Fair and Sound Secret Sharing from Homomorphic Time-Lock Puzzles

Jodie Knapp and Elizabeth A. Quaglia

Information Security Group, Royal Holloway, University of London,
jodie.knapp.2018@rhul.ac.uk, elizabeth.quaglia@rhul.ac.uk

Abstract. Achieving fairness and soundness in non-simultaneous rational secret sharing schemes has proved to be challenging. On the one hand, soundness can be ensured by providing side information related to the secret as a check, but on the other, this can be used by deviant players to compromise fairness. To overcome this, the idea of incorporating a time delay was suggested in the literature: in particular, time-delay encryption based on memory-bound functions has been put forth as a solution. In this paper, we propose a different approach to achieve such delay, namely using homomorphic time-lock puzzles (HTLPs), introduced at CRYPTO 2019, and construct a fair and sound rational secret sharing scheme in the non-simultaneous setting from HTLPs.

HTLPs are used to embed sub-shares of the secret for a predetermined time. This allows to restore fairness of the secret reconstruction phase, despite players having access to information related to the secret which is required to ensure soundness of the scheme. Key to our construction is the fact that the time-lock puzzles are homomorphic so that players can compactly evaluate sub-shares. Without this efficiency improvement, players would have to independently solve each puzzle sent from the other players to obtain a share of the secret, which would be computationally inefficient. We argue that achieving both fairness and soundness in a non-simultaneous scheme using a time delay based on CPU-bound functions rather than memory-bound functions is more cost effective and realistic in relation to the implementation of the construction.

1 Introduction

Threshold secret sharing (SS) schemes provide a way to split a secret into shares such that the secret can be reconstructed by a threshold number of mutually distrustful parties. Knowledge of fewer than the threshold number of shares reveals nothing about the secret [6,40]. SS schemes are an important primitive used in a variety of settings from multiparty computation [8,11], to attribute-based encryption [23,44], and threshold cryptography [5,14]. In a SS scheme, a trusted dealer splits the secret into shares and distributes one to each authorised party. Parties then communicate and process their collective shares in a reconstruction phase. During the communication phase, parties broadcast their shares in one of two ways: *simultaneously* or *non-simultaneously*. That is, with or without synchronicity. Properties of SS schemes are better understood, and easier to guarantee in the simultaneous setting [12], due to the fact that a non-simultaneous construction needs to ensure the final party to communicate is still incentivised to follow the protocol. However, simultaneous schemes are difficult to implement in practice, therefore attention has recently turned to non-simultaneous communication [3].

Typically, in the non-simultaneous setting [19,28,29], schemes consists of rounds, where one round of the reconstruction phase simply translates to a capped period of time in which parties have the opportunity to communicate their share. Parties learn the secret is reconstructed when they reconstruct some publicly known value (for example, an indicator), in what is known as a revelation round [32,25]. The previous round to the revelation round is assumed to be the one in which the secret can be reconstructed from, allowing parties to identify when they will reconstruct the correct secret.

There is abundant literature for cryptographic [4,5,9,21,25,30,31,32,37,41] and game-theoretical SS schemes [3,12,19,22,24,33], two somewhat independent research areas considering honest/malicious parties and rational players, respectively. We refer to Appendix G for a brief summary of past works. Rational secret sharing (RSS) was introduced by [24], where they consider the problem of secret sharing and multiparty computation assuming players prefer to learn the secret over not learning it, and secondly, prefer that as few as possible other players learn the secret. While for some applications the cryptographic setting is appropriate, for other applications of secret sharing it may be more suitable to view all parties as rational players. RSS is a good approach to capture more interesting scenarios, such as how to motivate or force players to participate honestly and even how a scheme can penalise players for deviant play. Furthermore, modelling players as rational is not limited to assuming players always want to learn the secret above all else. Indeed, as we will explore, an emerging scenario in RSS considers players that prefer to mislead others above learning the secret. For these reasons, our attention focuses on RSS schemes.

In RSS schemes, the outcome of the game influences the players' strategies, as they seek to maximise their payoff. Security of the game requires the strategies of players to be in some form of equilibrium which motivates them to honestly communicate[1]. Achieving an equilibrium between players' strategies is the most natural way to demonstrate a fundamental property of SS schemes, called *fairness* [16,27].

A fair scheme ensures that if a player deviates, the probability that they can recover the shared secret over honest players is negligible. That is, a player is at no advantage in learning the secret if they withhold or dishonestly send a share. In the simultaneous setting, [22,24] both achieve fairness using some form of publicly known indicator, and by demonstrating that their protocol is in a form of Nash equilibrium [36]. In the non-simultaneous setting, however, a basic threat to fairness arises: in a (t, n) threshold RSS scheme, the last player out of

[1] See Appendices D.1, and D.2 for further discussion on payoff functions and equilibrium concepts.

t can decide not to communicate their share and use all the other players' shares to reconstruct the secret, leaving the $(t-1)$ honest players with an insufficient number of shares to do so. The rational behaviour of all parties would therefore be to withhold their share. In works such as [19,29,33], fairness can be achieved similarly to the simultaneous setting, whereby players can recognise the revelation round using (or reconstructing) some form of public indicator. However, this only works under the assumption that players prefer everyone to obtain the correct output over misleading others [3,12]. If this assumption does not hold, an alternative way of providing fairness needs to be used, as another property of SS schemes is no longer ensured, *soundness*.

In RSS schemes, *soundness* [3,12] ensures players never reconstruct an incorrect secret except with negligible probability. In other words, honest players are guaranteed to output a correct value, or a special abort symbol $\perp$ [3]. Soundness is becoming of emerging relevance in the non-simultaneous setting, assuming rational players obtain a greater payoff from misleading other players compared to learning the secret. Soundness has been achieved in prior work [12] focusing on non-simultaneous communication as follows: before reconstruction begins, all players are given protocol-induced side information alongside their list of shares. They must assume that when a player aborts communication, the previous round was the revelation round. Even if a deviant player has aborted early, using this side information, honest players can check that they have the correct value after reconstruction. If not, they terminate the reconstruction altogether.

However, achieving soundness this way compromises fairness, as a deviant player can use the side-information to check whether they can abort early and learn the secret before honest players. The authors of [33] were the first to propose a fair RSS scheme that can tolerate *arbitrary* side-information, by proposing the use of time-delay encryption (TDE) [7,35]. The basic idea of a TDE scheme is to encrypt a message such that it can only be decrypted after a specific amount of time has elapsed. The scheme in [33] employs a cryptographic memory-bound function[2] (CMBF) [1,18] as a way to achieve time-delay in the recovery of an encrypted sub-share of the secret. The fairness of their scheme is restored by setting the runtime of rounds of the secret sharing scheme to be less than the time it takes to decrypt the encrypted shares. Thus, there is no way for a deviant player to learn anything about the secret during a reconstruction round, before they must decide whether to abort communication. In addition, a proof of the sender's work in computing their message is sent. The scheme proposed in [12] builds upon [33], by encrypting shares (shares are computed using Shamir's SS scheme) using the CMBF and further splits the encrypted shares into sub-shares, distributed to players. During processing, players independently evaluate the encrypted sub-shares to obtain the encrypted share, decrypt and then reconstruct the polynomial to obtain the secret. They use a *specific* form of side-information, called a checking share, which is an actual share of the se-

[2] A CMBF is a family of deterministic algorithms such that an efficiently generated key can decrypt the encrypted input, with a lower-bound on the number of memory-access steps to do so.

cret that players can use to confirm they have reconstructed the correct secret, thus achieving soundness.[3] We note that the memory-bound running times of employing the MBF in [18], a cryptographic version of which is used for time-delay in [12,33], endows a high cost on the players who have to verify the proof of work from messages received by other players. In addition, the players sending the message can potentially perform less work than what is stated in their accompanying proof [17,39]. These drawbacks suggest that a better time-delay mechanism should be explored to guarantee fairness, that reduces verification costs of the communicated messages and\or increases computational efficiency for honest players obtaining the secret shares after the delay.

1.1 Our Contributions

In this paper, we improve on [12] and propose a RSS scheme achieving fairness and soundness in the non-simultaneous communication setting from a CPU-bound function, as opposed to a CMBF, namely a *homomorphic time-lock puzzle*.

Informally, a time-lock puzzle (TLP) [38] embeds a secret into a puzzle such that it cannot be decrypted until a certain amount of time $\mathcal{T}$ has elapsed. Characteristics of a TLP include fast puzzle generation and security against parallel algorithms, assuming the sequentiality of the underlying mathematical problem [38]. A *homomorphic* time-lock puzzle (HTLP) scheme evaluates puzzles homomorphically using some operation, without the evaluator knowing the secret shares encapsulated within the corresponding puzzles. The resulting puzzle output contains the homomorphic evaluation of the input puzzles, enabling a more efficient way for decryptors to obtain the final output solution, as they can solve just one puzzle rather than solving all of the puzzles individually with standard TLPs, and then evaluating a final solution.

In our scheme, the dealer splits the secret into shares, and creates an additional share which is broadcast to all players, i.e., the checking share. The rest of the shares are split into sub-shares, embedded into HTLPs and distributed to the corresponding players in such a way that the HTLP scheme can reconstruct the share from them. Intuitively, the checking share is used to verify the soundness of the secret that players reconstruct, and the delay provided by the HTLP scheme is used to guarantee fairness in the presence of a checking share for players communicating non-simultaneously. More specifically, the HTLP scheme embeds the sub-shares into puzzles that cannot be decrypted before a round of communication in the reconstruction phase has finished. Fairness is achieved by setting each round of communication to have an upper time bound of $\mathcal{T}$. Thus, a player wishing to deviate from their prescribed strategy and quit communication will not be able to derive the secret before the end of the round, in which case, the other players realise the deviant player has quit and output the result of the previous rounds reconstruction. We show that even if a player quits in a round

[3] Note that [33] works under the assumption that players prefer everyone to obtain the correct output over misleading others, therefore soundness is not an issue that needs to be addressed.

and manages to learn the secret, the only case in which they can do so results in the honest players also learning the secret. Therefore there is no advantage in a player deviating from their prescribed strategy.

From our generic construction, which we show satisfies soundness and fairness, we provide a concrete instantiation using the multiplicative variant of the HTLP scheme proposed in [34]. The result is a concrete, efficient scheme whose security relies on standard assumptions.

We argue that our improvement on prior work is threefold: we base the time delay of the construction on CPU-bound functions, as opposed to CMBFs; we provide an efficiency gain by using HTLPs instead of TLPs; and our solution has inherent flexibility.

Basing the time-delay primitive on CPU-bound functions as opposed to memory-bound functions captures a more realistic, inexpensive way to implement a SS scheme construction. Processors are faster than memory and scale better; even more so, fast memory is considerably more expensive. In practice, it is easier to raise the computational requirements of a player than it is memory accesses, up to a point, as adding more processors to a computer is more accessible than making memory accesses faster. A justification for using MBFs in [12,33] is that disparities in the computational power of players can cause unfairness when using standard TLPs for time-delay. However, with reasonable assumptions on the CPU-power of players, this disparity is not significant.

Furthermore, we use a HTLP for time-delay, which requires less computational work on behalf of the players decrypting puzzles compared to using standard TLPs. This efficiency improvement means that the consequence of disparities in CPU-power becomes less significant. To see this, evaluating several puzzles homomorphically, and then solving just *one* puzzle, requires fewer computational steps than solving individual puzzles and evaluating a function over the outputs, as in [12].

Finally, the instantiation of our generic scheme can use any correct SS scheme with a suitable HTLP, dependent on the application. The HTLPs that we use, from [34], are adaptable in the following ways: different operators (linear, multiplicative, and XOR) can be used, we can augment the setup with puzzles of different time hardness parameters $(\mathcal{T}_1, \ldots, \mathcal{T}_n)$ or have a reusable setup, in which the scheme remains efficiently computable.

2 Definitions and Modelling

2.1 Secret Sharing

Informally, a (t, n) secret sharing scheme (SS) involves a dealer D, some secret s, and a set $P = \{P_1, P_2, \ldots, P_n\}$ of n players. The dealer distributes shares of a secret s chosen according to an efficiently samplable distribution of the set of secrets, labelled $\mathcal{S} = \{\mathcal{S}_\lambda\}_{\lambda \in \mathbb{N}}$, with security parameter λ. The key idea behind threshold SS is that no subset $t' < t$ of players in P can learn the secret s, including an adversary controlling t' players. Conversely, every subset $t' \geq t$ of players in P is capable of reconstructing s.

A SS protocol is composed of two phases, share and reconstruction. During the share phase, the dealer samples a secret s from $\mathcal{S}_\lambda$ and generates n shares from the secret to be distributed to each player in P. The dealer does this non-interactively, using the a share algorithm to generate the set of shares to be distributed. The dealer digitally signs (typically using information theoretically secure MACs) and encrypts the shares before distributing them to individual players over a broadcast channel [4].

The reconstruction phase itself is composed of two parts: communication and processing. The communication phase has players interact by sending their share over the broadcast channel to every other player in P (if a broadcast channel is not available to parties, then they have to send their share to each of the other players separately). Once players have communicated, they can move to the processing phase where they embark on reconstructing the secret s from the shares that they have received. This is under the assumption that a sufficient number of shares have been sent and received from other players, and that players followed the protocol (correctness). If an insufficient amount of shares have been received, the secret cannot be reconstructed, so players output $\perp$. Any player taking part in reconstruction proceeds to output their result.

Threshold secret sharing schemes have been explored extensively, and were introduced independently by Shamir [40] and Blakley [6]: Shamir's scheme is based on polynomial interpolation over a finite field of prime order, and Blakley's scheme is based on the uniqueness of hyperplane intersection. Extending the work of [40], [13,15,43] propose multiplicative homomorphic secret sharing schemes based on polynomial interpolation over finite groups with respect to multiplication, that need not be of prime order (See Appendix B).

Next, we recall the formal definition of a threshold secret sharing scheme, with the implicit assumption that the dealer has digitally signed the shares before distributing:

Definition 1 (**(t,n) Secret Sharing**). *Given a dealer D, a secret $s \in \mathcal{S}_\lambda$ for security parameter λ, and a set of n authorised players $P = \{P_1, \ldots, P_n\}$, a (t,n) secret sharing scheme is a tuple of three PPT algorithms (Setup, Share, Recon) defined as follows:*

- **Share Phase**: D takes as input the secret s and performs the following steps non-interactively:
  1. $pp \leftarrow \mathsf{Setup}(1^\lambda)$ a probabilistic algorithm that takes as input security parameter 1^λ and outputs public parameters pp, which are broadcast to all players in P.
  2. $\{s_1, \ldots, s_n\} \leftarrow \mathsf{Share}(pp, s)$ a probabilistic algorithm that takes as input the secret $s \in \mathcal{S}_\lambda$ and outputs n shares s_i, one for each player in P.
  3. Distribute s_i to player P_i for every $i \in [n]$ over a secret, authenticated channel.

