κ

```
Input: $(Z_\kappa, \phi(\kappa), \gamma^{\zeta,k}, \alpha)$
$U_\kappa$ parses $Z_\kappa$ and gets $H', Y', \alpha', t', \text{cgp}_{\kappa-1}$.
if $t' \geq t_{now} + \Delta$ and $\alpha' \overset{?}{=} \alpha$ and $k\alpha \overset{?}{=} cgp_{\kappa-1}$ and
  $H' \overset{?}{=} H$ and $Y' \overset{?}{=} Y$ and
  $remain(U_\kappa, U_{\kappa-1}) \geq cgp_{\kappa-1}$ then
    Send
      Cancel_Contract_Request$(H, Y, t', \text{cgp}_{\kappa-1}, \gamma^{\zeta,k})$
      to $U_{\kappa-1}$
    if acknowledgement received from $U_{\kappa-1}$ then
        $remain(U_\kappa, U_{\kappa-1}) =$
          $remain(U_\kappa, U_{\kappa-1}) - \text{cgp}_{\kappa-1}$
        establish $Cancel\_Contract(H, Y, t', \text{cgp}_{\kappa-1})$
          with $U_{\kappa-1}$
        Record $t_\kappa^{form} = current\_clock\_time$
    end
    else
        abort
    end
end
else
    abort.
end
```

- *Establishing Payment Contract*: U_0 initiates the next rounded provided it has received the cancellation contract and $cgp_0 \geq \zeta\alpha_0$. The conditional payment is forwarded till it reaches the payer U_κ. This proceeds as normal *HTLC*.

Procedure 2: Establishing Cancellation Contract: First Round of Locking Phase for $U_i, i \in [1, \kappa-1]$

```
Input: $(H', Y', t', \text{cgp}_i, \gamma^{\zeta,k})$
$U_i$ parses $Z_i$ and gets $H, Y, \alpha_i, t_{i-1}, \text{cgp}_{i-1}$.
if $\theta_{i+1} < \frac{f(\alpha_i)}{f(\alpha_i)+o_i^{t_i,\alpha_i}}$ and $H' \overset{?}{=} H$ and $Y \overset{?}{=} Y'$ and
  $t' + \Delta \overset{?}{\leq} t_{i-1}$ and $cgp_i - \gamma^{\zeta,k}\alpha_i t' \overset{?}{=} cgp_{i-1}$ and
  $\gamma^{\zeta,k}\alpha_i t' \geq \zeta\alpha_i$ and $remain(U_i, U_{i+1}) \geq \alpha_i$ and
  $remain(U_i, U_{i-1}) \geq cgp_{i-1}$ and (current_time not
  close to contract expiration time) then
    Sends acknowledgment to $U_{i+1}$ and waits for the
      off-chain contract to be established
    Send
      Cancel_Contract_Request$(H, Y, t_{i-1}, \text{cgp}_{i-1}, \gamma^{\zeta,k})$
      to $U_{i-1}$
    if acknowledgement received from $U_{i-1}$ then
        $remain(U_i, U_{i-1}) =$
          $remain(U_i, U_{i-1}) - \text{cgp}_{i-1}$
        establish $Cancel\_Contract(H, Y, t_{i-1}, \text{cgp}_{i-1})$
          with $U_{i-1}$
    end
    else
        abort
    end
end
else
    abort.
end
```

Procedure 3: Establishing Cancellation Contract: First Round of Locking Phase for U_0

```
Input: $(H', Y', t', \text{cgp}', \gamma^{\zeta,k})$
if $\theta_1 < \frac{f(\alpha_0)}{f(\alpha_0)+o_0^{t_0,\alpha_0}}$ and $t' \overset{?}{=} t_0$ and
  $cgp' \overset{?}{=} cgp_0 \geq \zeta\alpha_0$ and $H' \overset{?}{=} H$ and $Y' \overset{?}{=} Y$ and
  $remain(U_0, U_1) \geq \alpha_0$ then
    Sends acknowledgment to $U_1$
    Confirm formation of penalty contract with $U_1$
    Initiate the second round, the establishment of
      payment contract
end
else
    abort.
end
```

(i) A node $U_i, i \in [0, \kappa-1]$ forms the off-chain payment contract with U_{i+1}, locking α_i coins, if and only if U_i had accepted the formation of cancellation contract with U_{i+1} in the first round.

