Counter-Collusion Smart Contracts for Watchtowers in Payment Channel Networks

Yuhui Zhang Dejun Yang Guoliang Xue Ruozhou Yu

***Abstract*—Payment channel networks (PCNs) are proposed to improve the cryptocurrency scalability by settling off-chain transactions. However, PCN introduces an undesirable assumption that a channel participant must stay online and be synchronized with the blockchain to defend against frauds. To alleviate this issue, watchtowers have been introduced, such that a hiring party can employ a watchtower to monitor the channel for fraud. However, a watchtower might profit from colluding with a cheating counterparty and fail to perform this job. Existing solutions either focus on heavy cryptographic techniques or require a large collateral. In this work, we leverage smart contracts through economic approaches to counter collusions for watchtowers in PCNs. This brings distrust between the watchtower and the counterparty, so that rational parties do not collude or cheat. We provide detailed analyses on the contracts and rigorously prove that the contracts are effective to counter collusions with minimal on-chain operations. In particular, a watchtower only needs to lock a small collateral, which incentivizes participation of watchtowers and users. We also provide an implementation of the contracts in Solidity and execute them on Ethereum to demonstrate the scalability and efficiency of the contracts.**

I. Introduction

The past decade has seen a blooming of cryptocurrencies [25], e.g., Bitcoin [20] and Ethereum [15]. However, cryptocurrencies cannot scale for wide-spread use, due to high overhead and storage requirement [12]. For example, Bitcoin can only process up to 7 transactions per second (tps) [11], compared to over 47,000 peak tps handled by Visa [26].

Payment channel networks (PCNs), e.g., Bitcoin's Lightning Network [22] and Ethereum's Raiden Network [24], have been proposed to tackle the scalability issues [22]. PCNs can process instant and less valuable payments without involving slow and expensive blockchain transactions [27–29]. However, an essential assumption in PCN is that channel funds can be secure only if the channel participant stays online. A channel participant risks losing payments if it goes offline, since the other channel participant on this channel can request to close the channel by publishing an outdated fraud channel state proof (CSP). Meanwhile, opening and closing channels are expensive transactions on the blockchain. Thus, the channel participants want to avoid performing these operations frequently. Watchtowers have long been considered crucial for squashing fraud in PCNs [6]. The concept of watchtower was first proposed in the Lightning whitepaper [6]. Recently, Bitcoin Wallet Electrum has released its watchtower implementation [2]. Lightning Labs [1] and Blockstream [3] also announced their plans on the implementations in the Lightning Network Daemon (lnd) [9] and c-lightning [4], which are the main Lightning clients.

Zhang and Yang are affiliated with Colorado School of Mines, Golden, CO 80401. Xue is affiliated with Arizona State University, Tempe, AZ 85287. Yu is affiliated with North Carolina State University, Raleigh, NC 27695. Email:{yuhzhang, djyang}@mines.edu, xue@asu.edu, ryu5@ncsu.edu
This research was supported in part by NSF grants 1717197, 1717315, 2007083, 2007391, and 2008935. The information reported here does not reflect the position or the policy of the federal government.

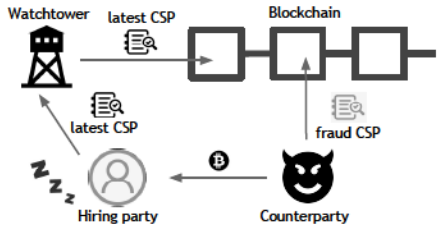

Fig. 1. Illustration of a watchtower. The watchtower monitors the channel for the offline hiring party. When the counterparty requests to close the channel with a fraud CSP, the watchtowers responds with the latest CSP.

Watchtowers [10, 17, 19, 21] are monitoring services that are always online and able to monitor the blockchain as shown in Fig. 1. A hiring party (*i.e.,* a channel participant) can hire a watchtower and go offline. The watchtower can be paid either for each detected fraud or simply for monitoring. Since the PCN protocol punishes detected frauds, channel participants rarely cheat. Hence, it is more reasonable to pay a watchtower based on its actual cost of watching, which is proportional to the amount of stored CSP data. On the other hand, a watchtower needs to prove to the hiring party that it has been indeed performing the channel monitoring job. In our construction, the hiring party sends the latest CSP to the watchtower after each transaction. The watchtower, in return, needs to provide the hiring party with a proof that it has been monitoring the blockchain and stored this CSP. This proof-scheme makes a payer-transaction scheme palatable to channel participants. However, a rational watchtower might profit from colluding with a counterparty (*i.e.*, the other channel participant on this monitored channel). For example, the counterparty can bribe the watchtower and request to close the channel by publishing a fraud CSP. As a result, the rational watchtower would fail to monitor the channel and not respond with the latest CSP.

In this paper, we tackle this collusion problem by addressing its root cause through economic approaches, instead of resisting collusions via heavy cryptographic methods. **The main contributions of this paper are**:

- To the best of our knowledge, we are the first to study the collusion problem for watchtowers in PCNs from a game theatrical perspective, where a watchtower could profit from coordinating with the counterparty of the monitored channel and not performing the monitoring job.

- To tackle this collusion problem, we design three smart contracts, Watchtower contract, Collusion contract, and Betrayal contract, for scenarios where a hiring party outsources a channel monitoring job to a watchtower.
- We provide detailed analyses of the smart contracts. Specifically, we rigorously prove that there is a unique sequential equilibrium, *i.e.*, the rational parties will never collude or cheat with the existence of all the three contracts, even if they are allowed. Thus, the rational parties will never execute the Collusion or Betrayal contracts and are involved in minimal on-chain operations. In addition, the Watchtower contract is compilable with the PCN protocol and does not require on-chain operations.
- We provide a proof-of-concept implementation of the smart contracts in Solidity and execute them on Ethereum. It demonstrates that even if the contracts happen to be executed, the financial cost is very low.

The remainder of the paper is organized as follows. In Section II, we provide a brief literature review of the related work. In Section III, we present the background in PCNs and relevant concepts in game theory. In Section IV, we formally describe the adversary model and give the necessary assumptions. In Section V, we design the Watchtower contract and conduct detailed game theoretical analysis. In Section VI, we design the Collusion contract and conduct detailed game theoretical analysis. In Sections VII, we design the Betrayal contract and conduct detailed game theoretical analysis. In Sections VIII, we conduct detailed game theoretical analysis of the full game induced by all the three contracts. In Section IX, we implement the smart contracts in Solidity and execute them on Ethereum. In Section X, we conclude this paper.

II. Related Work

The hazard of execution fork attacks against offline parties has raised a lot of attentions in the off-chain scaling community [1–3, 6], who have proposed several mitigations thus far. Existing approaches enable the hiring parties (i.e., channel participants) to employ watchtowers [23] to help defend against execution forks on their behalf. Dryja *et al.* proposed Monitor [14], which requires $O(N)$ storage, where N is the number of off-chain transactions that have occurred within the channel. Osuntokunn *et al.* designed Watchtower [21], which improves Monitor's efficiency, but cannot be deployed without consensus rule changes in the Bitcoin network. Both proposals suffer from the drawback that if the hired watchtower fails, there is little recourse for the hiring party, since the protocols do not provide evidence about the employment.

Pisa [19] and Outpost [17] provide the hiring parties with publicly verifiable cryptographic evidence in case the watchtower fails, which can be used to penalize the watchtower. However, penalty does not guarantee honest behaviors, i.e., responding upon fraud is not the watchtower's dominant strategy in these approaches. Cerberus [10] presented a watchtower extension of the Bitcon Lightning [7] channels, which needs to lock a large collateral more than the amount of the channel funds to resist against bribing. In this work, we address the collusion problem through economic approaches to avoid heavy cryptographic solutions and guarantee security against collusion with a small reasonable amount of deposit from watchtowers.

III. Background and Preliminaries

In this section, we provide the necessary background on permissionless, introduce the background in payment channel networks, and present relevant concepts in game theory.