[4] Privacy and authentication of the distribution of shares is a standard cryptographic assumption in secret sharing schemes [37].

- **Reconstruction Phase**: Any player in $P = \{P_1, \ldots, P_n\}$ is able to take part in this phase.
  1. Communication:
     (a) Each player P_i sends their share s_i over a secure broadcast channel to all other players in P.
     (b) P_i checks that they have received $(t-1)$ or more shares. If so, they proceed to processing.[5]
  2. Processing:
     Once P_i has a set of t' shares labelled S', they independently do the following:
     (a) $\{s, \perp\} \leftarrow \mathsf{Recon}(pp, S')$ a deterministic algorithm that takes as input the set S' of t' shares and outputs the secret s if $t' \geq t$ or outputs abort $\perp$ otherwise.

A (t, n) threshold SS scheme needs to satisfy the properties of correctness and secrecy, whose definitions are provided in Appendix A.3. Informally, correctness means that an honest execution of the scheme results in the true secret being output, except with negligible probability; and secrecy ensures that reconstruction with fewer shares than the threshold (t) results in abort $(\perp)$ being output, except with negligible probability.

2.2 Rational Secret Sharing

Using game-theory notions, players are considered to be rational if they have a preference in the outcome of the reconstruction phase. In a rational secret sharing (RSS) scheme, a players strategy is to maximise their payoff from the outcome of the game. The strategy σ_i taken by each player P_i must be determined by the dealer in order to achieve a fair outcome. Observe that depending on the scheme, the strategies of players in P may be the same or different.

In Definition 1, players only participate in the reconstruction phase. Therefore, we define a RSS scheme by providing a definition of the reconstruction phase only.

Definition 2 ((t,n) Rational Secret Reconstruction [12]). *A reconstruction phase $\Gamma_{t,n}$ is defined by $\Gamma_{t,n} = (\Gamma, \overrightarrow{\sigma})$ where Γ is the game to be played by players during the reconstruction phase and $\overrightarrow{\sigma} = (\sigma_1, \cdots, \sigma_n)$ denotes the strategy profile of the players in P prescribed by the dealer D during the share phase for that scheme.*

The outcome of the phase for all players is defined by the n-dimensional vector

$$\overrightarrow{\omega}((\Gamma, \overrightarrow{\sigma})_{t,n}) = (\omega_1, \ldots, \omega_n)$$

[5] Whilst not explicit in the definition, there is an upper bound on how long players can communicate their shares for. Therefore, at the end of their communication, if a player P_i has not obtained a sufficient number of shares, then they output $\perp$ at the end of the reconstruction phase.

where ω_i refers to the outcome of the phase for player P_i.

The outcome ω_i alludes to whether player P_i learns the entirety of s, nothing of s, is mislead into learning a fake secret s' or aborts the reconstruction phase altogether ($\perp$). It is important to note that the outcome of the phase depends on the strategy of the player.

One of the fundamental properties of secret sharing is *fairness* [42], which guarantees that no player has an advantage in the protocol over other players. The following defines fairness in the context of a RSS scheme. We use the following notation for a deviating strategy σ'_i for player P_i, to signify when a player behaves in a different way to how they are meant to. That is, they do not follow the protocol. In addition, P_{-i} represents all players in P excluding player P_i, and σ_{-i} signifies the honest strategies of this set of $(n-1)$ players, P_{-i}.

Definition 3 (Fairness [12]). *The reconstruction phase $\Gamma_{t,n}$ is* completely fair *if for every arbitrary alternative strategy σ'_i followed by player P_i for some $i \in [n]$, there exists a negligible function μ in the security parameter λ such that the following holds:*

$$\Pr[\omega_i(\Gamma, (\sigma'_i, \sigma_{-i})) = s] \leq Pr[\omega_{-i}(\Gamma, (\sigma'_i, \sigma_{-i})) = s] + \mu(\lambda).$$

That is, the probability of player P_i learning the secret when they deviate from their prescribed strategy in phase $\Gamma_{t,n}$ (but all other players follow their prescribed strategies) is only ever negligibly more than the probability of the other players learning the secret too. Consequently, such a player has no real advantage in deviating from their strategy.

How do we ensure that players (despite any preferences they may have) are motivated to follow a strategy in the non-simultaneous setting? This is typically done by assuming that the strategies of players are in a computationally strict Nash equilibrium (or some other variant of a Nash equilibrium) [16,27,36]. This concept makes certain that if every player $P_i \in P$ believes all other players in P are following their prescribed strategy in the phase, then they have nothing to gain in deviating from their own strategy and are penalised in some way by deviating. In our construction, we need to ensure players strategies are in a computationally strict Nash equilibrium when they additionally have access to side-information related to the secret. We discuss this further in Appendix D.2.

Another fundamental property of RSS is soundness. Simply put, soundness of the reconstruction phase output means that the probability of players following the scheme outputting an incorrect secret when another player deviates from their own strategy is negligible.

Definition 4 (Soundness [12]). *Reconstruction phase $\Gamma_{t,n}$ is* sound *if for every arbitrary alternative strategy σ'_i followed by player P_i for $i \in [n]$, there exists a negligible function μ in the security parameter λ such that the following holds:*

$$\Pr[\omega_{-i}(\Gamma, (\sigma'_i, \sigma_{-i})) \notin \{s, \perp\}] \leq \mu(\lambda)$$

In our construction, as we shall see, we achieve this property by using a checking share, similarly to [12]. A checking share is an actual share of the secret, kept separate from the other shares and publicly broadcast to players. In order to formalise our scheme, discussed in Section 3, we recall the definition of a homomorphic time-lock puzzle (HTLP) [34], on which our construction relies.

2.3 Homomorphic TLPs

Informally, a time-lock puzzle (TLP) scheme embeds a secret into a puzzle such that it cannot be decrypted until a certain amount of time $\mathcal{T}$ has elapsed. The seminal work of [38] outlined the characteristics of a TLP:

- Fast puzzle generation: namely, the time t required to generate a puzzle $\mathcal{Z}$ must be $t << \mathcal{T}$, for a given (time) hardness parameter $\mathcal{T}$.
- Security against parallel algorithms: that is, the encapsulated secret s is disguised within the puzzle $\mathcal{Z}$ for circuits of depth $< \mathcal{T}$, regardless of the size of the circuit.

However, when the decryptor is faced with a significant number of puzzles to solve, a standard TLP scheme requires the decryptor to solve each individual puzzle, which could be very inefficient. Driven by this limitation [34] introduced the notion of a homomorphic TLP (HTLP), a scheme that compactly evaluates puzzles homomorphically.

Homomorphic time-lock puzzles are augmented TLPs allowing anyone to evaluate a circuit C over sets of puzzles $(\mathcal{Z}_1, \ldots, \mathcal{Z}_n)$ homomorphically using operation Ψ [6], without the evaluator necessarily knowing the secret values $(s_1, \ldots, s_n)$ encapsulated within the corresponding puzzles. The resulting output (a puzzle $\mathcal{Z}$) contains the circuit output $C(s_1, \ldots, s_n)$, and the hardness parameter $\mathcal{T}$ does not depend on the size of the circuit C that was evaluated (this is called compactness).

Definition 5 (HTLP [34]). *Let $C = \{\mathcal{C}_\lambda\}_{\lambda \in \mathbb{N}}$ be a class of circuits and let secret space $\mathcal{S}_\lambda$ be a finite domain for security parameter λ. A homomorphic time-lock puzzle (HTLP) with respect to C and $\mathcal{S}_\lambda$ is defined by a tuple of four PPT algorithms ($\mathsf{HP.Setup}$, $\mathsf{HP.Gen}$, $\mathsf{HP.Solve}$, $\mathsf{HP.Eval}$) as follows:*

- $pp \leftarrow \mathsf{HP.Setup}(1^\lambda, \mathcal{T})$ is a probabilistic algorithm that takes as input security parameter 1^λ and hardness parameter $\mathcal{T}$ and outputs public parameters pp.
- $\mathcal{Z} \leftarrow \mathsf{HP.Gen}(pp, s)$ a probabilistic algorithm that takes as input the public parameters pp and a secret $s \in \mathcal{S}_\lambda$ and outputs a puzzle $\mathcal{Z}$.
- $s \leftarrow \mathsf{HP.Solve}(pp, \mathcal{Z})$ is a deterministic algorithm that takes as input public parameters pp and puzzle $\mathcal{Z}$, and outputs a solution s.

[6] What Ψ is depends on the application the HTLP is being used for. It could be addition, multiplication or XOR for example.

– $\tilde{\mathcal{Z}} \leftarrow \mathsf{HP.Eval}(pp, C, \Psi, \mathcal{Z}_1, \ldots, \mathcal{Z}_n)$ is a probabilistic algorithm taking as input a circuit $C \in \mathcal{C}_\lambda$, parameters pp, homomorphic-operation Ψ, and a set of n puzzles $(\mathcal{Z}_1, \ldots, \mathcal{Z}_n)$, and outputs a master puzzle $\tilde{\mathcal{Z}}$.

A HTLP scheme should satisfy correctness, security, and compactness. Informally, correctness means that if a scheme is executed properly, then the probability of the output being anything other than the solution is negligible. Captured within the definition of correctness of [34] is the time-delay in solving a HTLP. Informally, given a puzzle evaluated in the scheme, there exists a fixed polynomial over the security and time hardness parameters which bounds the runtime solving the puzzle in the HTLP scheme.

Intuitively, a scheme is considered secure if the output of execution is indistinguishable from random to an eavesdropping adversary. Compactness is a non-trivial property requiring that the complexity of decrypting an evaluated ciphertext does not depend on the function used to evaluate the ciphertext. Intuitively, it means that the ciphertext size should not grow through homomorphic operations and the output length of the homomorphically evaluated ciphertext only depends on the security parameter. In the context of a HTLP, compactness therefore requires the size of the evaluated puzzle ciphertexts to be independent of the size of the circuit, and for the runtime of the evaluation algorithm to be independent of the hardness parameter $\mathcal{T}$.

3 A Fair and Sound Non-Simultaneous Rational Secret Sharing Scheme

We consider a RSS scheme with the reconstruction phase defined as in Definition 2. In our construction, the dealer runs the share phase, where they sample a value for the number of shares needed to reconstruct the secret, as well as splitting the secret into shares and further into sub-shares, similarly to the approach in [12]. Then, the dealer distributes a unique, ordered list of sub-shares to each player, alongside broadcasting public parameters.

The reconstruction phase works in rounds, with the n players in P performing the communication phase and processing phase in parallel. In the first round of the reconstruction phase, only the communication phase occurs. The processing phase does not start until the second round onwards. Each round (after the first) of the reconstruction phase works as follows. Players communicate (following the order of their given list) the sub-share corresponding to the round of Γ that they are in, one at a time. They must check at the end of the round that they have obtained sub-shares from all other players.

At the same time, players process the sub-shares received in the previous round, evaluating them over some function to obtain a share of the secret. After a certain number of rounds, as decided by the dealer, a sufficient number of shares will have been derived and players can use these shares to reconstruct the correct secret. The concept of rounds in RSS means that players gradually recover the secret, by reconstructing just one share per round, motivating all players to continue following the reconstruction phase.

More specifically, we let the dealer D be honest and non-interactive, only taking part in the share phase. Following [28], we assume that the dealer (information-theoretically) authenticates the shares distributed to players so that a player cannot send an incorrect share to another player, and the set of shares that a player sends to other players is unique. These assumptions translate to only one of two actions that a player can perform in each round: communicate (follow their strategy) or remain silent. We assume the players' strategies in the reconstruction phase are in a (computationally) strict Nash equilibrium in order to motivate them to follow the phase and not deviate.

In the share phase of our construction, D samples r, the revelation value. The revelation value signifies how many correctly run rounds, or equivalently, how many recovered shares are sufficient for a player to reconstruct the secret. D determines r by randomly sampling from an efficiently samplable discrete distribution $\mathcal{G}$, keeping the value secret from all players. Next, D obtains the first $(r + 1)$ shares of the secret s; where the 0th share s_0 will be the checking share, and is kept separate and broadcast to all players before the reconstruction phase. We note that the checking share is only used to verify the output of the reconstruction phase, and cannot be used to reconstruct the secret itself. This is necessary in order to ensure soundness of the output.

Additionally, the dealer randomly samples a value d from an efficiently samplable discrete distribution $\mathcal{G}'$ and generates d fake shares, used to disguise the value r. Typically both $\mathcal{G}$ and $\mathcal{G}'$ are geometric distributions [21,12], see Appendix C. Letting $m = r + d$, the dealer proceeds to create n sub-shares for each of the m shares, so that each player has a sub-share of every share, for a total of m sub-shares in each of the n lists, one for each player.

Similarly to [33], we need the sub-shares to be encrypted before being distributed to a player in a way that no player can decrypt their sub-shares before a round of communication is over. This is done so that players communicating non-simultaneously do not know until after they have broadcast their share for a given round, whether or not that was the revelation round. This is crucial to achieve fairness and ensure that players continue to be motivated to follow the scheme [24].

Our construction achieves this time delay using homomorphic time-lock puzzles (HTLPs), first introduced in [34] (see Definition 5). Using a HTLP scheme with hardness parameter $\mathcal{T}$, the dealer sets the time limit for each round of communication to be bounded above by time $\mathcal{T}$. Encrypting the sub-shares creates so-called sub-puzzles[7] of the sub-shares, which the dealer distributes as a list to individual players before reconstruction begins.

Each round of the reconstruction phase $\Gamma_{r,r+1}$ has players communicate non-simultaneously the corresponding sub-puzzle from their list, whilst processing in parallel the sub-puzzles received from the previous round. In a round of the

[7] We call the HTLP encryption of the sub-shares sub-puzzles for ease of understanding. They are simply time-lock puzzles that can be homomorphically evaluated to obtain a puzzle of the share which corresponds to the homomorphic evaluation of the given sub-shares.

communication phase, players must send their sub-share before time $\mathcal{T}$. Once this time has elapsed, a player checks that they have received $(n-1)$ sub-puzzles from the other players. In this case, in the next round of the reconstruction phase, these n sub-puzzles will be processed.