(ii) The off-chain contract for payment in layman terms: *'U_{i+1} can claim α_i coins contingent to the release of $x : H = \mathcal{H}(x)$ within time t_i. If U_{i+1} does not* *respond, U_i unlocks α_i coins from the contract either by releasing preimage $r : Y = \mathcal{H}(r)$ or after the locktime elapses.'*

The pseudocode of the second round of Locking Phase for a party $U_i, i \in [0, \kappa - 1]$ is stated in Procedure 4.

Procedure 4: Establishing Payment Contract: Second Round of Locking Phase for $U_i, i \in [0, \kappa - 1]$

```
Input: $(H, Y, \alpha_i, t_i)$
if (cancellation contract locking penalty has been formed between $U_i$ and $U_{i+1}$ in the first round) and $t_{i-1} \geq t_i + \Delta$ and $\alpha_i \overset{?}{=} \alpha_{i-1} + fee(U_i)$ and ($U_{i+1}$ has agreed to form the contract) and (current_time not close to contract expiration time) then
    $remain(U_i, U_{i+1}) = remain(U_i, U_{i+1}) - \alpha_i$
    establish $Payment\_Contract(H, Y, t_i, \alpha_i)$ with $U_{i+1}$
end
else
    abort
end
```

(c) Release Phase: U_κ waits for a very short duration, say μ, to receive the payment contract from $U_{\kappa-1}$. If the payment contract has been forwarded by $U_{\kappa-1}$ within μ units of time and it is correct, then U_κ releases the preimage x for payment hash H and claims the coins from $U_{\kappa-1}$. If the latter has delayed beyond μ, or the payment contract forwarded by $U_{\kappa-1}$ is invalid, U_κ releases the cancellation preimage r. In case of dispute, the payer goes on-chain and releases one of the preimages for settling the contract. The rest of the parties $U_{i+1}, i \in [0, \kappa{-}2]$ either claim the coins or cancel the payment based on the preimage released. If U_{i+1} griefs and refuses to release preimage to U_i, the former has to pay the cumulative griefing-penalty cgp_i for affecting the nodes $U_k, 0 \leq k \leq i$, so that all the nodes obtain their due compensation. We discuss the Release Phase of the protocol for node U_κ and any intermediary $U_i, i \in [1, \kappa - 1]$ in Procedure 1 and Procedure 2 under Section VI of Supplemental File.

B. Effectiveness of HTLC-GP$^\zeta$

A corrupt node can still mount the attack by canceling the payment just before the off-chain contract's lock time elapses. However, we intend to study the impact of the reduced maximum allowed path length $\tilde{n}^{\zeta,k}$ on the effective *capacity locked* by the attacker in the network. We assume that for any node $U \in V$, $\mathbb{E}_U(\text{F}) > \mathbb{E}_U(\text{NF})$ and thus each self-payment gets routed and reaches the payee.

***Claim** 2: Given the total budget of the attack is $\mathcal{B}_{EX}$, incentive per attack being L, transaction value per payment being α, HTLC timeout period is D, time taken to settle a transaction on-chain being Δ, n is the maximum allowed path length for HTLC, $\tilde{n}^{\zeta,k}$ is the maximum allowed path length for HTLC-GP$^\zeta$ for a given pair of ζ and k, and a corrupt recipient rejects the payment at time $t' = D - \mu$, where $\mu \to 0$, the capacity locked in HTLC-GP$^\zeta$ is less than the capacity locked in HTLC, the loss percent being* $\frac{n-\tilde{n}^{\zeta,k}}{(n-1)\left(1+\gamma^{\zeta,k}nD+\gamma^{\zeta,k}n\Delta\frac{n-1}{2}\right)} + \frac{\gamma^{\zeta,k}\tilde{n}^{\zeta,k}((n-1)(D+\frac{(\tilde{n}^{\zeta,k}-1)\Delta}{2})-\frac{\tilde{n}^{\zeta,k}-1}{2}(D+\frac{(2\tilde{n}^{\zeta,k})\Delta}{3}))}{(n-1)\left(1+\gamma^{\zeta,k}nD+\gamma^{\zeta,k}n\Delta\frac{n-1}{2}\right)}$