A. Decentralized Ledger

Cryptocurrencies, e.g., Bitcoin [20] and Ethereum [15], are based on the blockchain technology, which is an append-only decentralized ledger of transactions shared among mutually distrusted entities. However, the consensus algorithm requires large local storage due to the high levels of data replication and high computational power for adding a block to the chain. Thus, cryptocurrencies cannot scale for wide-spread use.

B. Payment Channel Network (PCN)

To overcome the scalability issue, payment channels networks (PCNs) have been proposed to eliminate the need to commit each transaction on the blockchain [23]. A payment channel is protected by multi-signature smart contracts, which ensure validity, nonequivocality and non-repudiation. Two parties open a payment channel by each depositing a certain amount into a joint account and adding this opening transaction ($topen$) to the blockchain. Once the channel is opened, both parties exchange signed commitment transactions (ctxs) between each other. Now each signed ctx between them is essentially a distribution of $topen$'s funds agreed upon by both parties. When the channel closes, a closing transaction will be broadcast to the blockchain and will send funds to each party according to the latest ctx (referred to as $latest\_ctx$).

If one party goes offline permanently, the counterparty can sign and broadcast $latest\_ctx$ to the blockchain to close the channel unilaterally. This brings distrust between the two parties. Because a dishonest party can broadcast a more favorable outdated fraud ctx (referred to as $previous\_ctx$), since each party could store a stream of $previous_ctx$s backtracking to the channel opening. PCNs resolve this potential cheating by allowing $latest\_ctx$ to be exchanged with a revocation key that can allocate the entire channel funds to the victim party. If one party requests to close the channel unilaterally, it has to wait for a timelock, which gives the other party a time window to detect a fraud $previous\_ctx$ and raise a dispute on the blockchain. The dispute will be settled by the miners on the blockchain. Thus, an essential assumption is that a party has to be online at least once every timeblock to construct the corresponding jtx and broadcast it.

C. Watchtower

Watchtowers are service providers that are always online and able to monitor the blockchain. To go offline, a hiring party (*i.e.*, a channel participant) outsources the channel monitoring job to a watchtower and gives the latest *channel state proof* (CSP), a $[ctx\_txid\_prefix, ejtx]$ pair, to the watchtower. The

watchtower is expected to respond with the latest CSP, when the counterparty (*i.e.*, the other channel participant on the monitored channel) requests to close the channel unilaterally by publishing a fraud CSP (*i.e.*, an outdated $previous\_ctx$).

The hiring party gives the latest CSP to the watchtower, when a new channel update is agreed upon. The watchtower stores a CSP map, where the keys are ctx_txid_prefixs and the values are $ejtx$s. Then, it watches the blockchain for any transaction whose $txid\_prefix$ matches any of the keys in this map. If the watchtower finds a match, it extracts the $txid\_suffix$ from the blockchain $txid$, and uses this ctx_txid to decrypt the corresponding $ejtx$ from its map to get the raw jtx, which is already signed by the corresponding hiring party of that channel. The watchtower then broadcasts this jtx to penalize the corresponding counterparty.

D. Smart Contract

Smart contracts are machinery that can be enforced on the blockchain [13]. In other words, a smart contract is a piece of program stored and executed on the blockchain. Smart contracts capture the logic of contractual clauses between parties and are executed when certain events are triggers. In addition, a smart contract can maintain funds and store the code that decides the flow of the funds in the contract account. Smart contracts are executed by the consensus peers, and the correctness of execution is guaranteed by the consensus protocol. Ideally, smart contracts can be considered to be executed by a global machine that will faithfully execute every instruction.

E. Games and Strategies

In this paper, we design games in extensive form with imperfect information. Imperfect information means that the players has partial or no knowledge of the actions taken by others; while perfect information has a strong assumption that players knows every action of others. Blockchain is anonymous and global, and one can maintain several accounts to perform the actions. Therefore, we model the games as imperfect-information, which are more realistic and allows a wider scope of analysis. Informally speaking, an imperfect-information game in extensive form is a tree in terms of graph theory, where each node represents one player's choice, each edge represents a possible action, and the leaves represent final outcomes over which each player has a utility function. Formally, a *finite imperfect-information game* is defined as follows [18].

Definition 1 (Finite imperfect-information game)**.** *A finite imperfect-information game (in extensive form) is a tuple* $\mathcal{G} = (\mathcal{N}, \mathcal{A}, \mathcal{V}, \mathcal{T}, \chi, \rho, \sigma, u, \mathcal{I})$*, where:*

- $\mathcal{N}$ *is a set of* n *players.*
- $\mathcal{A}$ *is a single set of actions.*
- $\mathcal{V}$ *is a set of nonterminal choice nodes.*
- $\mathcal{T}$ *is a set of terminal nodes, disjoint from* $\mathcal{V}$*.*
- χ*:* $\mathcal{V} \to 2^{\mathcal{A}}$ *is the action function, which assigns to each choice node a set of possible actions.*
- ρ*:* $\mathcal{V} \to \mathcal{N}$ *is the player function, which assigns to each nonterminal node a player* $i \in \mathcal{N}$ *who chooses an action at that node.*

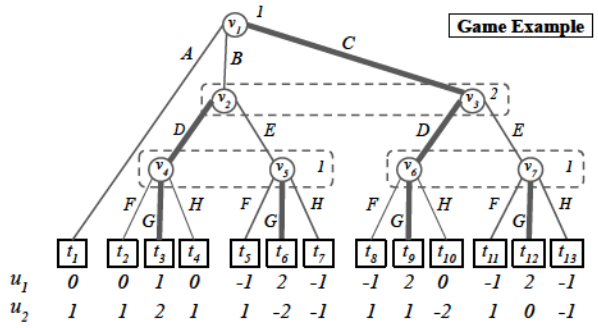

Fig. 2. Example of game tree. Bold edges indicate the actions in the unique sequential equilibrium.

- σ*:* $\mathcal{V} \times \mathcal{A} \to \mathcal{V} \cup \mathcal{T}$ *is the successor function, which maps a choice node and an action to a new choice node or terminal node such that* $\forall v_i, v_j \in \mathcal{V}$ *and* $a_i, a_j \in \mathcal{A}$*, if* $\sigma(v_i, a_i) = \sigma(v_j, a_j)$ *then* $v_i = v_j$ *and* $a_i = a_j$*.*
- $u = (u_1, \ldots, u_n)$ *is a vector of utility functions, where* $u_i : \mathcal{T} \to \mathbb{R}$ *is player* i*'s utility function on* $\mathcal{T}$*.*
- $\mathcal{I} = (\mathcal{I}_1, \ldots, \mathcal{I}_n)$*, where* $\mathcal{I}_i = (\mathcal{I}_{i,1}, \ldots, \mathcal{I}_{i,k_i})$ *is an equivalence relation on* $\{v \in \mathcal{V} : \rho(v) = i\}$ *(i.e., a partition of* i*'s choice nodes) subject to* $\chi(v) = \chi(v')$ *and* $\rho(v) = \rho(v')$*, whenever there is a* j *such that* $v, v' \in I_{i,j}$*. Intuitively, if two choice nodes are in the same information set* $\mathcal{I}_{i,j}$*, player* i *cannot differentiate them.*

A *strategy* of a player determines the action that the player will take in the game. In an imperfect-information game, the player i cannot differentiate two choice nodes v and v' in the same information set $\mathcal{I}_{i,j} \in \mathcal{I}_i$. In other words, the set of possible actions assigned to nodes in the same information set is the same, *i.e.*, $\forall v, v' \in \mathcal{I}_{i,j}$, $\chi(v) = \chi(v')$. Therefore, player i selects one of the available actions at each $\mathcal{I}_{i,j}$. Formally, a strategy in an imperfect-information game is defined as follows.