In the processing phase, players work independently and evaluate the n sub-puzzles from the previous round. In doing so, they will obtain a puzzle of the share for the previous round. This is computationally correct given that the sub-shares were derived by the dealer such that over some function the sub-shares homomorphically compute this share. The puzzle of the share is decrypted using the solve algorithm in the HTLP scheme to obtain the corresponding share.

Players attempt to reconstruct the actual secret from the shares that they have reconstructed so far. They determine whether they have reached the revelation round by using the checking share s_0 to confirm whether their solution is the real secret. If so, players output the secret s. If the reconstructed value as determined by the checking share, is $s' \neq s$, the players do not output a result. Instead, they will start the subsequent reconstruction phase round. Players repeat this cycle of steps until the have reconstructed the correct secret s, unless either of the following scenarios occur:

1. A deviant player has quit communicating in a round of the phase. Even if they correctly guess the right round to quit (round r), the time delay of the encrypted sub-puzzles ensures that the deviant player cannot decrypt the evaluated puzzle of the share before the end of a round.
   The non-deviant players quit communicating if at the end of the round they have received fewer than $(n-1)$ sub-puzzles. As a consequence, they cannot reconstruct a puzzle share for that round and will have an insufficient number of reconstructed shares, so the outcome for reconstruction will be $\perp$. The act of aborting means that no player learns the secret including the deviant player, as they are identified as a cheater before they can reconstruct the secret, if at all.
2. Players have sent the final, mth sub-puzzle from their list and so have no more sub-puzzles to share after this round. Players quit communication and attempt to reconstruct the secret from the shares that were reconstructed in the previous rounds.

3.1 Our Construction

Given an honest, non-interactive dealer D and a set of n rational players $P = \{P_1, \ldots, P_n\}$ communicating non-simultaneously, we use a HTLP to build a fair RSS scheme with a sound output. Assume that each round of the reconstruction phase is bounded by the time hardness parameter $\mathcal{T}$.

Definition 6 (Non-Simultaneous RSS Scheme).

Given security parameter λ, time hardness parameter $\mathcal{T}$, an efficiently samplable distribution of the set of secrets $\mathcal{S}_\lambda$ with operator Ψ, secret $s \in \mathcal{S}_\lambda$, efficiently

samplable discrete distributions $\mathcal{G}, \mathcal{G}'$, we construct a RSS scheme with reconstruction phase in the non-simultaneous setting as a tuple of three PPT algorithms $(\mathsf{Setup}', \mathsf{Share}', \mathsf{Recon}')$ from a secret sharing scheme $(\mathsf{Setup}, \mathsf{Share}, \mathsf{Recon})$ and a HTLP scheme
$(\mathsf{HP.Setup}, \mathsf{HP.Gen}, \mathsf{HP.Solve}, \mathsf{HP.Eval})$ as follows:

– **Sharing Phase:** The honest dealer D takes as input the secret $s \in \mathcal{S}_\lambda$ and performs the following steps non-interactively:
  1. $pp' \leftarrow \mathsf{Setup}'(1^\lambda, \mathcal{T})$ a probabilistic algorithm on inputs $1^\lambda, \mathcal{T}$ in which the dealer runs:
     (a) $pp_1 \leftarrow \mathsf{HP.Setup}(1^\lambda, \mathcal{T})$ which outputs public parameters pp_1.
     (b) $pp_2 \leftarrow \mathsf{Setup}(1^\lambda, \mathcal{T})$ which outputs the public parameters pp_2. Additionally let for $r \leftarrow_\$ \mathcal{G}$ be the sampled revelation value and d be a random value $d \leftarrow_\$ \mathcal{G}'$.

     Outputs are sampled values r, d and public parameters $pp' := \{pp_1, pp_2\}$.
  2. $\{s_0, \{list_1, \ldots, list_n\}\} \leftarrow \mathsf{Share}'(pp', s)$: a probabilistic algorithm that takes as input the secret $s \in \mathcal{S}_\lambda$ and public parameters pp'. The output consists of a checking share s_0 and lists labelled $list_j$ for $j \in [n]$, each composed of m sub-puzzles for $m = r + d$.
     (a) Run $\{s_0, \{s_1, \ldots, s_r\}\} \leftarrow \mathsf{Share}(pp_2, s)$ a probabilistic algorithm with inputs the public parameters pp_2 and secret $s \in \mathcal{S}_\lambda$. The outputs are $(r+1)$ shares of the secret; the checking share s_0 and s_i for $i \in [r]$.
     (b) $\{s_{r+1}, \ldots, s_m\} \leftarrow_\$ \mathcal{S}_\lambda$, randomly sample d fake shares from $\mathcal{S}_\lambda$.
     (c) For every $i \in [m]$, compute the list of sub-shares $\{s_{i,1}, \ldots, s_{i,n}\}$ such that $s_i = \underset{j \in [n]}{\Psi} s_{i,j}$.
     (d) Run $\mathcal{Z}_{i,j} \leftarrow \mathsf{HP.Gen}(pp_1, s_{i,j})$ a probabilistic algorithm that takes as input sub-shares $s_{i,j}$ and public parameters pp_1, and outputs sub-puzzles $\mathcal{Z}_{i,j}$, $\forall i \in [m], \forall j \in [n]$.
     (e) D distributes $\text{list}_j = \{\mathcal{Z}_{1,j}, \cdots, \mathcal{Z}_{r,j}, \mathcal{Z}_{r+1,j}, \cdots, \mathcal{Z}_{m,j}\}$ to the corresponding player P_j, for every $j \in [n]$.
  3. The dealer distributes the following:
     (a) D broadcasts $\{pp', s_0\}$ to all P the public parameters pp' and checking share s_0.
     (b) D distributes list_j to P_j for every $j \in [n]$.

– **Reconstruction Phase:** All players in $P = \{P_1, \ldots, P_n\}$ independently take part in this phase.

  1. Communication: We are in the kth round of the communication, for some $1 < k \leq m$.
     (a) P_j sends to all of P the sub-puzzle $\mathcal{Z}_{k,j}$ for every $j \in [n]$ non-simultaneously.
     (b) At the end of round k (after time $\mathcal{T}$ has elapsed), along with their own sub-puzzle, player P_j should have received $\{\mathcal{Z}_{k,1}, \ldots, \mathcal{Z}_{k,n}\}$ from all of P.

   (c) Move to round $(k+1)$ of communication and round k of processing, unless fewer than $(n-1)$ sub-puzzles have been received. In this case, proceed to abort communication and move to 2c with reconstructed shares $\{s_1, \ldots, s_{k-1}\}$.
2. Processing: We are in round $(k-1)$ of processing, for some $1 < k \leq m$.[8] For any $j \in [n]$, P_j does the following:
   (a) $\mathcal{Z}_{k-1} \leftarrow \mathsf{HP.Eval}(pp_1, \mathcal{T}, \Psi, \mathcal{Z}_{k-1,1}, \cdots, \mathcal{Z}_{k-1,n})$: Run the probabilistic algorithm $\mathsf{HP.Eval}$ with inputs the public parameters pp_1, hardness parameter $\mathcal{T}$, and the list of n sub-puzzles for the $(k-1)$th round, a player homomorphically evaluates sub-puzzles with operator Ψ to output share puzzle $\mathcal{Z}_{k-1}$.
   (b) $s_{k-1} \leftarrow \mathsf{HP.Solve}(pp_1, \mathcal{T}, \mathcal{Z}_{k-1})$: Run the probabilistic algorithm $\mathsf{HP.Solve}$ that takes as input the public parameters pp_1; hardness parameter $\mathcal{T}$; and puzzle share $\mathcal{Z}_{k-1}$ and outputs secret share s_{k-1}. Output the round share s_{k-1} and move to reconstructing s.
   (c) $\{s, \perp\} \leftarrow \mathsf{Recon}'(pp', s_0, \{s_1, \ldots, s_{k-1}\})$: where the players run $\{s, \perp\} \leftarrow \mathsf{Recon}(pp_2, \{s_1, \ldots, s_m\})$, a deterministic algorithm that inputs public parameters pp_2 and $(k-1)$ reconstructed shares of the secret $\{s_1, \ldots, s_{k-1}\}$. Player P_j uses checking share s_0 to confirm the soundness of their reconstructed value and outputs either the correct secret s or abort $\perp$.
   (d) If P_j outputs $\perp$, but no player quit in round k of communication and every player $P_j \in P$ has $list_j \neq \emptyset$, then players go to $(k+1)$th round of reconstruction phase. If either case holds, output $\perp$.

In Theorem 1 (Appendix F) we prove that our construction satisfies correctness, achieves soundness in the non-simultaneous setting using protocol-induced side information, and achieves fairness despite the presence of this side-information by using a HTLP to provide a time-delay to the scheme.

More specifically, in our security analysis (Appendix F), we summarise the scenarios in which a deviant player attempts to mislead. In particular, we demonstrate that if a player aborts in a round k with respect to revelation round r, regardless of the round that k is, the outcome for all players is the same. Analysing the scenarios in which a players quits communicating aids the proofs of fairness and correctness, by providing an intuition to the outcome of the reconstruction phase.

Fairness of the scheme is proven as follows: we show that Definition 3 is satisfied in our construction assuming the correctness and security of the HTLP scheme [34] (Appendix A.2), which is employed to implement a time-delay in the scheme. We use a reduction to break the correctness and security of the HTLP scheme, contradicting our assumptions, in order to show that there does not exist a deviant player with the ability to decrypt a puzzle in time less than $\mathcal{T}$. Furthermore, assuming the correctness and secrecy of the underlying SS scheme (Appendix A.3), we show that the probability of a deviant player learning the

[8] At least one round of communication is required before players can start processing.

secret, whilst other players do not, is negligible in the security parameter λ. Observe that we additionally show in Appendix D.2 that the rational players strategies $\overrightarrow{\sigma}$ are in a computationally strict Nash equilibrium (Definition 16) following the proofs of [12,33].

In order to prove soundness, we provide an Appendix E preceding the analysis of Theorem 1 to define the side information used to achieve soundness. We closely follow the proof of [12] by firstly defining a membership oracle. Informally, this is an oracle queried by players in reconstruction in order to check the soundness of their reconstructed value [33]. Following [12], we claim and prove that the checking share in our construction can be used in place of a sound membership oracle (Definition 18, Appendix E), as a specific form of protocol-induced side information to ensure soundness. Finally, we prove Theorem 2, that states our construction achieves soundness with a checking share.

We defer the reader to Appendix F for the full details of our proofs.

Next, we highlight the efficiency improvements our construction achieves by using a HTLP over standard TLPs. We then discuss how our results improve upon the scheme of [12], the most relevant related work.

HTLPs vs. TLPs The homomorphic property of a HTLP scheme means that solving a puzzle, the most computationally expensive step for the players, need only be run once rather than n times in the processing phase of our scheme. The computational cost of running HP.Solve is $\Omega(2^{\mathcal{T}})$-steps [9].

Indeed, if we were to use a standard TLP in the processing phase of our scheme, each player would independently have to solve each of the n sub-puzzles using P.Solve, and then evaluate the n sub-shares to obtain the share for that round. Conversely, by using a HTLP in our scheme, players must run HP.Eval *once* over the n sub-puzzles, outputting a master puzzle, and proceed to run HP.Solve *once* on this master puzzle to obtain the corresponding share. Thus, HTLPs are more efficient by a linear factor of n, where n corresponds to the number of players participating in the reconstruction phase.

It is important that the homomorphic property of the HTLP scheme satisfies Definition 9 of compactness [34]. This means that the runtime of homomorphically evaluating puzzles, is bounded above by a fixed polynomial that only depends on the security parameter λ and not the time hardness parameter $\mathcal{T}$. Otherwise, the trivial solution would be indeed to use a standard TLP scheme.

Comparison with [12] Our scheme closely follows the work of [12]. Their construction involves linearly evaluating sub-shares encrypted using memory-bound functions for the time-delay to ensure fairness of the scheme, and reconstructing the secret using Shamir's SS scheme. In contrast, our generic construction uses

[9] In a standard TLP scheme, the computational complexity of the puzzle solving algorithm P.Solve is the same as HP.Solve.

CPU-bound HTLPs to ensure a time-delay in rounds of the scheme, which we have argued in the Introduction constitutes an improvement.

Furthermore, the construction of [12] requires players to independently decrypt each share before they proceed to the secret reconstruction using Shamir SS scheme. The advantage of using HTLPs is that they provide an efficiency improvement for the honest players evaluating puzzles in comparison to using standard TLPs. Therefore our contributions are the efficiency improvements for honest players in homomorphically evaluating puzzles.

Finally, we have generalised our construction so that it can be adapted for different applications. The HTLP schemes of [34] are flexible in using different homomorphic operations, and can be extended to using puzzles with varying levels of hardness (different $\mathcal{T}$ values), with potential for public-coin setup schemes and reusable setup schemes. Unlike [12] who provide a concrete scheme, our construction is generic and adaptable to the application for which it is being used.

3.2 A Concrete Instantiation

Our final contribution is to provide a concrete fair and sound RSS scheme by instantiating our construction with a specific variant of Shamir's SS scheme and a multiplicative-HTLP (MHTLP [34]). In more detail, we instantiate our construction as follows:

- A multiplicative homomorphic threshold secret sharing scheme $(\mathsf{Setup}, \mathsf{Share}, \mathsf{Recon})$ (Appendix B), for a secret space $\mathcal{S}_\lambda$ over a finite group with respect to multiplication, defined as in [43,15,13],
- A MHTLP scheme $(\mathsf{MHP.Setup}, \mathsf{MHP.Gen}, \mathsf{MHP.Eval}, \mathsf{MHP.Solve})$ (Appendix A.2), which is multiplicatively homomorphic over a ring $(\mathbb{J}_N, \cdot)$.

The multiplicative operator $\otimes$ enables the dealer to split the ith share, for some $i \in [m]$, of the secret into n sub-shares in the following way,

$$s_{i,n} = s_i \cdot \left(\prod_{j=1}^{n-1} s_{i,j} \right)^{-1},$$

enabling players to homomorphically evaluate sub-puzzles by running $\mathsf{MHP.Eval}$, and $\mathsf{MHP.Solve}$ on the master puzzle output from evaluation to obtain the correctly reconstructed share for the ith round. In addition to the assumptions used in the security analysis of our generic construction, the instantiation relies on standard cryptographic and number theoretical assumptions, including the sequential squaring and decisional Diffie- Hellman assumptions for a MHTLP [34], found in Appendix A.2. We defer to Appendix C for a full description of the instantiation of our construction.