Proof: Discussed in Section VI of Supplemental File. ■

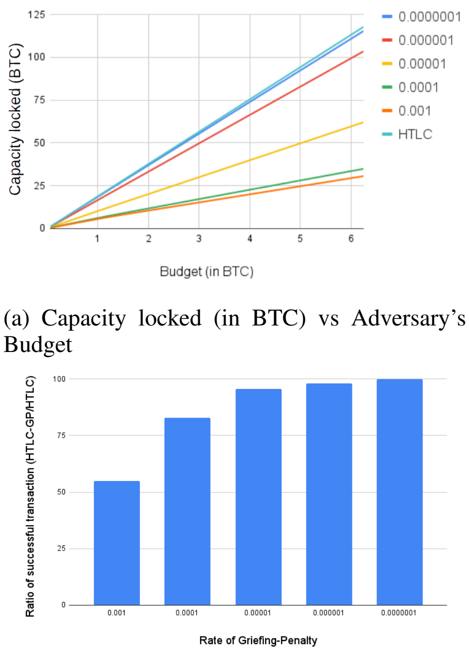
The loss percent is dominated by the term $\frac{n-\tilde{n}^{\zeta,k}}{n-1}$. For a given k, if ζ increases, the maximum path length $\tilde{n}^{\zeta,k}$ decreases, and so the loss incurred increases. In that case, the attacker would prefer to invest in other activities with higher returns rather than mount an attack on the network.

IX. Experimental analysis

(a) *Setup:* For our experiments, we use Python 3.8.2 and NetworkX, version 2.4 [34]. System configuration used is Intel Core i5-8250U CPU, Operating System: Kubuntu 20.04, Memory: 7.7 GiB of RAM. We use twelve snapshots [31] of Bitcoin Lightning Network taken over a year, starting from *September 2019*, have been used. Since our proposed strategy for countering griefing attacks requires both parties to fund the channel, we divide the channel's capacity into equal halves, allocating each half as the balance of a counterparty.

(b) *Evaluation Methodology*: We define the strategy used by the attacker in the network. The latter corrupts the nodes that are either pendant vertices or have just one channel in the network. It is easier and more cost-effective to target such peripheral nodes than nodes with high centrality. A highly central node earns a higher profit as transactions tend to get routed through such nodes. Also, the attacker needs to offer a higher incentive per attack, which may not be a good strategy. On the other hand, peripheral nodes can be easily incentivized to deviate, as they haven't gained much trust in the network. Such nodes do not expect to earn much by behaving altruistically. While analyzing the effectiveness of *HTLC-GP* and HTLC-GP$^\zeta$, we assume that U_{i-1} always forward the payment request to U_i.

The dataset and parameters used in our experiments are as follows: (i) *HTLC-GP*: The set of experiments are divided into two parts. Transaction value is varied between 10000-100000 satoshis and $\gamma \in [10^{-7}, 10^{-3}]$. In the first part, we analyze the decrease in capacity locked when a penalty is introduced. The budget of the attacker is varied between 0.05 $BTC -$ 6.25 BTC. The path length is set to $n = 20$ and D is set to 100. In the second part, we analyze the rate of the successful transaction in the absence of a griefing attack. We vary the number of transactions between 3000-9000 and path length κ is varied between 5 and 20. (ii) HTLC-GP$^\zeta$: We analyze the further decrease in capacity locked upon introduction of penalty, as well as guaranteed minimum compensation in the payment protocol for a given budget of the attacker. The budget of the attacker is varied between 0.05 BTC-6.25 BTC. The transaction value is varied between 10000 satoshis to 100000 satoshis. We vary the parameter k between 0.005 to 2. For a fixed k, ζ is varied so that path length ranges between 2 to 20. Both D and Δ are set to 100.