Definition 2 (Strategy)**.** *Let* $\mathcal{G} = (\mathcal{N}, \mathcal{A}, \mathcal{V}, \mathcal{T}, \chi, \rho, \sigma, u, \mathcal{I})$ *be an imperfect-information extensive-form game. A strategy of a player* i *is a tuple of actions* $s_i = (a_{i,1}, \ldots, a_{i,k_i})$*, where* $a_{i,j} \in \chi(v), v \in \mathcal{I}_{i,j} \in \mathcal{I}_i$ *and* $\forall v, v' \in \mathcal{I}_{i,j}, \chi(v) = \chi(v')$*.*

Definition 3 (Strategy profile)**.** *A strategy profile of a game is a tuple of all players' strategies* $s = (s_1, \ldots, s_n)$*, where* s_i *is one of the player* i*'s strategies.*

Definition 4 (Sequential equilibrium)**.** *A strategy profile* s *is a sequential equilibrium of an extensive-form game* $\mathcal{G}$*, if there exist probability distributions* $\mu(\mathcal{I}_{i,j})$*,* $\forall \mathcal{I}_{i,j} \in \mathcal{I}_i$ *such that:*

1) $(s, \mu) = \lim_{n \to \infty} (s_n, \mu_n)$ *for some sequence* $(s_1, \mu_1), (s_2, \mu_2), \ldots$*, where* s_n *is fully mixed, and* μ_n *is consistent with* s_n*.*
2) *For any information set* $\mathcal{I}_{i,j} \in \mathcal{I}_i$*, and any strategy* $s_i' \neq s_i$*, we have* $u_i(s|\mathcal{I}_{i,j}, \mu(\mathcal{I}_{i,j})) \geq u_i((s', s_{-i})|\mathcal{I}_{i,j}, \mu(\mathcal{I}_{i,j}))$*.*

An example of game tree is shown in Figure 2. In this game, there are 2 players $\mathcal{N} = \{1, 2\}$ and a set of actions $\mathcal{A} = \{A, \ldots, H\}$. The circle nodes denote the nonterminal choice nodes $\mathcal{V}$. The rectangle nodes denote the terminal choice nodes $\mathcal{T}$. The utilities (payoffs) of players 1 and 2 are u_1 and

u_2 at the bottom of the terminal nodes. An information set is denoted by the nodes within the dashed rectangles. There are 4 information sets: $\mathcal{I}_{1,1} = \{v_1\}, \mathcal{I}_{2,1} = \{v_2, v_3\}, \mathcal{I}_{2,1} = \{v_4, v_5\}$, and $\mathcal{I}_{2,2} = \{v_6, v_7\}$. The sequential equilibrium is a strategy profile $s = (s_1, s_2)$, where $s_1 = (C, G, G)$ and $s_2 = (D)$. The strategy s_1 indicates that player 1 plays C at $\mathcal{I}_{1,1}$, plays D at $\mathcal{I}_{1,2}$, and plays D at $\mathcal{I}_{1,3}$.

F. Monetary Variables

Below are the monetary variables that will be used in the contracts (listed in alphabetic order). They are all non-negative.

- b: the bribe that the counterparty agrees to pay the watchtower in the collusion agreement (Collusion contract).
- c: the watchtower's cost for monitoring the channel.
- f: the verification fee paid to the miners for settling a dispute on the blockchain.
- d_w: the amount that a watchtower agrees to deposit to get the channel monitoring job.
- d_c: the fund that the counterparty owns on the channel.
- d_t : the amount that both the counterparty and the watchtower agree to deposit in the collusion agreement (Collusion contract).
- w_h: the amount that the hiring party agrees to pay to a watchtower for monitoring the payment channel.
- w_c: the extra amount that the counterparty can earn by publishing a fraud channel state proof (CSP).

The following relations hold for obvious reasons:

- $w_h > c$: the watchtower does not accept under-paid jobs.
- $w_c > b$: the counterparty does not pay more bribe than its gain from the collusion.

The monetary variables w_h and d_w can be set by the hiring party in the Watchtower contract; the monetary variables b and d_t can be set by the counterparty in the Collusion contract.

IV. System Model and Assumptions

A. Cryptographic Assumptions

We make the typical cryptographic assumptions. We assume that there are secure communication channels, cryptographically secure hash functions, signatures, and encryption schemes. In addition, all parties (watchtowers, channel participants, external adversaries, etc) are computationally bounded.

B. Blockchain Assumptions

We assume a perfect blockchain, where both persistence and liveness hold [16]. Specifically, we assume that if a valid transaction is propagated in the blockchain, it cannot be censored and will be included in the "permanent" part of the blockchain immediately. Additionally, we assume that any channel participant can go offline (intentionally or unintentionally) for a (long) time period of up to T_{off}. Furthermore, we assume that watchtowers are resilient against DoS attacks and always online. This assumption is realistic, since watchtowers are required to deposit to participate in the system and thus will invest in the anti-DoS protection [10].

C. Adversary Model

We consider an honest hiring party who outsources the job of monitoring the payment channel. For the hiring party, the goal is to get its channel monitored while minimizing the cost. The watchtower is unreliable and could respond with a false channel state proof (CSP) for the outsourced monitoring job. Note that in this paper, we do not distinguish intentional and unintentional faults, because it is difficult to collect evidence. We assume the watchtower and the counterparty are physically isolated, because the watchtowers are usually maintained by the leading nodes in PCNs. We also assume each party is an individually rational adversary. Being rational means that a party always acts in a way that maximizes its payoff and is capable of thinking through all possible outcomes and choosing strategies that will result in the best possible outcome.

V. The Watchtower Contract

In this section, we present the Watchtower contract and analyze the game induced by the Watchtower contract.

A. The Contract

The Watchtower contract is an outsourcing contract signed by a hiring party (H) and a watchtower (W). The contract allows H to employ W to monitor the channel. W should watch the channel state and respond with the latest channel state proof (CSP) p timely, if the counterparty (C) requests to close the channel unilaterally. At a high level, it aims to incentivize honest behaviors by asking W to deposit beforehand. If the watchtower behaves honestly, the deposit will be refunded; if the watchtower cheats and is detected, the deposit will be taken by H. The contract is presented below:

1) The contract is between H and W.
2) H agrees to provide the latest CSP p to W and deposit w_h to the contract for W to monitor its channel.
3) W agrees to monitor the channel and deposit d_w to contract.
4) If either H or W fails to sign the contract before the deadline T_1, the contract will terminate, and any w_h and/or d_w held by the contract will be refunded.
5) When C requests to close the channel at T_2:
   a) If W sends p before $T_2 + \delta$, then $w_h + d_w$ is paid to W.
   b) Otherwise, $w_h + d_w$ is paid to H.
6) When C does not request to close the channel by T_3:
   a) If W provides p to H to prove that it has been indeed monitoring the channel, then $w_h + d_w$ is paid to W.
   b) Otherwise, $w_h + d_w$ is paid to H.

The Watchtower functionality that allows W to deposit has recently be implemented as an extension called Cerberus [5] of the Bitcoin Lightning Network [7]. Thus, it does not require on-chain operations that cost transaction and execution fees. Cerberus counters collusions by locking a large deposit from W. In this paper, we provide a contract-based solution, such that only a small deposit is required from W to counter collusions.

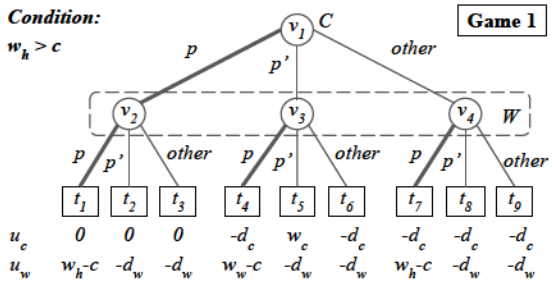

Fig. 3. Game induced by the Watchtower contract

B. The Game and Analysis

The game included by the Watchtower contract is shown in Game 1 in Fig. 3. In this game, the players are $\mathcal{N} = \{C, W\}$. Although the contract involves H, H can be eliminated from the game, because it has only one deterministic strategy, *i.e.*, outsourcing the monitoring job to W. W and C can communicate and send a fraud CSP $p' \neq p$ to close the channel. The action set is $\mathcal{A} = \{p, p', other\}$. The first two actions represent that the player sends p or p' before the deadline. The last means any other actions the player may take. Game 1 has 2 information sets: $I_{c,1} = \{v_1\}$, and $I_{w,1} = \{v_2, v_3, v_4\}$. The game tree captures $\mathcal{V}, \mathcal{T}, \chi, \rho$, and σ. The utilities (payoffs) are listed below the terminal nodes.