Final remarks In this paper we have proposed a construction for a fair and sound rational secret sharing scheme in the non-simultaneous setting of communication from homomorphic time-lock puzzles. We have argued the benefits of this novel approach, and we have suggested a concrete scheme, relying on standard assumptions.

References

1. M. Abadi, M. Burrows, M. Manasse, and T. Wobber. Moderately hard, memory-bound functions. *ACM Transactions on Internet Technology*, 5:299–327, 2005.
2. G. Asharov. Towards characterizing complete fairness in secure two-party computation. In Y. Lindell, editor, *Lecture Notes in Computer Science*, volume 8349, pages 291–316. Theory of Cryptography Conference, TCC 2014, Springer, 2014.
3. G. Asharov and Y. Lindell. Utility dependence in correct and fair rational secret sharing. In S. Halevi, editor, *Lecture Notes in Computer Science*, volume 5677, pages 559–576. Annual International Cryptology Conference, CRYPTO 2009, Springer, 2009.
4. A. Beimel. *Secure schemes for secret sharing and key distribution.* PhD thesis, Technion-Israel Institute of technology, Faculty of computer science, 1996.
5. A. Beimel. Secret-sharing schemes: A survey. In Y.M. Chee et al., editors, *Lecture Notes in Computer Science*, volume 6639, pages 11–46. International Conference on Coding and Cryptology, Springer, 2011.
6. G. R. Blakley. Safeguarding cryptographic keys. In *Proceedings of the AFIPS National Computer Conference, NCC 1979*, volume 48, pages 313–318. International Workshop on Managing Requirements Knowledge (MARK), IEEE, 1979.
7. J. Cathalo, B. Libert, and J. Quisquater. Efficient and non-interactive timed-release encryption. In S. Qing, W. Mao, J. Lopez, and G. Wang, editors, *Lecture Notes in Computer Science*, volume 3783, pages 291–303. International Conference on Information and Communications Security, ICICS 2005, Springer, 2005.
8. D. Chaum, C. Crépeau, and I. Damgard. Multiparty unconditionally secure protocols. In C. Pomerance, editor, *Lecture Notes in Computer Science*, volume 293, pages 11–19. Advances in Cryptology- CRYPTO 1987, Springer, 1988.
9. R. Cleve. Limits on the security of coin flips when half the processors are faulty. In *Proceedings of the Eighteenth Annual ACM Symposium on Theory of Computing*, page 364–369. STOC 1986, Association for Computing Machinery, 1986.
10. G. Couteau, T. Peters, and D. Pointcheval. Encryption switching protocols. In M. Robshaw and J. Katz, editors, *Lecture Notes in Computer Science*, volume 9814, pages 308–338. Advances in Cryptology, CRYPTO 2016, Springer, 2016.
11. R. Cramer, I. Damgård, and U. Maurer. General secure multi-party computation from any linear secret-sharing scheme. In B. Preneel, editor, *Lecture Notes in Computer Science*, volume 1807, pages 316–334. Advances in Cryptology- EUROCRYPT 2000, Springer, 2000.
12. S. J. De and Asim K. Pal. Achieving correctness in fair rational secret sharing. In M. Abdalla, Cristina N. R., and R. Dahab, editors, *Lecture Notes in Computer Science*, volume 8257, pages 139–161. International Conference on Cryptology and Network Security, CANS 2013, Springer, 2013.
13. Y. Desmedt, G. Di Crescenzo, and M. Burmester. Multiplicative non-abelian sharing schemes and their application to threshold cryptography. In J. Pieprzyk and R. Safavi-Naini, editors, *Lecture Notes in Computer Science*, volume 917, pages 19–32. Advances in Cryptology, ASIACRYPT 1994, Springer, 1994.
14. Y. Desmedt and Y. Frankel. Shared generation of authenticators and signatures. In J. Feigenbaum, editor, *Lecture Notes in Computer Science*, volume 576, pages 457–469. Advances in Cryptology- CRYPTO 1991, Springer, 1991.

15. Y. G. Desmedt and Y. Frankel. Homomorphic zero-knowledge threshold schemes over any finite abelian group. *SIAM journal on Discrete Mathematics*, 7(4):667–679, 1994.
16. Y. Dodis and T. Rabin. Cryptography and game theory. *Algorithmic Game Theory*, pages 181–207, 2007.
17. S. Doshi, F. Monrose, and A. D. Rubin. Efficient memory bound puzzles using pattern databases. In J. Zhou, M. Yung, and F. Bao, editors, *Lecture Notes in Computer Science*, volume 3989, pages 98–113. International Conference on Applied Cryptography and Network Security, ANCS 2006, Springer, 2006.
18. C. Dwork, A. Goldberg, and M. Naor. On memory-bound functions for fighting spam. In D. Boneh, editor, *Lecture Notes in Computer Science*, volume 2729, pages 426–444. Advances in Cryptology, CRYPTO 2003, Springer, 2003.
19. G. Fuchsbauer, J. Katz, and D. Naccache. Efficient rational secret sharing in standard communication networks. In D. Micciancio, editor, *Lecture Notes in Computer Science*, volume 5978, pages 419–436. Theory of Cryptography Conference, TCC 2010, Springer, 2010.
20. C. Gentry, S. Halevi, and V. Vaikuntanathan. i-hop homomorphic encryption and rerandomizable yao circuits. In T. Rabin, editor, *Lecture Notes in Computer Science*, volume 6223, pages 155–172. Annual Cryptology Conference, CRYPTO 2010, Springer, 2010.
21. S. D. Gordon, C. Hazay, J. Katz, and Y. Lindell. Complete fairness in secure two-party computation. *Journal of the ACM (JACM)*, 58(6):1–37, 2011.
22. S. D. Gordon and J. Katz. Rational secret sharing, revisited. In *Lecture Notes in Computer Science*, volume 4116, pages 229–241. International Conference on Security and Cryptography for Networks, SCN 2006, Springer, 2006.
23. V. Goyal, O. Pandey, and B. Sahai, A.and Waters. Attribute-based encryption for fine-grained access control of encrypted data. In *Proceedings of the 13th ACM Conference on Computer and Communications Security*, page 89–98. CCS 2006, Association for Computing Machinery, 2006.
24. J. Halpern and V. Teague. Rational secret sharing and multiparty computation: Extended abstract. In *Proceedings of the Thirty-Sixth Annual ACM Symposium on Theory of Computing*, page 623–632. STOC 2004, Association for Computing Machinery, 2004.
25. L. Harn, C. Lin, and Y. Li. Fair secret reconstruction in (t, n) secret sharing. *Journal of Information Security and Applications*, 23:1–7, 2015.
26. S. Hohenberger and B. Waters. Synchronized aggregate signatures from the rsa assumption. In *Lecture Notes in Computer Science*, volume 10821, pages 197–229. Advances in Crytology, EUROCRYPT 2018, Springer, 2018.
27. J. Katz. Bridging game theory and cryptography: Recent results and future directions. In R. Canetti, editor, *Lecture Notes in Computer Science*, volume 4948, pages 251–272. Theory of Cryptography Conference, TCC 2008, Springer, 2008.
28. G. Kol and M. Naor. Cryptography and game theory: Designing protocols for exchanging information. In R. Canetti, editor, *Lecture Notes in Computer Science*, volume 4948, pages 320–339. Theory of Cryptography Conference, TCC 2008, Springer, 2008.
29. G. Kol and M. Naor. Games for exchanging information. In *Proceedings of the Fortieth Annual ACM Symposium on Theory of Computing, STOC 2008*, page 423–432. Association for Computing Machinery, 2008.
30. H. Krawczyk. Secret sharing made short. In *Lecture Notes in Computer Science*, volume 773, pages 136–146. Advances in Cryptology, CRYPTO 1993, Springer, 1993.

31. C-S Laih and Y-C Lee. V-fairness (t, n) secret sharing scheme. *IEE Proceedings-Computers and Digital Techniques*, 144(4):245–248, 1997.
32. H-Y Lin and L. Harn. Fair reconstruction of a secret. *Information Processing Letters*, 55(1):45–47, 1995.
33. A. Lysyanskaya and A. Segal. Rational secret sharing with side information in point-to-point networks via time-delayed encryption. *IACR Cryptology ePrint Archive*, 2010:540, 2010.
34. G. Malavolta and S. A. K. Thyagarajan. Homomorphic time-lock puzzles and applications. In A. Boldyreva and D. Micciancio, editors, *Lecture Notes in Computer Science*, volume 11692, pages 620–649. Annual International Cryptology Conference, CRYPTO 2019, Springer, 2019.
35. T. C. May. Time-release crypto. In *Manuscript*, 1993.
36. J. Nash. Non-cooperative games. *Annals of mathematics*, pages 286–295, 1951.
37. T. P. Pedersen. Non-interactive and information-theoretic secure verifiable secret sharing. In J. Feigenbaum, editor, *Lecture Notes in Computer Science*, volume 576, pages 129–140. Advances in Cryptology, CRYPTO 1991, Springer, 1991.
38. R. L. Rivest, A. Shamir, and D. A. Wagner. Time-lock puzzles and timed-release crypto. *Technical Report MIT/LCS/TR-684*, 1996.
39. D. Rosenthal. On the cost distribution of a memory bound function. *arXiv preprint cs/0311005*, 2003.
40. A. Shamir. How to share a secret. *Communications of the ACM*, 22(11):612–613, 1979.
41. Y. Tian, J. Ma, C. Peng, and J. Zhu. Secret sharing scheme with fairness. In *10th International Conference on Trust, Security and Privacy in Computing and Communications*, pages 494–500. IEEE, 2011.
42. M. Tompa and H. Woll. How to share a secret with cheaters. *Journal of Cryptology*, 1(3):133–138, 1989.
43. H. Wang, K.Y. Lam, G.Z Xiao, and H. Zhao. On multiplicative secret sharing schemes. In E.P. Dawson, A. Clark, and C. Boyd, editors, *Lecture Notes on Computer Science*, volume 1841, pages 342–351. Information Security and Privacy, ACISP 2000, Springer, 2000.
44. B. Waters. Ciphertext-policy attribute-based encryption: An expressive, efficient, and provably secure realization. In D. Catalano, N. Fazio, R. Gennaro, and Nicolosi A., editors, *Lecture Notes in Computer Science*, volume 6571, pages 53–70. International Workshop on Public Key Cryptography- PKC 2011, Springer, 2011.

A Definitions and Assumptions

A.1 Number Theory

Let $N = p \cdot q$ be a composite, for primes p, q. Define $\mathbb{Z}_N^* = \{x \in \mathbb{Z}_N | gcd(x, N) = 1\}$ as the finite group of numbers modulo N, closed under the multiplication operation ($\otimes$). For prime p, the Jacobian subgroup $\mathbb{J}_p \subseteq \mathbb{Z}_p^*$ is defined as; $\mathbb{J}_p = \{x \in \mathbb{Z}_p^* | \exists y \in \mathbb{Z}_p^* \text{ s.t } y^2 = x \pmod{p}\}$. The same holds for prime q, so that the Jacobian subgroup $\mathbb{J}_N \subseteq \mathbb{Z}_N^*$ is formed of elements x such that $\mathbb{J}_N(x) = \mathbb{J}_p(x) \cdot \mathbb{J}_q(x)$.

A.2 Homomorphic-TLPs

The following are formal definitions of correctness, security and compactness of a HTLP scheme from section (2.3), [34]. The following definition considers the case when the evaluation algorithm is executed only once (it can be extended, as in [20]):

Definition 7 (Correctness [34]). *Define* ($\mathsf{HP.Setup}$, $\mathsf{HP.Gen}$, $\mathsf{HP.Solve}$, $\mathsf{HP.Eval}$) *as a correct HTLP scheme for the class family C where $C = \{\mathcal{C}_\lambda\}_{\lambda\in\mathbb{N}}$ is defined as a class of circuits, if $\forall\lambda \in \mathbb{N}$, all polynomials $\mathcal{T}$ in λ, all circuits $C \in \mathcal{C}_\lambda$ and respective inputs $(s_1, \ldots, s_n) \in \mathcal{S}^n$,all public parameters pp and for all puzzles $\mathcal{Z}_i$ in support of $\mathsf{HP.Gen}(pp, s_i)$, the following two conditions are satisfied:*

1. There exists a negligible function $\mu(\cdot)$ such that $Pr[\mathsf{HP.Solve}(pp, \mathsf{HP.Eval}(C, pp, \Psi, \mathcal{Z}_1, \ldots, \mathcal{Z}_n)) \neq C(s_1, \ldots, s_n)] \leq \mu(\cdot)$.
2. There exists a fixed polynomial $p(\cdot)$ such that $\mathsf{HP.Solve}(pp, \mathcal{Z})$ runtime is bounded by $p(1^\lambda, \mathcal{T})$, where $\mathcal{Z} \leftarrow \mathsf{HP.Eval}(C, pp, \Psi, \mathcal{Z}_1, \ldots, \mathcal{Z}_n)$.

Definition 8 (Security [34]). *A HTLP scheme* ($\mathsf{HP.Setup}$, $\mathsf{HP.Gen}$, $\mathsf{HP.Solve}$, $\mathsf{HP.Eval}$) *is secure with gap $\epsilon < 1$ if there exists a polynomial $\tilde{\mathcal{T}}(\cdot)$ such that for all polynomials $\mathcal{T}(\cdot) \geq \tilde{\mathcal{T}}(\cdot)$ and every polynomial-size adversary $(\mathcal{A}_1, \mathcal{A}_2) = \{(\mathcal{A}_1, \mathcal{A}_2)_\lambda\}_{\lambda\in\mathbb{N}}$ where the depth of $\mathcal{A}_2$ is bounded from above by $\mathcal{T}^\epsilon(\lambda)$, there exists a negligible function $\mu(\cdot)$, such that for all $\lambda \in \mathbb{N}$ it holds that*

$$Pr\left[b \leftarrow \mathcal{A}_2(pp, \mathcal{Z}, \tau) : \begin{array}{l} (\tau, s_0, s_1) \leftarrow \mathcal{A}_1(1^\lambda), \\ pp \leftarrow \mathsf{HP.Setup}(1^\lambda, \mathcal{T}(\lambda)), \\ b \leftarrow_\$ \{0,1\}, \\ \mathcal{Z} \leftarrow \mathsf{HP.Gen}(pp, s_b) \end{array}\right] \leq \frac{1}{2} + \mu(\lambda)$$

and $(s_0, s_1) \in \mathcal{S}^2$.