(a) Capacity locked (in BTC) vs Adversary's Budget

(b) Ratio of successful transaction (HTLC-GP/HTLC) upon varying γ

Fig. 5: γ is varied between 10^{-3} to 10^{-7}

A. Observations

- *Effectiveness of HTLC-GP*: The plot in Fig.5(a) shows that the capacity locked drops from 90% to 20% when γ is varied between 10^{-7} to 10^{-3}. We see a sharp decrease in capacity locked when γ increases from 10^{-6} to 10^{-5}, with the capacity locked dropping from 82% to 50%. When γ is 10^{-4}, the capacity locked drops to 25%. The plot in Fig.5(b) shows that the ratio of successful transaction executed drops to 54% when γ is 10^{-3} and it is around 99% when γ is 10^{-7}.
- *Effectiveness of HTLC-GP$^{\zeta}$*: k is varied between 0.005 and 2, and for each k, the factor ζ is varied so that the maximum path length ranges between 2 and 20. We observe that on varying k and ζ, γ varies between 10^{-7} to 10^{-3}. The drop in capacity locked in the network ranges between 18% to 46%. The drop in capacity locked is significant for the lower value of γ and the difference reduces for $\gamma > 10^{-5}$. Table II provides a comparative analysis of percentage loss in capacity between *HTLC-GP* and HTLC-GP$^{\zeta}$.

B. Discussions

- *HTLC-GP*: If γ increases, the net capacity locked by the attacker decreases but uncorrupt participants are forced to lock extra collateral for a given transaction. This results in a drop in the success rate of transactions being processed due to a lack of liquidity in channels.
- HTLC-GP$^{\zeta}$: When γ increases, percentage loss in capacity locked for *HTLC-GP* increases as well. But this is at the cost of a high failure rate of transactions. In this protocol, we observe that the capacity locked drops substantially even for lower values of γ when k and ζ are adjusted to reduce the maximum path length.

k	ζ	$\gamma^{\zeta,k}$	Maximum Path Length	γ	Maximum Path Length	Ratio of capacity locked	
		$HTLC-GP^{\zeta}$	$HTLC-GP^{\zeta}(n^{\zeta,k})$	HTLC-GP	HTLC-GP (n)	$\frac{HTLC-GP^{\zeta}}{HTLC}$	$\frac{HTLC-GP}{HTLC}$
0.005	0.00025	2.4×10^{-7}	20	2.4×10^{-7}	20	96.89%	96.89%
	0.0005	9.1×10^{-7}	10	9.1×10^{-7}	20	46.3%	89.86%
	0.0025	1.4×10^{-5}	2	1.4×10^{-5}	20	5.21%	50.7%
0.01	0.0005	4.7×10^{-7}	20	4.7×10^{-7}	20	94%	94%
	0.001	1.8×10^{-6}	10	1.8×10^{-6}	20	44.8%	81.5%
	0.005	2.8×10^{-5}	2	2.8×10^{-5}	20	5.1%	42%
0.05	0.0025	2.4×10^{-6}	20	2.4×10^{-6}	20	78%	78%
	0.005	9.1×10^{-6}	10	9.1×10^{-6}	20	38.2%	54%
	0.025	1.6×10^{-4}	2	1.6×10^{-4}	20	4.7%%	33%
0.1	0.005	4.8×10^{-6}	20	4.8×10^{-6}	20	67.5%	67.5%
	0.01	1.8×10^{-5}	10	1.8×10^{-5}	20	33.5%	45.5%
	0.05	3.3×10^{-4}	2	3.3×10^{-4}	20	4.45%	32.5%
0.25	0.0125	1.2×10^{-5}	20	1.2×10^{-5}	20	53%	53%
	0.025	4.5×10^{-5}	10	4.5×10^{-5}	20	28%	40%
	0.1125	6.9×10^{-4}	2	6.9×10^{-4}	20	4.2%	32%
0.5	0.025	2.4×10^{-5}	20	2.4×10^{-5}	20	44%	44%
	0.05	9.1×10^{-5}	10	9.1×10^{-5}	20	22.1%	38.5%
	0.2	1.1×10^{-3}	2	1.1×10^{-3}	20	3.8%	31.5%
0.75	0.0375	3.6×10^{-5}	20	3.6×10^{-5}	20	41%	41%
	0.075	1.36×10^{-4}	10	1.36×10^{-4}	20	21.2%	35.6%
	0.3	1.7×10^{-3}	2	1.7×10^{-3}	20	3.51%	30.04%
1	0.05	4.8×10^{-5}	20	4.8×10^{-5}	20	38%	38%
	0.1	1.8×10^{-4}	10	1.8×10^{-4}	20	20%	34%
	0.5	3.3×10^{-3}	2	3.3×10^{-3}	20	3.4%	29.98%
2	0.1	10^{-4}	20	10^{-4}	20	35%	35%
	0.2	3.6×10^{-4}	10	3.6×10^{-4}	20	18%	32%
	0.95	6.1×10^{-3}	2	6.1×10^{-3}	20	3.34%	29.01%

TABLE II: Capacity Locked when k and ζ is varied

(i) D is varied: The limit on the maximum timeout period in an *HTLC* is 2016 blocks [31]. So D cannot be increased indefinitely. $\gamma^{\zeta,k}$ will decrease if D increases. The capacity locked remains invariant as the cumulative penalty does not change abruptly.