Next, we analyze Game 1 and show that if $w_h > c$, both players will always send p and Game 1 will always terminate at t_1. Formally, we have:

Theorem 1. *If $w_h > c$, Game 1 has a unique sequential equilibrium $(s_c^1, s_w^1) = ((p), (p))$. According to the equilibrium, C will send the latest CSP p, and W will respond with p.*

Proof. We prove this by showing that the only reachable outcome of Game 1 is t_1. The intuition is that sending p always leads to the highest payoff for both players. At W's choice node v_2, W's payoff is $w_h - c$, $-d_w$, and $-d_w$, if the game ends at t_1, t_2, and t_3, respectively. Since $w_h - c > -d_w$, W will send p at v_2 to reach t_1. Similarly, W will send p at v_3 and v_4. Hence, W will always send p, no matter what C's action is. Inferring this, C knows that the only reachable nodes are t_1, t_4 and t_7. Since t_1 has a higher payoff than t_4 and t_7 for C, C will always send p. Thus, Game 1 has a unique sequential equilibrium that both players will send p. □

VI. The Collusion Contract

The Watchtower contract works, because there is no trust between the counterparty (C) and the watchtower (W) to collude. However, C and W can make credible and enforceable promises to collude by both signing a collusion contract. In this section, we present the Collusion contract and analyze the game induced by both the Watchtower and Collusion contracts.

A. The Contract

The Collusion contract is a collusion contract signed by C and W. The contract allows C to bribe W for collusion, *i.e.*, W should coordinate with C, when C requests to close this channel unilaterally with a fraud channel state proof (CSP) p'.

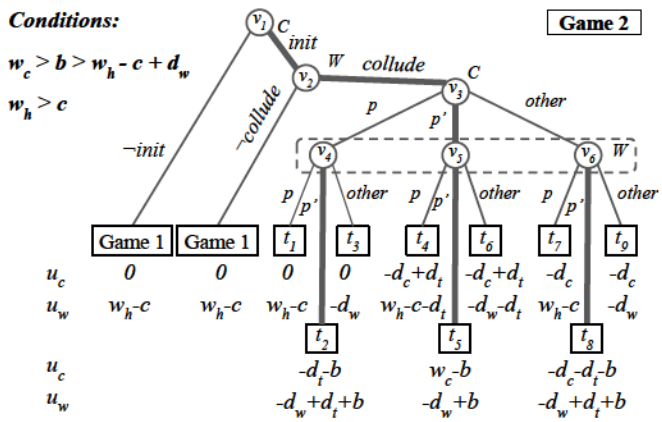

Fig. 4. Game induced by the Watchtower contract and the Collusion contract

At a high level, it aims to incentivize C and W to send a fraud CSP by redistributing the profit between them and punishing those who deviate from the collusion. The trust between C and W is built by asking both C and W to deposit beforehand. Either C or W, who deviates from collusion, will be punished by losing the deposit. The contract is presented below:

1) The contract is between C and W.
2) C agrees to provide p' and deposit $d_t + b$ to the contract.
3) W agrees to collude with C and deposit d_t to the contract.
4) If either C or W fails to sign the contract before the deadline T_2, the contract will terminate, and any b and/or d_t held by the contract will be refunded.
5) When C requests to close the channel by sending p' at T_2:
   a) If W sends p', d_t is refunded to C; $d_t + b$ is paid to W.
   b) Otherwise, $2d_t + b$ is paid to C.
6) When C sends a CSP other than p' at T_2:
   a) If W sends p', $2d_t + b$ is paid to W.
   b) Otherwise, $d_t + b$ is refunded to C; d_t is refunded to W.
7) When C does not request to close the channel by T_3: $d_t + b$ is refunded to C, and d_t is refunded to W.

The Collusion contract must be signed before T_2, so that C and W can trust each other to collude without any risk. C needs to provide p' before T_2. In Clause $5(b)(ii)$, when both C and W deviate from collusion, neither of them is punished.

In this contract, C agrees to pay a bribe b to motivate W to collude. Both C and W lock a deposit d_t to ensure that 1) the deviating player always gets a lower payoff than not deviating from the collusion; 2) the player following the collusion always gets a higher payoff than not following the collusion.

B. The Game and Analysis

The game induced by the Watchtower and Collusion contracts is shown in Game 2 in Fig. 4. In this game, the players are $\mathcal{N} = \{C, W\}$. The action set is $\mathcal{A} = \{\neg init, init, \neg collude, collude, p, p', other\}$. The first two actions represent that C has the option of to initiate the collusion or not. The followed two actions represent that W has the option of to collude with C or not. Game 2 has 4 information sets, *i.e.*, $\mathcal{I}_{c,1} = \{v_1\}$, $\mathcal{I}_{c,2} = \{v_3\}$ (belonging to C), and

$\mathcal{I}_{w,1} = \{v_2\}$, $\mathcal{I}_{w,2} = \{v_4, v_5, v_6\}$ (belonging to W). The payoffs (utilities) are listed below the terminal nodes.

If C plays $\neg init$ or W plays $\neg collude$, they will end up with playing Game 1 (in Fig. 3), because the only activated contract is the Watchtower contract. We will not show the analysis of these two branches here, as it is exactly the same as in Section V-B (subject to relabeling of nodes). Because only one terminal node t_1 is reachable in Game 1, we can replace each branch with a single terminal node $Game\ 1$, and it has the same payoffs as t_1 in Game 1. If the Collusion contract in initiated and signed, the payoffs are decided jointly by both the Watchtower and Collusion contracts.

Next, we analyze Game 2 and show that if $w_c > b > w_h - c + d_w$ and $w_h > c$, both players will collude and send p' and Game 2 will always terminate at t_5. Formally, we have:

Theorem 2. *If $w_c > b > w_h - c + d_w$ and $w_h > c$, Game 2 has a unique sequential equilibrium (s_c^2, s_w^2):*

$$\begin{cases} s_c^2 = & (init, s_c^1, s_c^1, p'), \\ s_w^2 = & (s_w^1, collude, s_w^1, p'). \end{cases}$$

According to the equilibrium, C will initiate the collusion, W will agree to collude, C will send the fraud CSP p', and W will respond with p'.

Proof. We prove this by showing that the only reachable outcome of Game 2 is t_5. The intuition is that colluding and sending p' always leads to the highest payoff for both players. At W's choice node v_4, the payoff of W is $w_h - c$, $-d_w + d_t + b$, and $-d_w$, if the game ends at t_1, t_2, and t_3, respectively. Since $b > w_h - c + d_w$, W will play p' at v_4 to reach t_2. Similarly, W will play p' at v_5 and v_6. Hence, W will always play p', no matter what C's action is. Inferring this, C knows that the only reachable nodes are t_2, t_5 and t_8. Since t_5 has the highest payoff for C, C will always play p' to reach t_5. Inferring this, W will always $collude$ at v_2, because t_5 has a higher payoff than $Game\ 1$ for W. Inferring this, C will always $init$ the collusion at v_1. Thus, Game 2 has a unique sequential equilibrium that both players will collude and send p'. □

VII. The Betrayal Contract

The Collusion contract works, because it brings trust between the counterparty (C) and the watchtower (W). However, the hiring party (H) can create distrust between C and W by incentivizing W to betray the collusion and behave honestly. In this section, we present the Betrayal contract and analyze the sub-game induced by the Watchtower and Betrayal contracts.