Definition 9 (Compactness). *Define* ($\mathsf{HP.Setup}$, $\mathsf{HP.Gen}$, $\mathsf{HP.Solve}$, $\mathsf{HP.Eval}$) *as a compact HTLP scheme for the class family C where $C = \{\mathcal{C}_\lambda\}_{\lambda\in\mathbb{N}}$ is defined as a class of circuits, if $\forall\lambda \in \mathbb{N}$, all polynomials $\mathcal{T}$ in λ, all circuits $C \in \mathcal{C}_\lambda$ and respective inputs $(s_1, \ldots, s_n) \in \mathcal{S}^n$, all public parameters pp and for all puzzles $\mathcal{Z}_i$ in support of $\mathsf{HP.Gen}(pp, s_i)$, the following two conditions are satisfied:*

1. There exists a fixed polynomial $p(\cdot)$ such that $|\mathcal{Z}| = p(\lambda, |C(s_1, \ldots, s_n)|)$, where $\mathcal{Z} \leftarrow \mathsf{HP.Eval}(C, pp, \mathcal{Z}_1, \ldots \mathcal{Z}_n)$.
2. There exists a fixed polynomial $\tilde{p}(\cdot)$ such that the runtime $\mathsf{HP.Eval}(C, pp, \Psi, \mathcal{Z}_1, \ldots, \mathcal{Z}_n)$ is bounded by $\tilde{p}(1^\lambda, |C|)$.

Let g be a generator of the quadratic residues of $\mathbb{Z}_N^*$. The following assumptions are needed, depending on the operator Ψ, in proving the security of the HTLP schemes in [34].

Assumption 1 (Strong RSA Modulus [26]) *Let λ be the security parameter and N is the product of two λ-bit, distinct safe primes p, q, where $p = 2p' + 1$ and $q = 2q' + 1$. Let e be a randomly chosen prime such that $2^\lambda < e < 2^{\lambda+1} - 1$. Let $\mathbb{QR}_N$ be the group of quadratic residues in $\mathbb{Z}_N^*$ of order $p' \cdot q'$. Given (N, e) and a random $h \in \mathbb{QR}_N$, it is hard to compute x such that $x^e \equiv h \pmod N$.*

Assumption 2 (Sequential Squaring [34]) *Let N be a uniform strong RSA integer, g a generator of $\mathbb{J}_N$, and $\mathcal{T}(\cdot)$ be a polynomial. Then there exists some $0 < \epsilon < 1$ such that for every polynomial-size adversary $\mathcal{A} = \{\mathcal{A}_\lambda\}_{\lambda \in \mathbb{N}}$ who's depth is bounded from above by $\mathcal{T}^\epsilon(\lambda)$, there exists a negligible function $\mu(\cdot)$ such that*

$$Pr\left[b \leftarrow \mathcal{A}(N, g, \mathcal{T}(1^\lambda), x, y) : \begin{array}{c} x \leftarrow_\$ \mathbb{J}_N; b \leftarrow_\$ \{0,1\} \\ \text{if } b = 0 \text{ then } y \leftarrow_\$ \mathbb{J}_N \\ \text{if } b = 1 \text{ then } y := x^{2^{\mathcal{T}(1^\lambda)}} \end{array}\right] \le \frac{1}{2} + \mu(\lambda).$$

Note that the restriction of the domain of x, y to $\mathbb{J}_N$ to avoid trivial attacks where the distinguisher computes the Jacobi symbol of the group element.
In [10], they showed that the decisional Diffie-Hellman (DDH) assumption over $\mathbb{J}_N$ is implied by the DDH assumption over $\mathbb{Z}_p^*$ and $\mathbb{Z}_q^*$ and by the quadratic residue assumption over $\mathbb{Z}_N^*$.

Assumption 3 (Decisional Diffie-Hellman [34]) *Let N be a uniform strong RSA integer. Then for every polynomial-size adversary $\mathcal{A} = \{\mathcal{A}_\lambda\}_{\lambda \in \mathbb{N}}$ there exists a negligible function $\mu(\cdot)$ such that*

$$Pr\left[b \leftarrow \mathcal{A}(N, g, g^x, g^y, g^z) : \begin{array}{c} (x, y) \leftarrow_\$ \{1, \ldots, \phi(N)/2\}; b \leftarrow_\$ \{0,1\} \\ \textit{if } b = 0 \textit{ then } z \leftarrow_\$ \{1, \ldots, \phi(N)/2\} \\ \textit{if } b = 1 \textit{ then } z := x \cdot y \pmod{\phi(N)/2} \end{array}\right] \le \frac{1}{2} + \mu(\lambda)$$

A.3 Threshold SS Schemes

A (t, n) SS scheme, as given in Definition 2.1, needs to satisfy the properties of correctness and secrecy as follows,

Definition 10 (Correctness [19]). *A (t, n) secret sharing scheme defined as $(\mathsf{Setup}, \mathsf{Share}, \mathsf{Recon})$ is* correct *if $\forall \lambda \in \mathbb{N}$ and for all possible sets of n authorised players $P = \{P_1, \ldots, P_n\}$, given $\mathsf{Setup}(1^\lambda) \to pp$; for all secrets $s \in \mathcal{S}_\lambda$ and any subset of t' shares S' from $\mathsf{Share}(pp, s) \to \{s_1, \ldots, s_n\}$ communicated by players in P, there exists a negligible function $\mu(\cdot)$ such that*

$$\Pr[\mathsf{Recon}(pp, S') \neq s] \le \mu(\lambda).$$

Definition 11 (Secrecy [19]). *A (t, n) secret sharing scheme defined as $(\mathsf{Setup}, \mathsf{Share}, \mathsf{Recon})$ is* secret *if $\forall \lambda \in \mathbb{N}$ and for all possible sets of n authorised players $P = \{P_1, \ldots, P_n\}$, given $\mathsf{Setup}(1^\lambda) \to pp$; for all secrets $s \in \mathcal{S}_\lambda$ and any subset of $t' < t$ shares S' from $\mathsf{Share}(pp, s) \to \{s_1, \ldots, s_n\}$ communicated by players in P, there exists a negligible function $\mu(\cdot)$ such that*

$$\Pr[\mathsf{Recon}(pp, S') \neq \bot] \le \mu(\lambda).$$

B Homomorphic Multiplicative Threshold Secret Sharing

Let $P = \{P_1, \dots, P_n\}$ be a group of n players, D be the honest dealer, $\mathcal{S}_\lambda$ the set of secrets under security parameter λ. Assume the share for P_i, $i \in [n]$ is selected from the set $\mathcal{S}_i$[10]. A (t, n) threshold scheme consists if D running setup and sharing algorithms and taking a secret input from $\mathcal{S}_\lambda$, mapping $\mathcal{S}_\lambda \rightarrow \mathcal{S}_1 \times \dots \times \mathcal{S}_n$ to assign shares to every players in P of the form (x_i, s_i) for a $(t-1)$-degree polynomial,

$$f(x) = s + a_1 x + a_2 x^2 + \dots + a_{t-1} x^{t-1}$$

such that $\{a_1, \dots, a_{t-1}\} \leftarrow_\$ \mathcal{S}_\lambda$ and $\{x_1, \dots, x_n\} \in \mathcal{S}_\lambda$are distinct inputs, with outputs$s_i \in \mathcal{S}_i$ corresponding to x_i, $\forall i \in [n]$.

The reconstruction algorithm sees a subset of the players, $A \subseteq P$ either return the secret or the phase of reconstruction fails.

In multiplicative homomorphic threshold secret sharing ([43]) over groups ([15,13]) the secret space S_λ is a finite group with respect to the operation $\otimes$, and for any t distinct players P_i for $i \in \mathcal{S}'$ such that $\mathcal{S}' \subseteq \mathcal{S}_\lambda$, $|\mathcal{S}'| = t$, $\mathcal{S}' = \{i_1, \dots, i_t\}$, there exists a family of function for all $l \in [t]$ such that;

$$f_{i_l, \mathcal{S}'} : \mathcal{S}_{i_l} \rightarrow \mathcal{S}$$

with $\{i_1, \dots, i_l\}$ publicly ordered with the following property:
For any secret $s \in \mathcal{S}_\lambda$ and shares $s_{i_1}, \dots, s_{i_t}$ that have been distributed to P_i, $i \in \mathcal{S}'$ by the dealer D on input s, the secret can be expressed as follows,

$$s = f_{i_1, \mathcal{S}'}(s_{i_1}) \otimes \dots \otimes f_{i_t, \mathcal{S}'}(s_{i_t})$$

such that, provided the share set $\mathcal{S}_i$ is a group, the function $f_{i, \mathcal{S}'} : \mathcal{S}_i \rightarrow \mathcal{S}_\lambda$ is a group homomorphism for $\forall i, \mathcal{S}'$; then $\forall l \in [t]$, define

$$f_{i_l, \mathcal{S}'}(s_{i_l}) = f(x_{i_l}) \prod_{j \in \mathcal{S}', j \neq i_l} \tfrac{-x_j}{(x_{i_l} - x_j)} \ .$$

assuming that $(x_{i_l} - x_j) \in \mathcal{S}_\lambda$ with a multiplicative inverse. In other words, a unit of $\mathcal{S}_\lambda$.

C A Concrete Instantiation

We provide a concrete example of our construction by instantiating it with a generalised version of Shamir's SS scheme (Appendix B). The instantiation uses a multiplicative homomorphic threshold secret sharing scheme ($\mathsf{Setup}, \mathsf{Share}, \mathsf{Recon}$), such that the secret space $\mathcal{S}_\lambda$ is a finite group with respect to multiplication, defined as in [13,15,43]; and the multiplicative-HTLP scheme defined in [34] as the tuple of algorithms ($\mathsf{MHP.Setup}, \mathsf{MHP.Gen}, \mathsf{MHP.Eval}, \mathsf{MHP.Solve}$).

So far, we have detailed a HTLP scheme (Definition 2.3) and provided our generic construction (Definition 6) with a homomorphic operation Ψ, to be used

[10] We drop the λ in the set of shares for simplicity of notation.

as a tool in realising our construction. For the concrete instantiation we will make use of a multiplicative HTLP (MHTLP) which is multiplicatively homomorphic over the ring $(\mathbb{J}_N, \cdot)$ using operator $\otimes$.

We consider an honest dealer D, n rational players $P = \{P_1, \dots, P_n\}$ and efficiently samplable distribution $\mathcal{S}_\lambda = \mathbb{J}_N$ (see Appendix A.1 for further details[11]) under security parameter λ for secret $s \in \mathbb{J}_N$. Here, we define $N = p \cdot q$ to be a strong RSA prime (See Appendix A.2, Assumption 1, [26]), time hardness parameter $\mathcal{T}$, and efficiently samplable discrete distributions $\mathcal{G}, \mathcal{G}'$ [12]. The multiplicative homomorphic SS scheme is an $(r, r+1)$ threshold scheme such that, for value $r \leftarrow_\$ \mathcal{G} = \{1, \dots, N^2\}$, given $r+1$ shares of s, a threshold of r shares is sufficient to reconstruct s. We implicitly assume that both the multiplicative homomorphic SS and HTLP scheme share parameters. In the share phase, the honest dealer D takes as input the secret $s \in \mathbb{J}_N$ and performs the following steps:

1. $pp' \leftarrow \mathsf{Setup}'(1^\lambda, \mathcal{T})$ on inputs $1^\lambda, \mathcal{T}$ in which the dealer runs:
   (a) $pp_1 \leftarrow \mathsf{MHP.Setup}(1^\lambda, \mathcal{T})$:
   Uniformly sample $\tilde{g} \leftarrow_\$ \mathbb{Z}_N^*$ and set $g := -\tilde{g}^2 (\bmod\ N)$ such that $g \in \mathbb{J}_N$, where g is the generator of $\mathbb{J}_N$. Compute $h := g^{2^{\mathcal{T}}}$.[13] Define $pp_1 := (\mathcal{T}, N, g, h)$.
   (b) $pp_2 \leftarrow \mathsf{Setup}(1^\lambda, \mathcal{T})$: Define $pp_2 := \{p, \{y_0, \dots, y_r\}\}$ with prime $p > \{s, n\}$ and with $\{y_0, \dots, y_r\} \leftarrow_\$ \mathbb{J}_N$.
   Additionally, let $r \leftarrow_\$ \mathcal{G}$ be the sampled revelation value r and $d \leftarrow_\$ \mathcal{G}'$ a random value, and output r, d and public parameters $pp' := \{pp_1, pp_2\}$.
2. $\{s_0, \{list_1, \dots, list_n\}\} \leftarrow \mathsf{Share}'(pp', s)$: takes as input the secret $s \in \mathbb{J}_N$ and public parameters pp'. The output consists of a checking share s_0 and lists labelled $list_j$ for $j \in [n]$, each composed of m sub-puzzles for $m = r + d$.
   (a) $\{s_0, \{s_1, \dots, s_r\}\} \leftarrow \mathsf{Share}(pp_2, s)$: inputs are the public parameters pp_2 and secret $s \in \mathbb{J}_N$. The dealer outputs $(r+1)$ shares s_i, determined as follows:
   D chooses a random r degree polynomial $f(x) = a_0 + a_1 x + a_2 x^2 + \dots + a_r x^r$ where $a_0 = s$ and $\{a_1, \dots, a_r\} \leftarrow_\$ \mathbb{J}_\mathbb{N}$. The dealer then determines shares s_i for $i \in \{0, \dots, r\}$, computed as $s_i = f(y_i)\ (\bmod\ N)$ such that $s_i \in \mathbb{J}_N$.
   (b) $\{s_{r+1}, \dots, s_m\} \leftarrow \mathbb{J}_N$, randomly sample d fake shares from secret space $\mathbb{J}_N$.
   (c) In our construction, shares $\{s_1, \dots, s_r\}$ are split into n sub-shares so that they can be distributed to P (along with other, fake sub-shares). Checking share s_0 is publicly broadcast and used to check soundness.

[11] $\mathbb{J}_N$ is the cyclic group of elements of $\mathbb{Z}_N^*$ with Jacobi symbol $+1$.

[12] We suggest geometric distributions over $\mathbb{N}$, with the parameter dependent on players outcome preferences (see Appendix D.2), denoted β. Let β be the probability of the first success in a repeated Bernoulli trial, that is, repeatedly tossing a biased coin until the first head appears [21].

[13] The dealer can optimise h by reducing the exponent modulo $\phi(N)/2$ first, as suggested in [34].