(ii) ζ is varied: For a fixed value of k, ζ can be increased, reducing the maximum path length available for routing. This will increase the cost of the attack. When the majority of participants in the network adhere to non-attacking behavior, then the compensation offered can be reduced, readjusting the path length. Hence, the parameters must be chosen accordingly.

X. Scalability Analysis

We analyze the performance of *HTLC, HTLC-GP* and HTLC-GP$^{\zeta}$ on the snapshots selected in Section IX. The number of transaction request is varied between 100 to 65000. Transaction chose the shortest feasible path as per the availability of capacity in the network so the path length varied between 4 to 12. For *HTLC-GP* and HTLC-GP$^{\zeta}$, γ is varied between 10^{-6} to 0.001. We analyze the performance of the protocols in two different models - (i) All uncorrupt players are altruistic and (ii) All uncorrupt players are rational

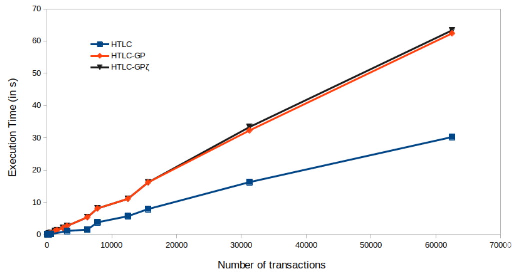


Fig. 6: Comparative analysis of execution time of *HTLC, HTLC-GP*, and HTLC-GP$^{\zeta}$

(i) Uncorrupt players are altruistic: All the nodes follows the step of protocol without assesing the risk. Taking an average over all the instances, we plot the dependency of execution time on the number of transaction request processed

in the network in Fig. 6. We observe that the execution time of HTLC-GP$^\zeta$ is around 62s when number of transaction request is 65000. Since execution steps of *HTLC-GP* and HTLC-GP$^\zeta$ are same, both takes the same time for execution for a given path length and rate of griefing-penalty. The execution time of *HTLC* does not exceed 30s when the number of transaction request is 65000. The reason for an increase in execution time of *HTLC-GP* (or HTLC-GP$^\zeta$) is that it becomes difficult after sometime to get a path with sufficient capacity for locking the payment as well as the penalty. Hence a node has to repeatedly search for neighbor that would have sufficient capacity to route payment.

(ii) Uncorrupt players are rational: We varied the belief θ between 0 to 0.8. However, *HTLC* fails to execute when $\theta > 0.025$. None of the nodes in the network chooses to participate for the fear of loss due to griefing attack. This is not the case with *HTLC-GP* (or HTLC-GP$^\zeta$). The protocol executes succesfully even for $\theta \leq 0.7$. The analysis of belief based on which a participant would be willing to forward payment is provied in Section III of Appendix. It is difficult to compare the run-time of the protocols when one of them fails to execute in a rational model. This shows that both *HTLC-GP* and HTLC-GP$^\zeta$ are robust in a rational model. For a given channel, we computed the execution cost [35][6] of setting up the game model and the time taken by a participant to decide whether to forward or not forward a payment. The execution time was of the order of microseconds. So the possibility of an additional cost incurred while setting up the game model between any two parties can be ruled out.

Inference: The scalability analysis shows that the execution time of *HTLC-GP* (or HTLC-GP$^\zeta$) is around twice of the run time of *HTLC* in presence of altruistic players. However, players are rational and they will not follow the protocol blindly. We have assessed and shown in Fig. 5(a) that a griefing attack can prove to be fatal for the network in terms of unutilized coins. In a rational model, *HTLC* fails even when the belief of a player being corrupt is as low as 0.025. On the contrary, players are willing to participate in both *HTLC-GP* or HTLC-GP$^\zeta$ even if the belief of a player being corrupt is as high as 0.7. Adding a penalization mechanism and controlling the maximum allowed path length for payment proves beneficial in curbing the impact of such attacks, safeguarding the interest of honest participants in the network.