A. Challenge

The main challenge to address the collusion problem is to get out of the counter/counter-back loop. H could always counter the Collusion contract by providing a higher reward w_h to W, which makes the collusion less profitable and changes the equilibrium. However, C could always counter-back by providing a higher bribe b, so that the collusion becomes more profitable again. This counter/counter-back loop could fall into the endless competition between C and H until one runs out of funds. The Betrayal contract focuses on the root cause and tackles the collusion problem by bringing distrust between C and W. Specifically, H offers W the total penalty from C, if W reports the collusion and respond correctly. The goal of the Betrayal contract is to incentivize W to report the collusion, but not necessarily to betray the collusion.

B. The Contract

The Betrayal contract is a report contract signed by H and W. The contract allows W to report collusion and respond upon fraud. W could report the collusion to H and respond with a channel state proof (CSP) p^*, when C tries to bribes W to collude and close this channel unilaterally. At a high level, it tries to incentivize W to report the collusion by paying C's deposit d_c to W. The contract is presented below:

1) The contract is signed between H and W, both of whom have signed the Watchtower contract.
2) H agrees to deposit d_c for W to report the collusion.
3) W agrees to deposit f and report the collusion.
4) If either H or W fails to sign the contract before T_2, the contract will terminate, and any d_c or/and f held by the contract will be refunded.
5) If W reports a collusion before T_2:
   a) If C sends a fraud CSP at T_2,
      i) If W responds with $p^* = p$, $d_c + f$ is paid to W.
      ii) Otherwise, d_c is paid to H.
   b) Otherwise, f is paid to H.
6) If W does not report a collusion before T_2, then d_c is refunded to H, and f is refunded to W.

Note that W does not have to sign the Collusion contract to report a collusion. W can misreport a collusion, unintentionally or intentionally. W can respond with $p*$ anonymously from another address maintained by W, when C requests to close the channel. By providing the evidence to H in the Betrayal contract, W can be prove that it has performed the channel monitoring job.

C. The Sub-game and Analysis

Before analyzing the full game induced by the Watchtower, Collusion, and Betrayal contracts, we first present and analyze a sub-game induced by the Watchtower and Betrayal contracts. The sub-game illustrates the situation when the Collusion contract is not activated, because either the collusion coalition is not initiated by C or is rejected by W.

The sub-game included by the Watchtower and Betrayal contracts is shown in Game 3 in Fig. 5. In this game, the players are $\mathcal{N} = \{C, W\}$. The action set is $\mathcal{A} = \{\neg report, report\ p* = p, report\ p* \neq p, p, p', other\}$. The first action means that the player has the option of not to report the collusion. The followed two actions represent that the player can report the collusion and send p^* before the deadline. Game 3 has 4 information sets, *i.e.*, $\mathcal{I}_{c,1} = \{v_2, v_3\}$ (belonging to C), $\mathcal{I}_{w,1} = \{v_1\}$, $\mathcal{I}_{w,2} = \{v_4, v_5, v_6\}$, and $\mathcal{I}_{w,3} = \{v_7, v_8, v_9\}$ (belonging to W). The payoffs (utilities) are listed below the terminal nodes.

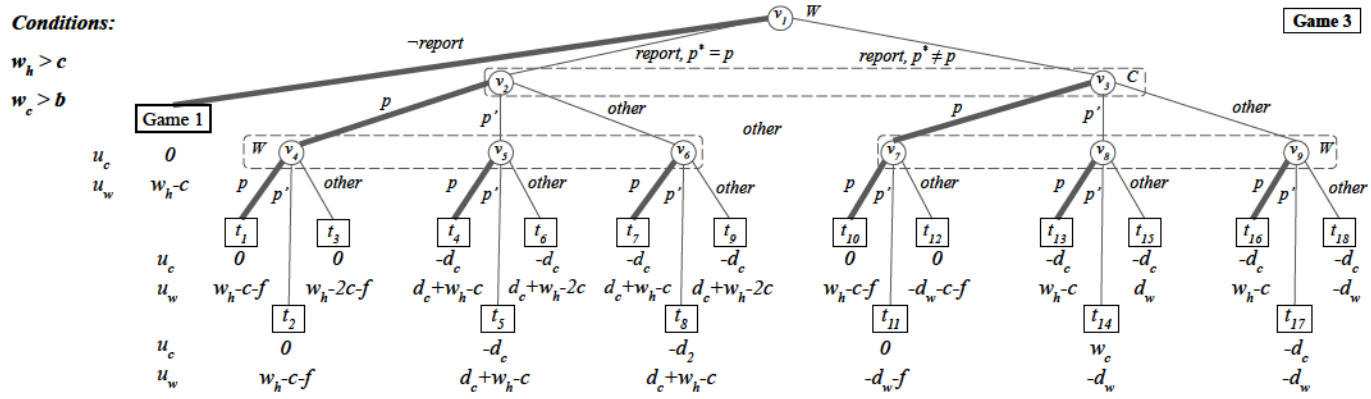

Fig. 5. Sub-game induced by the Watchtower contract and the Betrayal contract

If W chooses not to report the collusion, the only activated contract is the Watchtower contract and the branch is exactly the same as the game tree of Game 1. We will not show the analysis of this branch here, as it is exactly the same as in Section V-B (subject to relabeling of nodes). Because only one terminal node t_1 is reachable in Game 1, we can replace this branch with a single terminal node $Game\ 1$, and it has the same payoffs as t_1 in Game 1. If W decides to report, W can report the collusion and then send either $p^* = p$ (branch to v_2) or $p^* \neq p$ (branch to v_3). In both cases, the payoffs are decided jointly by the Watchtower and Betrayal contracts.

Next, we analyze Game 3 and show that if $w_h > c$ and $w_c > b$, W will not report collusion and both players will send p. Game 3 will terminate at t_1 in Game 1. Formally, we have:

Theorem 3. *If $w_h > c$ and $w_c > b$, then Game 3 has a unique sequential equilibrium (s_c^3, s_w^3), where*

$$\begin{cases} s_c^3 = & (p), \\ s_w^3 = & (\neg report, p, p, p). \end{cases}$$

According to this equilibrium, W will not report a collusion, C will send the latest CSP p, and Wwill respond with p.

Proof. We prove this by showing that the only reachable outcome of Game 3 is t_1 in Game 1. The intuition is that not reporting collusion and sending p always leads to the highest payoff for each player. At W's choice node v_7, the payoff of W is $w_h - c - f$, $-d_w - f$, and $-d_w - c - f$, if the game ends at t_{11}, t_{12}, and t_{13}, respectively. Since $w_h - c - f > -d_w - f$, W will play p at v_7 to reach t_{10}. Similarly, W will play p at v_8 and v_9. Hence, W will always play p, no matter what C's action is. Inferring this, C knows that the only reachable nodes are t_{10}, t_{13} and t_{16}. Since t_{10} has the highest payoff for C, C will play p at v_3 to reach t_{10}. Similarly, W will always play p at v_4, v_5, and v_6, no matter what C's action is; C will always play p at v_2 to reach t_1. Inferring this, W will always $\neg report$ at v_1, because $Game\ 1$ has a higher payoff than t_1 and t_{10} for W. Thus, Game 3 has a unique sequential equilibrium that W will not report collusion and both players will send p'. □

VIII. The Full Game Induced by All the Contracts

In this section, we present and analyze the full game induced by the Watchtower, Collusion contract, and Betrayal contracts.