Distributing sub-shares to P enables just one share to be reconstructed per round, gradually releasing the secret to players in the reconstruction phase.

How are sub-shares created by the dealer during the setup phase?

Here, D creates n sub-shares for each of the m shares they have derived as follows: for a fixed $i \in [m]$, the dealer samples sub-puzzles $s_{i,j} \leftarrow_{\$} \mathbb{QR}_N$ for $j \in \{1, n-1\}$ and defines the nth sub-share to be

$$s_{i,n} = s_i \cdot \left(\prod_{j=1}^{n-1} s_{i,j} \right)^{-1}$$

for every $i \in [m]$. The ith share is defined as

$$s_i = \prod_{j \in [n]} s_{i,j} \pmod N. \; (*)$$

(d) Next, D generates sub-puzzles $\mathcal{Z}_{i,j}$ to embed the sub-shares as follows. It runs $\mathsf{MHP.Gen}(pp_1, s_{i,j})$ on input public parameters pp_1 and sub-shares $s_{i,j}$: $\forall i \in [m]$, $\forall j \in [n]$, uniformly sample $r_{i,j} \leftarrow_{\$} \{1, \cdots, N^2\}$, and generate the elements $u_{i,j} := g^{r_{i,j}} \pmod N$, $v_{i,j} := h^{r_{i,j}} \cdot s_{i,j} \pmod N$. Define the n sub-puzzles for the ith share to be $\mathcal{Z}_{i,j} := (u_{i,j}, v_{i,j}) \in \mathbb{J}_N^2$. The dealer outputs sub-puzzles $\mathcal{Z}_{i,j}$, $\forall i \in [m], \forall j \in [n]$.

(e) D distributes $\text{list}_j = \{\mathcal{Z}_{1,j}, \cdots, \mathcal{Z}_{r,j}, \mathcal{Z}_{r+1,j}, \cdots, \mathcal{Z}_{m,j}\}$ to the corresponding player P_j, for every $j \in [n]$.

3. The dealer distributes the following:
   (a) D broadcasts $\{pp', s_0\}$ to all P the public parameters pp' and checking share $s_0 = f(y_0) \pmod N$, determined in the second step above.
   (b) D distributes list_j to P_j for every $j \in [n]$.

In the reconstruction phase, whilst sending sub-puzzles for communication phase k, for some $1 < k \leq m$, players simultaneously process the set of n sub-puzzles that they obtained in the previous round. In the processing phase, assume that players are in round $(k-1)$ of processing. For any $j \in [n]$, P_j does the following:

1. $\mathcal{Z}_{k-1} \leftarrow \mathsf{MHP.Eval}(pp_1, \mathcal{T}, \otimes, \mathcal{Z}_{k-1,1}, \cdots, \mathcal{Z}_{k-1,n})$: input public parameters pp_1, hardness parameter $\mathcal{T}$, and the list of n sub-puzzles for the $(k-1)$th round. Compute $u_{k-1} := \Pi_{j=1}^{n} u_{k-1,j} \pmod N$, $v_{k-1} := \Pi_{j=1}^{n} v_{k-1,j} \pmod N$ and output the homomorphic puzzle $\mathcal{Z}_{k-1} := (u_{k-1}, v_{k-1})$ of share s_{k-1}.
2. $s_{k-1} \leftarrow \mathsf{MHP.Solve}(pp_1, \mathcal{T}, \mathcal{Z}_{k-1})$: on input the public parameters pp_1, hardness parameter $\mathcal{T}$, and puzzle share $\mathcal{Z}_{k-1}$, compute $w_{k-1} := u_{k-1}^{2^{\mathcal{T}}} \pmod{\text{N}}$ by sequential squaring. Output $s_{k-1} := v_{k-1}/w_{k-1}$ as the solution to the puzzle. Move to reconstructing s from shares $\{s_1, \ldots, s_{k-1}\}$.
   The dealer in a SS scheme will have created sub-shares using $(*)$ for the concrete instantiation of our construction using a MHTLP. As a consequence, following the correctness of the MHTLP construction [34], $s_{k-1} = \Pi_{j=n} s_{i,j}$, which is precisely the $(k-1)$th share.
3. $\{s, \perp\} \leftarrow \mathsf{Recon}'(pp', s_0, \{s_1, \ldots, s_{k-1}\})$: from the secret sharing scheme, run $\{s, \perp\} \leftarrow \mathsf{Recon}(pp_2, \{s_1, \ldots, s_{k-1}\})$ with inputs of the public parameters pp_2 and $(k-1)$ reconstructed shares of the secret $\{s_1, \ldots, s_{k-1}\}$.
   Outputting the secret is done firstly by players using polynomial interpolation (see Appendix B):

$$f'(x) = \prod_{i\in[k-1]} \left[f(y_i) \cdot \prod_{l\in[k-1], l\neq i} \tfrac{(x-x_l)}{(x_i-x_l)} \right] = a'_0 + a'_1 x + a'_2 x^2 + \ldots + a'_{k-1} x^{k-1}$$

using the shares $\{s_1, \ldots, s_{k-1}\}$ in such that $a'_0 = s'$.

4. P_j uses checking share s_0 to see if $f'(s_0) = f(s_0)$. If this equality holds, it must be the case that $(k-1) = r$ and so $s' = s$ is output. If $(k-1) \neq r$, the equality will not hold, which means that $s' \neq s$. The output may be $\perp$ depending on the following.
5. If P_j outputs $\perp$, but no player quit in round k of communication and every player $P_j \in P$ has $list_j \neq \emptyset$, then players go to $(k+1)$th round of reconstruction phase. However, if either case holds, output $\perp$ as the final outcome of the reconstruction game.

The concrete instantiation of our scheme relies on standard assumptions, as it is based on Shamir's SS and the MHTLP scheme from [34], both of which rely on standard cryptographic and number theoretic assumptions. More specifically, letting the modulus N be a strong RSA modulus; if the sequential squaring (Assumption 2 in Appendix A.2) and decisional Diffie-Hellman (Assumption 3 in Appendix A.2) assumptions hold over over $\mathbb{J}_N$, then the MHTLP scheme of [34] is proven secure.

In addition, the compactness property of the MHTLP scheme ensures, informally, that the length of evaluated puzzle $\mathcal{Z}_i$ of share s_i, only depends on the security parameter λ. As a consequence, in the context of our construction, it is more efficient for players to solve the evaluated puzzle $\mathcal{Z}_i$ than it is to solve individual sub-puzzles $\mathcal{Z}_{i,j}$ for a fixed $i \in [m]$, $\forall j \in [n]$, obtain sub-shares $s_{i,j}$ and using $(*)$ as a function to evaluate the share s_i. More succinctly, it is more efficient for the players to evaluate the function of embedded sub-shares than it is for them to evaluate the function of the decrypted sub-puzzles.

D Equilibrium Concepts

D.1 Utilities

Definition 12 (Utility Functions [12]).

The set U_i for each player P_i, is defined as the set of utility *values* resulting from the possible outcomes of the game, which are polynomial in the security parameter (λ) of the protocol. These values are determined by the outcomes $\overrightarrow{o}$ of the reconstruction phase, which depends on the strategies $\overrightarrow{\sigma}$ taken by the n rational parties.
The set consists of the following, $U_i = \{U_i^{TN}, U_i^{TT}, U_i^{NN}, U_i^{NT}, U_i^{FN}, U_i^{NF}\}$.
The parameters T, N,and F define the utility gained from the following actions of players (in which the players may be honest or dishonest):

- T signifies the case where a player learns the secret.
- N signifies the case where a player does not learn the secret.

– F signifies the case where a player learns a fake secret (mislead).

Therefore, the utility $U_i^{NF}$for P_i, is the value P_i gains from not learning the secret, whilst misleading the other $(n-1)$ parties into learning a fake (incorrect) secret. There are two preference relation scenarios which are of importance in defining our games for rational players, $\forall i \in [n]$:

1. $U_i^{TN} > U_i^{TT} > U_i^{NN} > U_i^{FN}$ and $U_i^{NF} \geq U_i^{TT}$;
2. $U_i^{TN} > U_i^{TT} > U_i^{NN} > U_i^{FN}$ and $U_i^{NF} < U_i^{TT}$.

Definition 13 (Utility Independence [3]).

Let $\tilde{U} \in U$ be a set for a specific utility function, consisting of all the corresponding utility values for every player in P. Define the set of polynomial utility functions $U' = \{U_i^{TN}, U_i^{TT}, U_i^{NN}, U_i^{NT}, U_i^{FN}, U_i^{NF}\}_{i=1}^n \setminus \tilde{U}_{i=1}^n$, as the set excluding all $\tilde{U}_{i=1}^n$ values. A mechanism $(\Gamma, \overrightarrow{\sigma})$ is said to be $\tilde{U}-$*utility independent* if for all polynomial utility functions $\tilde{U}_{i=1}^n$, the elements in $U = U' \cup \tilde{U}_{i=1}^n$ satisfies a certain preference relationship $\mathcal{R}$. Therefore, $(\Gamma, \overrightarrow{\sigma})$ is a is a fair reconstruction mechanism for preference relationship $\mathcal{R}$ among the elements of U.
Informally, rational players participating in a reconstruction protocol may obtain a positive utility from learning nothing and misleading others (U^{NF}) . FSS schemes provide the utility of U^{TT} to everyone, meaning that all parties learn the secret. Therefore, $(\Gamma, \overrightarrow{\sigma})$ is *sound* provided $U^{NF} < U^{TT}$. What happens in the case of $U^{NF} \geq U^{TT}$? That is, the first preference relation, as above.

In the non-simultaneous channel model, soundness no longer holds. Our construction will follow the ideas in scheme of [12] that is U^{NF}-independent despite the presence of protocol-induced auxiliary information. This is done assuming players are in a computationally strict Nash equilibrium state.

D.2 Nash Equilibrium

A variant of a Nash equilibrium is a formalisation of what it means for players in a secret sharing scheme to follow their strategies. Let $\Gamma_{r,r+1} = (\Gamma, \overrightarrow{\sigma})$ be the scheme that players in P are following, where $\sigma_i \in \overrightarrow{\sigma}$ is a strategy for player P_i, for some $i \in [n]$. We need to show that the players are in a computationally strict Nash equilibrium with side information (the checking share), assuming the utility of misleading is greater than the utility of honestly following the scheme. We prove this following the analysis in [12] in showing that players strategies $\overrightarrow{\sigma}$ are in a computationally strict Nash equilibrium with protocol induced side-information.

Definition 14 (Computationally Strict Nash Equilibrium [12]). *Given reconstruction phase $\Gamma_{r,r+1} = (\Gamma, \overrightarrow{\sigma})$, a strategy profile $\overrightarrow{\sigma}$ for Γ is said to be in a computationally strict Nash Equilibrium if for every $i \in [n]$ and every deviating strategy $\sigma_i' \neq \sigma_i$ that player P_i uses, it holds that $U_i(\sigma_i, \overrightarrow{\sigma_{-i}}) > U_i(\sigma_i', \overrightarrow{\sigma_{-i}})$.*

To rephrase, the utility (gain) of P_i following an alternative strategy σ'_i is less than the utility of P_i following their prescribed strategy σ_i from $\Gamma_{r,r+1}$, assuming all other players are following their prescribed strategy $\overrightarrow{\sigma_{-i}}$.

A rational player will adopt the strategy that results in them obtaining the highest utility value in the reconstruction phase. Definition 14 is used to ensure that players choose to follow their prescribed strategy, and thus ensure that fairness of the reconstruction scheme is achieved.

Definition 15 (Fair reconstruction Mechanism [3]). *Let $U = \{U_1, \ldots, U_n\}$ be the set of utility functions for players in P. The reconstruction phase $\Gamma_{r,r+1} = (\Gamma, \overrightarrow{\sigma})$ is fair for utility functions U if $\overrightarrow{\sigma}$ is a computational Nash equilibrium, and the probability that the outcome $\overrightarrow{\omega} = \perp$ for players in P when they follow $\overrightarrow{\sigma}$ is negligible.*

Furthermore, the notion of equivalent play was defined by the authors of [19] and augmented by [33] to incorporate a scheme where players have access to *any* side information related to the secret (or access to an oracle, see Appendix E). They defined $Aux = \{aux_i\}_{i\in[n]}$ to be the (protocol-induced) side information related to the secret that players in P have access to during $\Gamma_{r,r+1}$. When deviant player P_i takes alternative strategy σ'_i, then $\sigma'_i \approx_{Aux} \overrightarrow{\sigma}$ yields equivalent player if given the views of all other players $P_j \in P_{-i}$, including their side information aux_j for player P_j, no polynomial time algorithm can distinguish whether P_i is following prescribed strategy $\sigma_i \in \overrightarrow{\sigma}$ or alternative strategy σ'_i.

Definition 16 (Computationally Strict Nash Equilibrium with Protocol-Induced Side Information [12,33]). *Prescribed strategy $\overrightarrow{\sigma}$ in reconstruction phase $\Gamma_{r,r+1} = (\Gamma, \overrightarrow{\sigma})$ is a computational Nash equilibrium in the presence of protocol induced side-information $Aux = \{aux_i\}_{i\in[n]}$ if it is a Nash equilibrium with protocol-induced side information such that, given security parameter λ, for every $P_i \in P$ and for any PPT alternative strategy $\sigma'_i \not\approx_{Aux} \overrightarrow{\sigma}$, there exists a negligible function μ over the security parameter λ such that,*

$$U_i((\sigma'_i, \sigma_{-i}), Aux) < U_i(\overrightarrow{\sigma}, Aux) + \mu(\lambda).$$

We make the following assumptions about our construction in Definition 6:

- The discrete distributions $\mathcal{G}, \mathcal{G}'$ that the dealer randomly samples r and d from respectively, are set over the *utility parameter* β, which is determined by the utility functions of the players in P.
- We define the utility parameter β following [12], by setting $\beta < \frac{(U^{TT} - U^{NN})}{(U^{TN} - U^{NN})}$ as the parameter for discrete distributions $\mathcal{G}, \mathcal{G}'$ used in our construction.[14] We define U_i^{TN} as the utility function for P_i learning s, and P_{-i} learning nothing, U_i^{TT} as the utility function for all of P learning s and U_i^{NN} as the utility function for all of P outputting $\perp$.

[14] Under the assumption that some rational players may prefer to mislead other players, rather than follow their strategy, the utility parameter is defined in this way to ensure that the reconstruction phase is independent of this preference.

– Assume that some players may prefer to mislead others, denoting the corresponding utility function as U_i^{NF} for player P_i misleading players in P_{-i}.