XI. Conclusion

In this paper, we perform a strategic analysis of griefing attacks in Lightning Network. We define a two-player game model where one party chooses its strategy based on its' belief of the other player's type. We have analyzed the effectiveness of payment protocol *HTLC-GP* in the same model. We observe that the cost of attack increases with the introduction of the penalty. However, *HTLC-GP* is found to be weakly effective in countering the attack as it is dependent on the rate of griefing penalty. To further increase the cost of the attack, we introduce the concept of guaranteed minimum compensation for the affected parties and control the maximum path length used for routing. We discuss a modified payment protocol HTLC-GP$^\zeta$, and our experimental results show that the former is more effective than *HTLC-GP* in countering the griefing attack. As a part of our future work, we would like to analyze the impact of network congestion on the uncorrupt party's willingness to lock penalty and extend it to a multi-party game model. We would also like to analyze the griefing attack in the *BAR model* [27] since inclusion of *Byzantine* node will lead to a more realistic modeling of the attack in Bitcoin Network. Lastly, we would like to propose an incentive-compatible countermeasure in presence of rational miners. Tsabary et al. [**?**] had proposed *MAD-HTLC* to counter bribery attack in *HTLC* but they have not discussed griefing attack. Our objective would be to combine the best of the both world and propose a stronger protocol.

Acknowledgement

We thank our anonymous reviewers for their valuable feedback. The work was partially supported by the European Research Council (ERC) under the European Union's Horizon 2020 research (grant agreement 771527- BROWSEC), by the Austrian Science Fund (FWF) through the projects PROFET (grant agreement P31621), and the project W1255- N23, by the Austrian Research Promotion Agency (FFG) through the COMET K1 SBA and COMET K1 ABC, by the Vienna Business Agency through the project Vienna Cybersecurity and Privacy Research Center (VISP), by the Austrian Federal Ministry for Digital and Economic Affairs, the National Foundation for Research, Technology and Development and the Christian Doppler Research Association through the Christian Doppler Laboratory Blockchain Technologies for the Internet of Things (CDL-BOT).

References

[1] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," *Decentralized Business Review*, p. 21260, 2008.

[2] R. Vlastelica, "Why bitcoin wont́ displace visa or mastercard soon." https://www.marketwatch.com/story/why-bitcoin-wont-displace-visa-or-mastercard-soon-2017-12-15, December 2017.

[3] 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*, pp. 106–125, Springer, 2016.

[4] L. Gudgeon, P. Moreno-Sanchez, S. Roos, P. McCorry, and A. Gervais, "Sok: Layer-two blockchain protocols," in *International Conference on Financial Cryptography and Data Security*, pp. 201–226, Springer, 2020.

[5] C. Decker and R. Wattenhofer, "A fast and scalable payment network with bitcoin duplex micropayment channels," in *Symposium on Self-Stabilizing Systems*, pp. 3–18, Springer, 2015.

[6] J. Poon and T. Dryja, "The bitcoin lightning network: Scalable off-chain instant payments," 2016.

[7] C. Egger, P. Moreno-Sanchez, and M. Maffei, "Atomic multi-channel updates with constant collateral in bitcoin-compatible payment-channel networks," in *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, pp. 801–815, 2019.

[8] D. Robinson, "Htlcs considered harmful," in *Stanford Blockchain Conference*, 2019.

[6] As discussed in [35], the cost incurred in a game model can be in terms of execution time or space or locking of extra coins in this context. We show that the execution time is negligible. However, an intermediate party has to lock additional coins in the form of penalty in *HTLC-GP*. Opportunity coins of keeping additional coins unutilized have been taken into consideration while discussing the payoff structure of $\Gamma_{HTLC-GP}$.

[9] Z. Lu, R. Han, and J. Yu, “Bank run Payment Channel Networks,” *IACR Cryptol. ePrint Arch.*, vol. 2020, p. 456, 2020.

[10] Z. Lu, R. Han, and J. Yu, “General congestion attack on htlc-based payment channel networks,” in *3rd International Conference on Blockchain Economics, Security and Protocols (Tokenomics 2021)*, 2021.