A. The Game and Analysis

The full game included by the Watchtower, Collusion, and Betrayal contracts is shown in Game 4 in Fig. 6. In this game, the players are $\mathcal{N} = \{C, W\}$. The action set is $\mathcal{A} = \{init, \neg init, collude, \neg collude, \neg report, report\ p* = p, report\ p* \neq p, p, p', other\}$. The first two action represent that the player has the option of to initiate the collusion or not. The followed two actions represent that the player has the option of to collude or not. Game 4 has 7 information sets, *i.e.*, $\mathcal{I}_{c,1} = \{v_1\}$, $\mathcal{I}_{c,2} = \{v_4, v_5, v_6\}$ (belonging to C), and $\mathcal{I}_{w,1} = \{v_2\}$, $\mathcal{I}_{w,2} = \{v_3\}$, $\mathcal{I}_{w,3} = \{v_7, v_8, v_9\}$, $\mathcal{I}_{w,4} = \{v_{10}, v_{11}, v_{12}\}$, $\mathcal{I}_{w,5} = \{v_{13}, v_{14}, v_{15}\}$ (belonging to W). The payoffs (utilities) are listed below the terminal nodes.

In this game, if the collusion is not initiated by C or is rejected by W, C and W will end up with playing Game 3, because there is no activated Collusion contract. If W agrees to collude with C and does not report to H, they will enter a subtree rooted at v_4, where the outcome is the same as Game 2, because there is no activated Betrayal contract. Otherwise, if W agrees to collude with C but also reports to H, they will enter subtrees rooted v_5 and v_6. The payoffs are decided jointly by the Watchtower, Collusion, and Betrayal contracts.

Next, we analyze Game 3 and show that if $w_h > c$ and $w_c > b > w_h - c + d_w$, both players will not initiate or agree to collude, and they will always send p. Formally, we have:

Theorem 4. *If $w_c > b > w_h - c + d_w$ and $w_h > c$, Game 4 has a unique sequential equilibrium (s_c^4, s_w^4):*

$$\begin{cases} s_c^4 = & (\neg init, s_c^3, s_c^3, p') \\ s_w^4 = & (s_w^3, collude, s_w^3, report\ p^* = p', p', p', p'). \end{cases}$$

According to this, C will not initiate the collusion, W will not report a collusion, C will send p, and W will respond with p.

Proof. We prove it by backward induction. Let the strategy profile in the sequential equilibrium be $s = (s_c^4, s_w^4)$, where

$$\begin{cases} s_c^4 = & ([\phi_1(\neg init), \phi_2(init)], s_c^3, s_c^3, \\ & [\phi_3(p), \phi_4(p'), \phi_5(other)]) \\ s_w^4 = & (s_w^3, [\psi_1(\neg collude), \psi_2(collude)], s_w^3, \\ & [\psi_3(\neg report), \psi_4(report\ p^* = p), \psi_5(report\ p^* \neq p)], \\ & [\psi_6(p), \psi_7(p'), \psi_8(other)], [\psi_9(p), \psi_{10}(p'), \psi_{11}(other)], \\ & [\psi_{12}(p), \psi_{13}(p'), \psi_{14}(other)]), \end{cases} \quad (1)$$

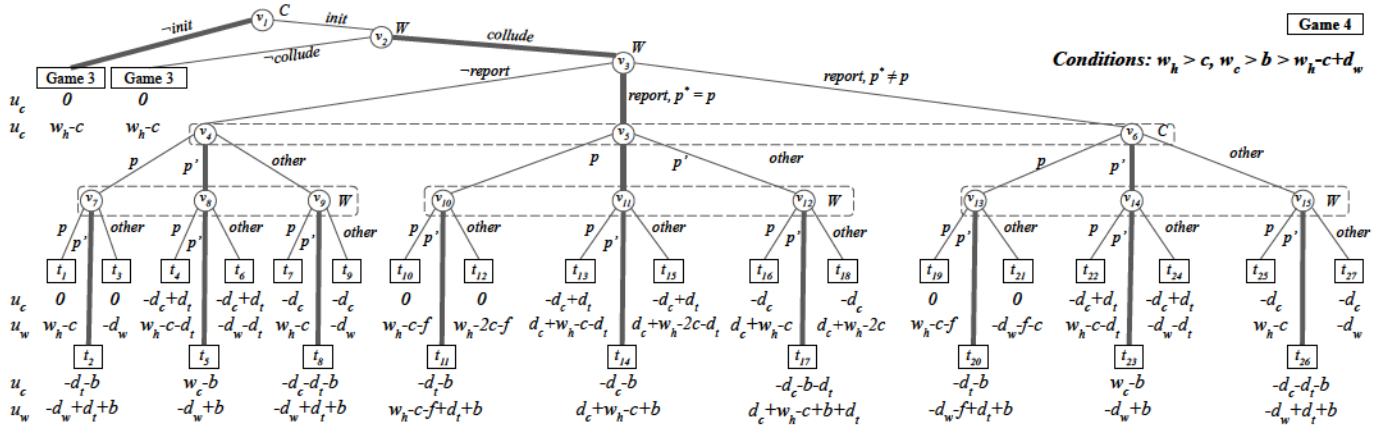

Fig. 6. Full game induced by the Watchtower contract, the Collusion contract, and the Betrayal contract

where ϕ_i and ψ_j are probabilities that satisfy the following: $0 \leq \phi_i \leq 1, \phi_1 + \phi_2 = 1, \phi_3 + \phi_4 + \phi_5 = 1$, $0 \leq \psi_j \leq 1, \psi_1 + \psi_2 = 1, \psi_3 + \psi_4 + \psi_5 = 1, \psi_6 + \psi_7 + \psi_8 = 1, \psi_9 + \psi_{10} + \psi_{11} = 1, \psi_{12} + \psi_{13} + \psi_{14} = 1$.

At the information set $\mathcal{I}_{w,3} = \{v_7, v_8, v_9\}$, W tries to maximize its expected payoff:

$$u_w(s; \mathcal{I}_{w,3}) = \phi_3 u_w(s; v_7) + \phi_4 u_w(s; v_8) + \phi_5 u_w(s; v_9), \quad (2)$$

where

$$\begin{cases} u_w(s; v_7) & = \psi_6 u_w(t_1) + \psi_7 u_w(t_2) + \psi_8 u_w(t_3), \\ u_w(s; v_8) & = \psi_6 u_w(t_4) + \psi_7 u_w(t_5) + \psi_8 u_w(t_6), \\ u_w(s; v_9) & = \psi_6 u_w(t_7) + \psi_7 u_w(t_8) + \psi_8 u_w(t_9). \end{cases}$$

It can be inferred that $\psi_6 = 0, \psi_7 = 1, \psi_8 = 0$, because the following holds when $w_h > c$ and $w_c > b > w_h - c + d_w$: $u_w(t_2) > u_w(t_1) > u_w(t_3)$; $u_w(t_5) > u_w(t_4) > u_w(t_6)$; $u_w(t_8) > u_w(t_7) > u_w(t_9)$. Thus, when $\psi_6 = 0, \psi_7 = 1, \psi_8 = 0$, it maximizes the expected payoff for W at $\mathcal{I}_{w,3}$. Therefore, we have $u_w(s; v_7) = u_w(t_2)$, $u_w(s; v_8) = u_w(t_5)$ and $u_w(s; v_9) = u_w(t_8)$. Substituting the above into Eq. (2), we have $u_w(s; \mathcal{I}_{w,3}) = \phi_3 u_w(t_2) + \phi_4 u_w(t_5) + \phi_5 u_w(t_8)$.