Fairness. Similarly to [12], given the assumptions above we prove that our construction $\Gamma_{r,r+1}$ is fair by showing that the players strategies $\overrightarrow{\sigma}$ are a computational Nash equilibrium with protocol-induced auxiliary information, with utility functions for a player $P_i \in P$ defined as $U_i^{TN} > U_i^{TT} > U_i^{NN}$.

Proof. In the analysis of Theorem 1, we showed that if P_i deviates in round $(k+1) < r$, then all of P have outcome $\overrightarrow{\omega} = \perp$. If P_i quits in round $(k+1) = (r+1)$, then all of P have outcome $\overrightarrow{\omega} = s$. Given that P_i deviates in round $(k+1) = r$, let us denote $\Pr[\omega_i(\Gamma, (\sigma_i', \sigma_{-i}) = s] = \delta$ and $\Pr[\omega_i(\Gamma, (\sigma_i', \sigma_{-i}) = \perp] = (1-\delta)$. Given the utility parameter β, the probability of the secret being reconstructed with $(k+1) = r$ shares is $\delta = \beta(1-\beta)^{r-1}$. Therefore, we have the following inequality for P_i deviating in any round $(k+1)$:

$$\delta U_i^{TN} + (1-\delta)U_i^{NN} \leq \beta U_i^{TN} + (1-\beta)U_i^{NN} < U_i^{TT},$$

given that $r \geq 1$. Therefore, deviating with strategy σ_i' in round $(k+1)$ does not give P_i an advantage over others. Following the correctness of our construction $\Gamma_{r,r+1}$ (see Theorem 1), we additionally showed that the probability of players outcome $\overrightarrow{\omega} = \perp$ when they all follow $\overrightarrow{\sigma}$ is negligible. As a consequence, Definition 15 is satisfied. □

E Soundness with Checking Shares

Before proving soundness of our construction (Theorem 1), following the proofs of [12,33], we provide the necessary definitions for the proof. The authors of [12] follow the work of [33] by introducing protocol-induced (i.e. chosen by the protocol designer) auxiliary-information in the form of an extra share of the secret, known as the checking share.

The authors of [33] propose giving each player in P some *arbitrary* auxiliary information or access to a membership oracle $O_{k,j}^{s_0}$. Either of these enable a player $P_j \in P$ to confirm whether the value $s' \in \mathcal{S}_\lambda$ they reconstructed in the kth round of reconstruction is the correct secret or not. Intuitively, a membership oracle should not reveal any information about the secret itself, meaning that no player can learn anything important about s by simply observing the oracle or querying it with arbitrary inputs. The authors of [12] first define what a membership oracle is, in order to use *specific* protocol-induced side information to ensure soundness. We present the following definitions from [12].

Definition 17 (Membership Oracle [12]). *Given the secret s and reconstructed value s', both in secret space $\mathcal{S}_\lambda$ and checking share s_0, we define a membership oracle $O_{k,j}^{s_0} : \mathcal{S}_\lambda \rightarrow \{0,1\}$ queried by player $P_j \in P$ in the kth round of the reconstruction phase as follows:*

$$O_{k,j}^{s_0}(s') = \begin{cases} 1 \ \textit{if } s' = s, \\ 0 \ \textit{otherwise.} \end{cases}$$

Note that the behaviour of the oracle may be dependent on s. For example, it may output 1 on s and 0 on all other inputs, or it may output 1 on input s' if $f(s) = f(s')$ for some function f, where the function depends on the reconstruction process of the underlying SS scheme. Given this, in order to ensure soundness of the reconstruction phase with reconstructed value $s' \in \mathcal{S}_\lambda$, the oracle must always output the correct decision on whether this is actually the secret or not. Similarly to [12], we want to ensure soundness, therefore we must define a sound membership-oracle.

Definition 18 (Sound Membership Oracle[12]). *Given input $s' \in \mathcal{S}_\lambda$, a correct membership oracle $O^{s_0}_{k,j} : \mathcal{S}_\lambda \to \{0, 1\}$ for player $P_j \in P$ in the kth round of reconstruction with access to checking share s_0 has the following properties:*

1. *$Pr[O^{s_0}_{k,j}(s') = 1] \leq \mu(\lambda)$ for any $s' \neq s$;*
2. *$Pr[O^{s_0}_{k,j}(s') = 0] \leq \mu(\lambda)$ for $s' = s$.*

for a negligible function $\mu(\lambda)$, over security parameter λ.

Definition 19 (Protocol-Induced Membership Oracle [12]). *A sound membership oracle $O^{s_0}_{k,j}$ provided by scheme $(\mathsf{Setup}', \mathsf{Share}', \mathsf{Recon}')$, given to player P_j for $j \in [n]$ for the kth round of the reconstruction phase is called a protocol-induced membership oracle.*

For our construction, the sound membership oracle can be modelled as protocol induced auxiliary information in the form of a checking share, like in [12]. Assuming players do not have a preference to mislead, let $(\Gamma_f, \overrightarrow{\sigma_f})$ be a fair reconstruction phase where players follow their prescribed strategies of communicating, reconstruct some function f in order to reconstruct the secret s. Note that function f is the honestly constructed function created by the dealer during the share phase, dependent on the underlying SS scheme and its assumed security, in which players participating in the fair reconstruction phase $(\Gamma_f, \overrightarrow{\sigma_f})$ with a sufficient number of shares can reconstruct to obtain the secret.

In our construction, similar to the work of [12], we propose a fair reconstruction phase $(\Gamma_{f_k}, \overrightarrow{\sigma_{f_k}})$ for players $P_j \in P$ with k shares and access to protocol-induced auxiliary information in the form of a checking share s_0, assuming that some players may prefer to mislead other players into outputting an incorrect secret. Informally, this works as follows: Let player P_i deviate with strategy σ'_i in the $(k+1)$th round, and the non-deviant players P_{-i} are following strategies $\overrightarrow{\sigma_{f_k}}$ throughout the reconstruction phase. Strategy $\sigma_{f_k,j}$ for P_j instructs a player to follow their normal strategy $\sigma_{f,j}$ of reconstructing a value $s' \in \mathcal{S}_\lambda$ or outputting $\perp$ in the case that P_i quits communicating in the $(k + 1)$th round. If a value $s' \in \mathcal{S}_\lambda$ is reconstructed from reconstructed function f_k, strategy $\sigma_{f_k,j}$ instructs P_j to check the soundness of value s' obtained in the kth round, by using the checking share s_0. Let $s_0 = (y_0, f(y_0))$ for some input y_0 computed by the dealer in the share phase; if $f_k(y_0) = f(y_0)$, then P_j concludes that $s' = s$, otherwise P_j concludes that $s' \neq s$.

Following the claim made by the authors of [12], we claim and prove that the checking share in our construction can be used in place of the sound membership oracle of Definition 18, as protocol-induced auxiliary-information to ensure soundness.

Claim. Let $(\mathit{\Gamma}_{f_k}, \overrightarrow{\sigma_{f_k}})$ be the reconstruction mechanism $(\mathit{\Gamma}_f, \overrightarrow{\sigma_f})$ with $P_j \in P$ possessing k shares, checking share s_0, and reconstructed value $s' \in \mathcal{S}_\lambda$. Let $s_0 = (y_0, f(y_0))$ for some input y_0 computed by the dealer in the share phase; if $f_k(y_0) = f(y_0)$, then P_j concludes that $s' = s$, otherwise P_j concludes that $s' \neq s$.

Proof. By Definition 18, we can assume that the the following conditions hold:

1. $\Pr[f_k(y_0) = f(y_0)] \leq \mu(\lambda)$ for $s' \neq s$.
2. $\Pr[f_k(y_0) \neq f(y_0)] \leq \mu(\lambda)$ for $s' = s$.

Function f is honestly constructed by the dealer during the share phase such that the secret can be recovered from it. Due to the security of the underlying SS scheme, which we assume in Theorem 1, the function f must be one-to-one. This uniqueness property means that the function is unique and given some reconstructed value $s' \neq s$, the probability of f and and reconstructed function f_k being equal for a specific input is negligible. So the first inequality holds.

The second inequality holds by the correctness of our construction, proved in Theorem 1. This says that the players will always output s when they have reconstructed $s' = s$, except with negligible probability. Thus the checking share follows Definition 18, and can be used to provide soundness in the reconstruction phase $(\mathit{\Gamma}_{f_k}, \overrightarrow{\sigma_{f_k}})$. □

F Security Analysis of Our Construction

Theorem 1. *Our non-simultaneous rational secret scheme* (Setup', Share', Recon') *satisfies correctness, fairness and soundness in the presence of side information related to the secret, assuming the following properties:*

- *correctness, security, and compactness of the HTLP scheme,*
- *correctness and secrecy of the SS scheme,*
- *the checking share side information is correct, protocol-induced auxiliary information.*

Proof. We begin by proving the correctness of our construction, which follows from the correctness of the SS and HTLP schemes underlying it. For all possible players in P, $\forall \lambda, \mathcal{T} \in \mathbb{N}$ and for all possible secrets $s \in \mathcal{S}_\lambda$; given $\mathsf{Setup}'(1^\lambda, \mathcal{T}) \rightarrow pp'$ and $\mathsf{Share}'(pp', s) \rightarrow \{s_0, \{list_1, \ldots, list_n\}\}$ run as in our construction (Definition 6) using homomorphic operator Ψ, our scheme ($\mathsf{Setup}', \mathsf{Share}', \mathsf{Recon}'$) is correct for some negligible function $\mu(\cdot)$ over the security parameter λ. Namely, at round r of the reconstruction phase, players reconstruct s using the first r reconstructed shares. Correctness holds except with negligible probability for the following reasons:

- Following the correctness of ($\mathsf{HP.Setup}, \mathsf{HP.Gen}, \mathsf{HP.Eval}, \mathsf{HP.Solve}$), the HTLP scheme (recalled in Appendix 7) constructed using operator Ψ such that for $\forall i \in [r]$, and shares $s_i = \underset{j\in[n]}{\Psi} s_{i,j}$ there exists a negligible function $\mu'(\cdot)$ such that $\Pr[\mathsf{HP.Solve}(pp, \mathsf{HP.Eval}(pp_1, \mathcal{T}, \Psi, \mathcal{Z}_{i,1}, \ldots, \mathcal{Z}_{i,n})) \neq s_i] \leq \mu'(\lambda)$.
- Assuming shares $\{s_1, \ldots, s_r\}$ were constructed correctly as we just described, and following the assumed correctness of the underlying SS scheme we have, for some negligible function $\mu''(\cdot)$,
  $\Pr[\mathsf{Recon}'(pp', s_0, \{s_1, \ldots, s_r\}) \neq s] \leq \mu''(\lambda)$.

Next we analyse the fairness and soundness of our construction. We note that the outcome of the reconstruction phase $\Gamma_{r,r+1}$ is the same for every player in P irrespective of the round in which a deviant player quits in relation to the revelation round. To see this we provide an explanation of what happens when deviant player P_i quits and the resulting outcome. We will only look at a player who is the last to communicate in a round. If a player quits before every other player has sent their sub-share for that round, then they will not be able to reconstruct the corresponding share for the round that they quit in. In this case, the deviant player in no better a position than other players.[15]

Suppose P_i follows the reconstruction phase with their prescribed strategy σ_i (that is, to share the corresponding sub-puzzle) for the first k rounds, then follows the alternative strategy σ_i' (quit communicating) in the $(k+1)$th round.

The other players discounting P_i are labelled P_{-i}, and are assumed to be following the same strategy, prescribed by the dealer, of communicating. This is represented by the $(n-1)$ vector of strategies σ_{-i}. To restore fairness, P_{-i} must abort communication after the $(k+1)$th round has finished (bounded above by time $\mathcal{T}$) and they realise that P_i has not sent their sub-puzzle $\mathcal{Z}_{k+1,i}$, concluding that the previous round was the revelation round r. P_{-i} are not able to homomorphically evaluate $(k+1)$ sub-puzzles, since P_i did not communicate $\mathcal{Z}_{k+1,i}$, so they cannot reconstruct the $(k+1)$th share. As a consequence, P_{-i} move to reconstructing the secret from the k previously reconstructed shares $\{s_1, \ldots, s_k\}$ by running Recon'. The checking share s_0, essential to ensuring soundness, is used before P_{-i} outputs a result to confirm whether their outcome from reconstruction is the true secret.

As we have mentioned, the outcomes $\overrightarrow{\omega} = (\omega_1, \ldots, \omega_n)$ for every player depend on the round that P_i quits relative to revelation value r. We denote the outcome of deviant player P_i as ω_i and the outcome for the other $(n-1)$ non-deviant players P_{-i} as ω_{-i} in the following cases:

Case 1. $(k+1) < r$:
As P_i is last to communicate, they will be able to reconstruct the $(k+1)$th puzzle $\mathcal{Z}_{k+1}$ of share s_{k+1} using the sub-puzzles communicated by all other players in the $(k+1)$th round. Essential to ensuring fairness, intuitively, the time-delay is

[15] Notice that the non-deviant players will also be unable to reconstruct the share for that round.

ensured by the correctness of the HTLP (see Definition 7, Appendix A.2). Using the HTLP time-delay scheme in our construction prevents the deviant player P_i from solving $\mathcal{Z}_{k+1}$ (by running $\mathsf{MHP.Solve}$) before time $\mathcal{T}$ has elapsed and other players begin the reconstruction process. Thus, P_i has no time to reconstruct the secret before players P_{-i} begin reconstructing the secret. Indeed, by the property of compactness in the HTLP scheme, the players P_{-i} are able to evaluate the sub-puzzles to obtain the corresponding share before the end of the reconstruction phase round.

Furthermore, the total number of shares that P_i possesses remains insufficient ($< r$) to reconstruct the secret. As a consequence, P_i will be identified as a cheater when P_{-i} use the checking share to verify their reconstruction solution from shares $\{s_1, \dots, s_k\}$, as they too have an insufficient number of shares to reconstruct s. The output of the phase will be $\perp$. **Outcome for all players:** $\overrightarrow{\omega} = \perp$.

Note that P_i could attempt to reconstruct the secret with previously reconstructed shares plus the checking share s_0. Whilst this means that P_i potentially has r shares (as is the case if they quit when $k + 1 = r - 1$) and can reconstruct the correct secret, the value of r remains unknown and the deviant player no longer has the checking share to verify that they have reconstructed the correct secret. Even if P_i adopts this strategy, the best P_i can do is correctly guess the value of r. Additionally, players P_{-i} will reconstruct an incorrect value and identify P_i as a cheater.