[11] E. Rohrer, J. Malliaris, and F. Tschorsch, “Discharged payment channels: Quantifying the lightning network’s resilience to topology-based attacks,” in *2019 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW)*, pp. 347–356, IEEE, 2019.

[12] S. Mazumdar, P. Banerjee, and S. Ruj, “Time is money: Countering griefing attack in lightning network,” in *2020 IEEE 19th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)* [13], pp. 1036–1043.

[13] P. Banerjee, S. Mazumdar, and S. Ruj, “(Full version of Time is Money: Countering Griefing Attack in Lightning Network) Griefing-Penalty: Countermeasure for Griefing Attack in Bitcoin-compatible PCNs,” *CoRR*, vol. abs/2005.09327, 2020.

[14] G. Malavolta, P. Moreno-Sanchez, C. Schneidewind, A. Kate, and M. Maffei, “Anonymous Multi-Hop Locks for Blockchain Scalability and Interoperability,” in *26th Annual Network and Distributed System Security Symposium, NDSS 2019*, 2019.

[15] P. Rogaway and T. Shrimpton, “Cryptographic hash-function basics: Definitions, implications, and separations for preimage resistance, second-preimage resistance, and collision resistance,” in *International workshop on fast software encryption*, pp. 371–388, Springer, 2004.

[16] R. S. Gibbons, “Dynamic games of complete information,” in *Game Theory for Applied Economists*, pp. 55–142, Princeton University Press, 1992.

[17] D. Fudenberg and J. Tirole, “Perfect bayesian equilibrium and sequential equilibrium,” *journal of Economic Theory*, vol. 53, no. 2, pp. 236–260, 1991.

[18] P. Zappalà, M. Belotti, M. Potop-Butucaru, and S. Secci, “Game theoretical framework for analyzing blockchains robustness,” in *Proceedings of the 4th International Symposium on Distributed Computing, Leibniz International Proceedings in Informatics (LIPIcs), Freiburg (virtual conference), Germany*, pp. 49:1–49:3, 2020.

[19] S. Rain, Z. Avarikioti, L. Kovács, and M. Maffei, “Towards a Game-Theoretic Security Analysis of Off-Chain Protocols,” in *36th IEEE Computer Security Foundations Symposium (CSF) (pp. nn-nn). IEEE Computer Society, Washington, DC, USA*, 2023.

[20] J. Xu, D. Ackerer, and A. Dubovitskaya, “A game-theoretic analysis of cross-chain atomic swaps with htlcs,” in *2021 IEEE 41st International Conference on Distributed Computing Systems (ICDCS)*, pp. 584–594, IEEE, 2021.

[21] R. Han, H. Lin, and J. Yu, “On the optionality and fairness of atomic swaps,” in *Proceedings of the 1st ACM Conference on Advances in Financial Technologies*, pp. 62–75, 2019.

[22] “A proposal for up-front payments.” https://lists.linuxfoundation.org/pipermail/lightning-dev/2019-November/002282.html, November 2019.

[23] “Proof-of-closure as griefing attack mitigation.” https://lists.linuxfoundation.org/pipermail/lightning-dev/2020-April/002608.html, April 2020.

[24] G. Danezis and I. Goldberg, “Sphinx: A compact and provably secure mix format,” in *2009 30th IEEE Symposium on Security and Privacy*, pp. 269–282, IEEE, 2009.

[25] S. Azouvi and A. Hicks, “SoK: Tools for Game Theoretic Models of Security for Cryptocurrencies,” *Cryptoeconomic Systems*, vol. 0, apr 5 2021. https://cryptoeconomicsystems.pubpub.org/pub/azouvi-sok-security.

[26] J. Garay, J. Katz, U. Maurer, B. Tackmann, and V. Zikas, “Rational protocol design: Cryptography against incentive-driven adversaries,” in *2013 IEEE 54th Annual Symposium on Foundations of Computer Science*, pp. 648–657, IEEE, 2013.

[27] A. S. Aiyer, L. Alvisi, A. Clement, M. Dahlin, J.-P. Martin, and C. Porth, “Bar fault tolerance for cooperative services,” in *Proceedings of the twentieth ACM symposium on Operating systems principles*, pp. 45–58, 2005.

[28] J. M. Buchanan, “Opportunity cost,” in *The world of economics*, pp. 520–525, Springer, 1991.