Similarly, at $\mathcal{I}_{w,4} = \{v_{10}, v_{11}, v_{12}\}$, when $\psi_9 = 0, \psi_{10} = 1, \psi_{11} = 0$, it maximizes the expected payoff for W. Also, at $\mathcal{I}_{w,5} = \{v_{13}, v_{14}, v_{15}\}$, when $\psi_{12} = 0, \psi_{13} = 1, \psi_{14} = 0$, it maximizes the expected payoff for W. Therefore, we have

$$\begin{cases} u_w(s; \mathcal{I}_{w,3}) & = \phi_3 u_w(t_2) + \phi_4 u_w(t_5) + \phi_5 u_w(t_8), \\ u_w(s; \mathcal{I}_{w,4}) & = \phi_3 u_w(t_{11}) + \phi_4 u_w(t_{14}) + \phi_5 u_w(t_{17}), \\ u_w(s; \mathcal{I}_{w,5}) & = \phi_3 u_w(t_{20}) + \phi_4 u_w(t_{23}) + \phi_5 u_w(t_{26}). \end{cases}$$

Moving backward, at $\mathcal{I}_{c,2} = \{v_4, v_5, v_6\}$, C tries to maximize its expected payoff, which is $u_c(s; \mathcal{I}_{c,2}) = \psi_3 u_c(s; v_4) + \psi_4 u_c(s; v_5) + \psi_5 u_c(s; v_6)$, where

$$\begin{cases} u_c(s; v_4) & = \phi_3(\psi_6 u_c(t_1) + \psi_7 u_c(t_2) + \psi_8 u_c(t_3)) \\ & \quad + \phi_4(\psi_9 u_c(t_4) + \psi_{10} u_c(t_5) + \psi_{11} u_c(t_6)) \\ & \quad + \phi_5(\psi_{12} u_c(t_7) + \psi_{13} u_c(t_8) + \psi_{14} u_c(t_9)), \\ u_c(s; v_5) & = \phi_3(\psi_6 u_c(t_{10}) + \psi_7 u_c(t_{11}) + \psi_8 u_c(t_{12})) \\ & \quad + \phi_4(\psi_9 u_c(t_{13}) + \psi_{10} u_c(t_{14}) + \psi_{11} u_c(t_{15})) \\ & \quad + \phi_5(\psi_{12} u_c(t_{16}) + \psi_{13} u_c(t_{17}) + \psi_{14} u_c(t_{18})), \\ u_c(s; v_6) & = \phi_3(\psi_6 u_c(t_{19}) + \psi_7 u_c(t_{20}) + \psi_8 u_c(t_{21})) \\ & \quad + \phi_4(\psi_9 u_c(t_{22}) + \psi_{10} u_c(t_{23}) + \psi_{11} u_c(t_{24})) \\ & \quad + \phi_5(\psi_{12} u_c(t_{25}) + \psi_{13} u_c(t_{26}) + \psi_{14} u_c(t_{27})). \end{cases}$$

By substituting $\psi_6 = 0, \psi_7 = 1, \psi_8 = 0$; $\psi_9 = 0, \psi_{10} = 1, \psi_{11} = 0$ and $\psi_{12} = 0, \psi_{13} = 1, \psi_{14} = 0$, we have

$$\begin{cases} u_c(s; v_4) & = \phi_3 u_c(t_2) + \phi_4 u_c(t_5) + \phi_5 u_c(t_8), \\ u_c(s; v_5) & = \phi_3 u_c(t_{11}) + \phi_4 u_c(t_{14}) + \phi_5 u_c(t_{17}), \\ u_c(s; v_6) & = \phi_3 u_c(t_{20}) + \phi_4 u_c(t_{23}) + \phi_5 u_c(t_{26}). \end{cases}$$

Because $u_c(t_5) > u_c(t_2) > u_c(t_8)$, it maximizes $u_c(s; v_4)$, when $\phi_3 = 0, \phi_4 = 1, \phi_5 = 0$. Similarly, it maximizes $u_c(s; v_5)$ and $u_c(s; v_6)$, when $\phi_3 = 0, \phi_4 = 1, \phi_5 = 0$. Thus, C will always choose $\phi_3 = 0, \phi_4 = 1, \phi_5 = 0$ to maximize $u_c(s; \mathcal{I}_{c,2})$. Thus, we have $u_c(s; v_4) = u_c(t_5)$, $u_c(s; v_5) = u_c(t_{14})$ and $u_c(s; v_6) = u_c(t_{23})$.

Moving backward, at $\mathcal{I}_{w,2} = \{v_3\}$, W tries to maximize its expected payoff, which is

$$u_w(s; \mathcal{I}_{w,2}) = \psi_3 u_w(t_5) + \psi_4 u_w(t_{14}) + \psi_5 u_w(t_{23}). \quad (3)$$

Because $u_w(t_{14}) > u_w(t_5) = u_w(t_{23})$, W will always choose $\psi_3 = 0, \psi_4 = 1, \psi_5 = 0$ to maximize $u_w(s; v_3)$. Substituting the above into Eq. (3), we have $u_w(s; \mathcal{I}_{w,2}) = u_w(t_{14})$.

Moving backward, at $\mathcal{I}_{w,1} = \{v_2\}$, W tries to maximize its expected payoff: $u_w(s; \mathcal{I}_{w,1}) = u_w(s; v_2) = \psi_1 u_w(Game\ 3) + \psi_2 u_w(t_{14})$. Because $u_w(t_{14}) > u_w(Game\ 3)$, W will always choose $\psi_1 = 0, \psi_2 = 1$ to maximize $u_w(s; v_2)$. Thus, we have $u_w(s; \mathcal{I}_{w,1}) = u_w(t_{14})$.

Moving backward, at $\mathcal{I}_{c,1} = \{v_1\}$, C tries to maximize its expected payoff, which is $u_c(s; \mathcal{I}_{c,1}) = u_c(s; v_1) = \phi_1 u_c(Game\ 3) + \phi_2 u_c(t_{14})$. Because $u_c(Game\ 3) > u_c(t_{14})$, W will always choose $\phi_1 = 1, \phi_2 = 0$ to maximize $u_c(s; v_1)$. Therefore, $u_c(s; \mathcal{I}_{c,1}) = u_c(s; v_1) = u_c(Game\ 3)$.

Since Game 3 terminates at t_1 in Game 1, Game 4 terminates at t_1 in Game 1. Substituting ϕ_i and ψ_j into Eq. (1), we have:

$$\begin{cases} s_c^4 = & (\neg init, s_c^3, s_c^3, p') \\ s_w^4 = & (s_w^3, collude, s_w^3, report\ p^* = p', p', p', p'). \end{cases}$$

Thus, the unique sequential equilibrium has been proven. □

B. Insights on Contract Design

According to Theorem 4, two conditions ($w_c > b > w_h - c + d_w$ and $w_h > c$) must be satisfied to have the unique sequential equilibrium. When $b \leq w_h - c + d_w$, the collusion will not happen, because there would be no motivation for W to collude, according to the analysis on the Collusion contract. In addition, it is obvious that $w_c > b$, since C would not pay W more bribe than its gain from the collusion. Also, $w_h > c$

is set that by H in the Watchtower contract, because W does not accept unpaid jobs. Therefore, these three contracts are effective to counter collusion, as long as $w_c > w_h - c + d_w$. The monetary variables w_h and d_w can be set by H in the Watchtower contract. Therefore, as long as H sets $d_w < w_c$, we have $w_c > w_h - c + d_w$. H can infer w_c based on the transaction history of this channel.

It can also be observed that the watchtower is required to lock a small and reasonable collateral $d_w < w_c$. Since the value of cryptocurrencies fluctuates dramatically, the time value of money (TVM) concept in financial management also applies to cryptocurrencies, or might even contribute more according to the historic volatility. Thus, a small collateral can incentivize the watchtowers to participate and guarantee security in PCNs.

Based on the analysis of Game 4, the rational watchtower and counterparty will not collude or cheat with the existence of all the three contracts. Thus, the rational parties will never execute the Collusion or Betrayal contracts, even if allowed. This indicates that the rational parties are involved in minimal operations on the blockchain. In addition, the Watchtower contract [5] is compilable with the PCN protocol and does not require on-chain storage or execution.

IX. Implementation

We provide a proof-of-concept implementation of the proposed smart contracts to evaluate the scalability and efficiency. In the following, we present a high-level overview of the experiments. Because the Watchtower contract has been embedded in the Bitcoin Lightning [7] protocol, our experiment involves two contracts: the Collusion and Betrayal contracts. We implement the contracts in Solidity and execute them on Ethereum with Geth [8]. The contracts are loosely coupled with the actual channel monitoring jobs as an external service. The actual channel monitoring jobs can be treated as blackboxes, and the contracts do not need to know their internal details.