Case 2. $(k + 1) = r$:
Whilst P_i actually has the correct number of shares to reconstruct s (they do not know this at the time), by the design of our construction there is an upper bound of time $\mathcal{T}$ for the length of a round in the phase, as in Case 1, so P_i cannot solve $\mathcal{Z}_{k+1}$ within this time limit of round $(k + 1)$ to reconstruct the share s_{k+1}. Furthermore, other players will have started reconstruction of the secret as follows.

When the $(k+1)$th round of communication has finished, P_{-i} will abort after not receiving sub-puzzle $\mathcal{Z}_{k+1,i}$ from P_i, moving to outputting the result from reconstruction of the secret with the previous $k = (r - 1)$ derived shares. As the other players have an insufficient number of shares to reconstruct s, the checking share will demonstrate that their output is incorrect, so $\omega_{-i} = \perp$, identifying P_i as a cheater in the process. P_i will not be able to reconstruct the secret s before players P_{-i} have output $\perp$. **Outcome for all players:** $\overrightarrow{\omega} = \perp$.

Case 3. $(k + 1) > r$:
The non-deviant players P_{-i} will have previously reconstructed k shares from the phase. If $k = r$, P_{-i} have the precise number of shares to reconstruct s and can verify their output using checking share s_0, that is $\omega_{-i} = s$. Despite P_i quitting in the $(k + 1)$th round, all players will be able to reconstruct the secret s when $k = r$. Our construction uses an $(r, r + 1)$ secret sharing scheme such that one of the $(r + 1)$ shares is a checking share that cannot be used to reconstruct the secret if soundness must be ensured. P_i can only deviate from their strategy up until the $(r+1)$th round of communication. **Outcome for all players:** $\overrightarrow{\omega} = s$.

Remark 1. Even if deviant players decide to send randomly generated sub-puzzles in the communication phase, the soundness of our scheme ensures that their deviance would be detected by honest players in the processing phase of reconstruction. This is because the honest players possess the checking share, which will confirm that they have reconstructed an incorrect secret. As a result, the output of reconstruction from such an attack would be $\perp$.

Remark 2. For simplicity, we consider the case of one deviating player, however we note that the construction tolerates up to $(r-1)$ deviant players co-operating.

Now that we have explained the various outcomes to the reconstruction phase, we show that fairness and soundness of our construction are ensured.

Fairness. Following Definition 3, we want to show that there exists some negligible function μ under security parameter λ such that

$$\Pr[\omega_i(\Gamma,(\sigma_i',\sigma_{-i}))=s] \leq \Pr[\omega_{-i}(\Gamma,(\sigma_i',\sigma_{-i}))=s]+\mu(\lambda) \qquad (1)$$

Let us note that there are two, mutually exclusive scenarios in which P_i learns s when deviating:

- When P_i takes strategy σ_i' in round $(k+1)=(r+1)$, as in Case 3. Define this scenario as $\texttt{Event}_1$, which has probability $\Pr[\omega_i(\Gamma,(\sigma_i',\sigma_{-i}))=s] = \Pr[\omega_{-i}(\Gamma,(\sigma_i',\sigma_{-i}))=s]$, *or*
- When P_i takes strategy σ_i' in round $(k+1)=r$, and P_{-i} have an insufficient number of shares to reconstruct s, by the secrecy of the underlying SS scheme. Define $\texttt{Event}_2$ to be the scenario in which P_i learns s but P_{-i} does not learn s.
  Firstly, the only way for P_i to learn s in this event is to evaluate the rth round sub-puzzles to obtain puzzle share $\mathcal{Z}_r$, solve $\mathcal{Z}_r$ to obtain share s_r and then reconstruct s from the r shares. The correctness of the HTLP states that the runtime for solving the puzzle is bounded by a fixed, positive polynomial $p(\lambda,\mathcal{T})$ and the security of the HTLP means that the solution of the puzzles is hidden for all players that run in (parallel) time $\mathcal{T}^{\epsilon}(\cdot)<\mathcal{T}(\cdot)$, for some $\epsilon<1$. The definition of security follows the standard cryptographic security notion of indistinguishable-CPA security. Suppose P_i is able to do this, meaning that there exists some algorithm able to solve a puzzle in time $\mathcal{T}^{\epsilon}(\lambda)$. We can use this in a reduction to break the correctness and security of the HTLP scheme, contradicting these assumptions in our construction. As a consequence, the probability that P_i solves $\mathcal{Z}_r$ to reconstruct the share s_r in time $\mathcal{T}^{\epsilon}$ is $\Pr[\mathsf{HP.PSolve}(pp_1,\mathcal{T}^{\epsilon},\mathcal{Z}_r)\rightarrow s_r] \leq 1/2+\mu(\lambda)$ for some negligible function μ over the security parameter of the HTLP. In addition, assuming the correctness of the underlying SS scheme, we have

$$\begin{aligned}&\Pr[\omega_i(\Gamma,(\sigma_i',\sigma_{-i}))=s]\\&=\Pr[(\mathsf{HP.PSolve}(pp_1,\mathcal{T}^{\epsilon},\mathcal{Z}_r)\rightarrow s_r)\cap(\mathsf{Recon}(pp,\{s_1,\ldots,s_r\})\neq\perp)]\\&\leq(1/2+\mu(\lambda))\cdot\mu'(\lambda)\leq\mu''(\lambda),\end{aligned}$$

for some negligible function μ''.
Secondly, assuming the secrecy of the SS scheme, for players P_{-i} with $k < r$ shares, the probability $\Pr[\mathsf{Recon}(pp, \{s_1, \dots, s_k\}) \neq \bot] \leq \mu'(\lambda)$. That is, $\Pr[\mathsf{Recon}(pp, \{s_1, \dots, s_k\}) = \bot] \in [1 - \mu'(\lambda), 1]$, and so
$\Pr[\omega_{-i}(\Gamma, (\sigma'_i, \sigma_{-i})) = \bot] \in [1 - \mu'(\lambda), 1]$. Given this fact,

$$\begin{aligned}\Pr[\mathtt{Event}_2] &= \Pr[(\omega_i(\Gamma, (\sigma'_i, \sigma_{-i})) = s) \cap (\omega_{-i}(\Gamma, (\sigma'_i, \sigma_{-i})) = \bot)] \\ &\leq \mu''(\lambda) \cdot 1 = \mu''(\lambda).\end{aligned}$$

Given the two disjoint scenarios in which P_i can reconstruct and learn s, we have;

$$\begin{aligned}\Pr[\omega_i(\Gamma, (\sigma'_i, \sigma_{-i})) = s] =& \Pr[\mathtt{Event}_1 \cup \mathtt{Event}_2] = \Pr[\mathtt{Event}_1] + \Pr[\mathtt{Event}_2] \\ &= \Pr[\omega_{-i}(\Gamma, (\sigma'_i, \sigma_{-i})) = s] + \Pr[\mathtt{Event}_2] \\ &\leq \Pr[\omega_{-i}(\Gamma, (\sigma'_i, \sigma_{-i})) = s] + \mu''(\lambda).\end{aligned}$$

Therefore Equation 1 is satisfied. Modelling the players in secret reconstruction as rational calls for their strategies to satisfy a form of equilibrium that motivates them to follow their strategy, ensuring fairness. We demonstrated in Appendix D.2 we prove that players strategies $\overrightarrow{\sigma}$ are in a computationally strict Nash equilibrium despite the presence of the checking share, satisfying Definition 16. Our proof of fairness follows the proofs of [12,33].

Soundness. Following Definition 4, we want to show that there exists some negligible function μ under security parameter λ such that

$$\Pr[\omega_{-i}(\Gamma, (\sigma'_i, \sigma_{-i})) \notin \{s, \bot\}] \leq \mu(\lambda) \tag{2}$$

Theorem 2 (Soundness with a Checking Share). *Let $\Gamma_{r,r+1} = (\Gamma_{f_k}, \overrightarrow{\sigma_{f_k}})$ be the (r, r+1) fair secret reconstruction phase of our construction (Definition 6), assuming player $P_j \in P$ has k shares, access to protocol-induced auxiliary information in the form of a checking share $s_0 = (y_0, f(y_0))$, defined as above such that function f is determined by the dealer to reconstruct the secret. Then, the reconstruction phase is sound.*

Proof. By the assumptions of Theorem 1, the reconstruction phase of our construction is fair despite every player having access to protocol-induced auxiliary information in the form of a checking share s_0, and satisfies correctness.

Suppose that deviant player P_i follows an alternative strategy σ'_i which sees player P_i follow their normal strategy σ_i for the first k rounds, and then deviate in round $(k+1)$ by quitting communication.

Regardless of the round that P_i deviates in with respect to the value r, assuming players have access to the checking share of the (r, r+1) such that Claim in Appendix E holds, the fair reconstruction phase $(\Gamma_{f_k}, \overrightarrow{\sigma_{f_k}})$ for rational players satisfies soundness. More precisely, given an value $s' \in \mathcal{S}_\lambda$, suppose P_i either reconstructs $s' \notin \{s, \bot\}$ with r shares or $k < r$ shares:

$$\begin{aligned}
&\Pr[\omega_{-i}(\Gamma_{f_k}, (\sigma'_i, \sigma_{-i,f_k})) \notin \{s, \perp\}] = \Pr[(\omega_{-i}(\Gamma_{f_k}, (\sigma'_i, \sigma_{-i,f_k})) = s')] \\
&= \Pr[(\mathsf{Recon}'(pp', s_0, \{s_1, \ldots, s_r\}) = s') \cup (\mathsf{Recon}'(pp', s_0, \{s_1, \ldots, s_k\}) = s')] \\
&= \Pr[\mathsf{Recon}'(pp', s_0, \{s_1, \ldots, s_r\}) \neq s] + \Pr[\mathsf{Recon}'(pp', s_0, \{s_1, \ldots, s_k\}) = s'] \\
&\leq \mu(\lambda) + \Pr[\mathsf{Recon}'(pp', s_0, \{s_1, \ldots, s_k\}) = s'] = \mu(\lambda) + \Pr[f_k(y_0) = f(y_0)] \\
&\leq \mu(\lambda) + \mu'(\lambda) \leq \mu''(\lambda),
\end{aligned}$$

for some negligible function μ'', where negligible function μ' in the penultimate inequality comes from the fact that $\Pr[f_k(y_0) = f(y_0)] \leq \mu'(\lambda)$ when $s' \neq s$. Therefore Equation 2 is satisfied. □

G Related Work

Cryptographic Secret Sharing Schemes In order to achieve fairness, [32] proposed that in addition to obtaining a share of the secret, each party possesses a check vector which they can use to verify the validity of other parties' shares, and a certificate vector, which is used to prove the validity of their own share in the reconstruction phase. The dealer in the scheme chooses an indicator, which is a form of public information unrelated to the secret that must be reconstructed. In their scheme, the secret is hidden in a sequence of elements, such that the subsequent element of the sequence is the indicator and the rest of the elements are dummy secrets.

The authors of [25] continued the work of [32]. The scheme in [32] works under the assumption that all parties sharing are legitimate. In other words, their scheme only deals with inside adversaries. Whereas [25] protects the secret from inside adversaries as well as unauthorised parties (outside adversaries) who are not legitimate shareholders.

In [31], a V-fair (t, n) SS scheme is proposed, where given n-parties, they have an equal probability of obtaining the secret, even if $V < (t/2)$ parties are dishonest. This is achieved by the dealer dividing the secret into multiple sub-secrets with different threshold values, and generating shares for each of the sub-secrets.

The authors of [9] showed that complete fairness cannot be achieved in *general*, without an honest majority. Intuitively, complete fairness means that it is possible for an adversary to learn the output of secret reconstruction if, and only if, the honest parties learn the output too [2]. In the setting of secure two-party computation, if just one of the parties is dishonest, there is no longer an honest majority, and so it was believed that no *non-trivial* function could be computed with complete fairness. However, the work of [21] demonstrated the existence of *some* non-trivial functions, based on cryptographic assumptions, which can be computed with complete fairness in the two-party setting.

In [21], the reconstruction phase is based on rounds, such that parties input a share of the secret into *some* function in every round and the round in which the party learns the secret depends on the value of their input (in contrast to standard protocols). If one party aborts after learning the secret in a round, and the second party has not yet received the function output, then the second party assumes the first learned the secret in the round they aborted and reconstructs the function output for that round independently. The scheme of [41] extend the work in [21], achieving a more efficient scheme. At a high level, their reconstruction scheme provides complete fairness by hiding the secret in a sequence of secrets such that the validity of the shares can be verified and used to detect deviant parties.

RSS schemes The main idea of the scheme in [28] is that no information about the players' inputs into rounds is revealed until the round in which the secret is recovered, in which players are communicating non-simultaneously. They achieve this by introducing a new cryptographic tool called meaningful/meaningless encryption, where players are motivated to follow the scheme as they do not know whether a round of reconstruction is meaningful or not.

Following on from [28], [19] propose a scheme that does not require simultaneous broadcast channels or physical assumptions in both two-player and multi-player RSS scheme instantiations. The protocol follows a series of fake rounds, followed by a real round. That is, in the real round, every player learns the secret, and in the fake rounds no information about the secret is revealed. Players cannot know whether a round is real or fake. They identify the real round in the subsequent round, where they reconstruct public information in the form of a flag/indicator (akin to the schemes of [25,32] in the cryptographic model). Similarly, the authors of [29] use the same idea of players reconstructing an indicator.

In the non-simultaneous setting, reconstructing an indicator can cause unfairness in the scheme. The player who is last to communicate will be able to reconstruct the indicator before other players, and therefore know before others that the previous round reconstructed the secret. Therefore, reconstructing a publicly known value such as an indicator, is not suitable for non-simultaneous schemes.
How can players determine when the secret has been reconstructed without some form of indicator? If some players derive a greater payoff from misleading, they can abort communicating in order to trick other players in reconstructing an incorrect value. With no way to check if this is the secret, soundness is not ensured.

The work of [33] fairness can be achieved in a RSS scheme in the presence of *arbitrary* side information, something that had previously not been achieved. In [33], the authors use a time-delay encryption (TDE) scheme to restore the fairness of the scheme over a standard point-to-point network that does not require broadcasting, that is, has a loose form of synchronicity. Subsequently, [12] used *specific* protocol-induced side information to provide the first fair and sound RSS scheme in the non-simultaneous setting, additionally achieving independence from the utility of misleading. By proposing a scheme that is independent of this utility, [12] disprove one of the impossibility results proposed in [3].