[29] Y. Narahari, *Game theory and mechanism design*, vol. 4. World Scientific, 2014.

[30] BtcDrak, M. Friedenbach, and E. Lombrozo, “Bip 112, checksequenceverify.” https://github.com/bitcoin/bips/blob/master/bip-0112.mediawiki, 2015-08-10.

[31] A. Mizrahi and A. Zohar, “Congestion attacks in payment channel networks,” in *International Conference on Financial Cryptography and Data Security*, pp. 170–188, Springer, 2021.

[32] F. Béres, I. A. Seres, and A. A. Benczúr, “(v1) A Cryptoeconomic Traffic Analysis of Bitcoin’s Lightning Network,” *Cryptoeconomic Systems*, jun 22 2020. https://cryptoeconomicsystems.pubpub.org/pub/b8rb0ywn.

[33] D. Goldschlag, M. Reed, and P. Syverson, “Onion routing,” *Communications of the ACM*, vol. 42, no. 2, pp. 39–41, 1999.

[34] A. Hagberg, P. Swart, and D. S Chult, “Exploring network structure, dynamics, and function using networkx,” tech. rep., Los Alamos National Lab.(LANL), Los Alamos, NM (United States), 2008.

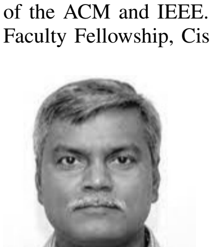
[35] J. Y. Halpern and R. Pass, “Algorithmic rationality: Game theory with costly computation,” *Journal of Economic Theory*, vol. 156, pp. 246–268, 2015.

Subhra Mazumdar completed both her Ph.D. and M.Tech in Computer Science from Indian Statistical Institute Kolkata in 2022 and 2018 respectively. She is currently working as a Project Assistant in Security and Privacy Group at TU Wien and Christian Doppler Laboratory, Blockchain Technologies for the Internet of Things, Vienna Austria.

Prabal Banerjee joined Indian Statistical Institute as a Ph.D. student in 2016. He received his B.Sc. and M.Sc. in Computer Science from St. Xavier’s College, Kolkata and Chennai Mathematical Institute, Chennai respectively. He is currently working on scaling Ethereum as a Researcher at Polygon (Matic Network).

Abhinandan Sinha is an Assistant Professor at Ahmedabad University, India. He has a Bachelor’s in Science with Honors in Statistics from Calcutta University, Masters in Statistics and PhD in Quantitative Economics from Indian Statistical Institute, Kolkata, India. He was previously a researcher at Le Centre national de la recherche scientifique (CNRS) India, on Multidimensional Poverty Indices. His research interest lies in the political economy of development, with mathematical modelling of game theoretical applications. He also devotes time for research and policy-based startups and writes occasional columns on relevant issues.

Sushmita Ruj is a Senior Lecturer at the School of Computer Science and Engineering, University of New South Wales, Sydney, Australia. Her research interests are in Blockchains, Applied Cryptography, Data Privacy. She received her B.E. degree in Computer Science from IIEST, Shibpur, India, and Masters and PhD in Computer Science from Indian Statistical Institute. She was previously a Senior Research Scientist at CSIRO’s Data61 and an Associate Professor at Indian Statistical institute. She serves as a reviewer of Mathematical Reviews, Associate editor of Elsevier Journal, Information Security and Applications, Elsevier journal on Pervasive and Mobile Computing. She is a senior member of the ACM and IEEE. She is a recipient of Samsung GRO award, NetApp Faculty Fellowship, Cisco Academic Grant and IBM OCSP grant.



Bimal Kumar Roy has received the Ph.D. degree in combinatorics from the University of Waterloo, Waterloo, Canada, in 1982. He is currently a Professor at Applied Statistics Unit and the former Head of R C Bose Centre for Cryptology and Security, Indian Statistical Institute, Kolkata, India. He is also the Founder and General-Secretary, Cryptology Research Society of India. He served as the Director of Indian Statistical Institute, Kolkata, from 2010–2015. Over the past 40 years, he has published many research papers on the subject of Cryptography. His primary research interests include Cryptology, Data obfuscation, Design of secure Electronic voting machine, Sensor Networks, Combinatorics, Design of Experiments, and Optimization.