A. Scalability

As we surveyed in Section II, Cerberus [5, 10] has provided an implementation of the watchtower functionality, which is compilable with the Lightning Network [7] protocol. Specifically, it allows the hiring party to employ a watchtower and enable the watchtower to lock and reclaim a deposit, which has the similar logic steps as our Watchtower contract except for Cerberus's requirement of a large deposit. This extension of the Lightning Network has shown that watchtowers can scale well in PCNs, *i.e.*, the number of on-chain transactions is constant and independent of the number of transactions in an off-chain channel, similarly to the current Lightning protocol.

B. Overhead and Financial Cost

According to the analysis and discussion in Section VIII, The rational parties will never execute the Collusion or Betrayal contracts. The Watchtower contract is compilable with the PCN protocol, and does not require executions on the blockchain. Thus, we evaluate the cost of executing the Collusion and Betrayal contracts on Ethereum, in case that a party happens to sign and execute the contracts. The results are presented in Table I. The cost is in the amount of gas consumed by each function, and the converted monetary value in US dollar (\$). The gas price was 2×10^{-9} ether (2 Gwei) and the exchange rate was 1 ether = \$158.52 as of April 2020. The total cost of the Collusion contract is about 2 million gas (\$0.63). The total cost of the Betrayal contract is about 2.4 million gas (\$0.76).

Contract	Operation	Cost(Gas)	Cost(\$)
Collusion	Init	1,637,844	0.5139
	Create	225,583	0.0708
	Collude	62,733	0.0197
	Distribute	95,242	0.0299
Betrayal	Init	1,672,264	0.5247
	Create	137,546	0.0432
	Betray	68,725	0.0216
	Distribute	544,476	0.1708

TABLE I
Cost of using the smart contracts on Ethereum. We have approximated the cost in USD (\$) using the conversion rate of 1 ether = \$156.89 and the gas price of 2 Gwei which reflects the real world costs as of April 2020.

The financial cost for running the smart contracts is roughly related to the computational and storage complexity of the function. For example, the cost of the $Init$ operation is dramatically higher than the other operations. The fundamental reason is that the cost of data storage on the blockchain is very expensive. Thus, the execution cost can be significantly reduced by recycling the contracts. Specifically, a hiring party may outsource the channel monitoring job to a watchtower from time to time. Thus, it is not cost-effective or necessary to initiate a new smart every time. In the smart contracts, there is a $reset()$ function that can be called by the contract owner after the contract has concluded. This function can clean up the contract and reset it to the initial state, which allows the reusage of contracts.

X. Conclusion

In this paper, we studied the collusion problem for watchtowers in PCNs, where a hiring party outsources the payment channel monitoring job to a watchtower, but the watchtower could benefit from coordinating with the counterparty and not performing the job. To address this problem, we proposed a smart contract based solution by leveraging economic approaches. Specifically, we designed three smart contracts, the Watchtower, Collusion, and Betrayal contracts, to bring distrust between parties and guarantee security in PCNs. We conducted detailed analyses of the contracts and rigorously proved that there is a unique sequential equilibrium, where the rational parties will not collude or cheat and are involved in minimal on-chain operations. In particular, a watchtower only needs to lock a small deposit, which incentivizes the participation of watchtowers and users. Moreover, a proof-of-concept implementation of the smart contracts was provided and executed on Ethereum. The performance demonstrated that the contracts are gas-efficient.

REFERENCES

[1] "Bitcoin lightning fraud? laolu is building a watchtower to fight it." [Online]. Available: https://www.coindesk.com/laolu-building-watchtower-fight-bitcoin-lightning-fraud

[2] "Bitcoin wallet electrum now supports lightning, watchtowers and submarine swaps." [Online]. Available: https://www.coindesk.com/bitcoin-wallet-electrum-now-supports-lightning-watchtowers-and-submarine-swaps

[3] "Blockstreams watchtowers will bring a new justice system to the lightning network." [Online]. Available: https://www.coindesk.com/blockstreams-watchtowers-will-bring-a-new-justice-system-to-the-lightning-network

[4] "c-lightning Daemon." [Online]. Available: https://github.com/ElementsProject/lightning/tree/master/lightningd/

[5] "Cerberus script." [Online]. Available: https://github.com/OrfeasLitos/cerberus-script

[6] "Fraud-fighting watchtowers to arrive in next bitcoin lightning release." [Online]. Available: https://www.coindesk.com/fraud-fighting-watchtowers-are-coming-with-the-next-big-lightning-release

[7] "The lightning network." [Online]. Available: https://lightning.network/

[8] "Ethereum wiki: Geth," 2015. [Online]. Available: https://github.com/ethereum/go-ethereum/wiki/geth

[9] "Lnd: The lightning network daemon," 2017. [Online]. Available: https://github.com/lightningnetwork/lnd

[10] G. Avarikioti, O. S. T. Litos, and R. Wattenhofer, "Cerberus channels: Incentivizing watchtowers for bitcoin," *FC*, 2020.

[11] 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 *FC*. Springer, 2016, pp. 106–125.

[12] C. Decker and R. Wattenhofer, "A fast and scalable payment network with bitcoin duplex micropayment channels," in *SSS*. Springer, 2015, pp. 3–18.

[13] C. Dong, Y. Wang, A. Aldweesh, P. McCorry, and A. van Moorsel, "Betrayal, distrust, and rationality: Smart counter-collusion contracts for verifiable cloud computing," in *CCS*. ACM, 2017, pp. 211–227.

[14] T. Dryja and S. B. Milano, "Unlinkable outsourced channel monitoring," *Talk transcript) https://diyhpl.us/wiki/transcripts/scalingbitcoin/milan/unlinkable-outsourced-channel-monitoring*, 2016.

[15] Ethereum, 2015. [Online]. Available: https://www.ethereum.org/

[16] J. Garay, A. Kiayias, and N. Leonardos, "The bitcoin backbone protocol: Analysis and applications," in *Eurocrypt*. Springer, 2015, pp. 281–310.

[17] M. Khabbazian, T. Nadahalli, and R. Wattenhofer, "Outpost: A responsive lightweight watchtower," in *AFT*. ACM, 2019, pp. 31–40.

[18] K. Leyton-Brown and Y. Shoham, "Essentials of game theory: A concise multidisciplinary introduction," *Synthesis lectures on artificial intelligence and machine learning*, vol. 2, no. 1, pp. 1–88, 2008.

[19] P. McCorry, S. Bakshi, I. Bentov, A. Miller, and S. Meiklejohn, "Pisa: Arbitration outsourcing for state channels." *IACR Cryptology ePrint Archive*, vol. 2018, p. 582, 2018.

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

[21] O. Osuntokunn, "Hardening lightning," *BPASE*, 2018.

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

[23] P. Prihodko, S. Zhigulin, M. Sahno, A. Ostrovskiy, and O. Osuntokun, "Flare: An approach to routing in lightning network," in *Whitepaper*, 2016.

[24] Raiden, 2018. [Online]. Available: https://raiden.network/

[25] F. Reid and M. Harrigan, "An analysis of anonymity in the bitcoin system," in *Security and privacy in social networks*. Springer, 2013, pp. 197–223.

[26] M. Trillo, "Stress test prepares visanet for the most wonderful time of the year (2013)," 2013.

[27] R. Yu, G. Xue, V. T. Kilari, D. Yang, and J. Tang, "CoinExpress: A fast payment routing mechanism in blockchain-based payment channel networks," in *ICCCN*. IEEE, 2018, pp. 1–9.

[28] Y. Zhang and D. Yang, "RobustPay: Robust payment routing protocol in blockchain-based payment channel networks," in *ICNP*. IEEE, 2019.

[29] Y. Zhang, D. Yang, and G. Xue, "CheaPay: An optimal algorithm for fee minimization in blockchain-based payment channel networks," in *ICC*. IEEE, 2019.