Efficient Rational Secret Sharing in Standard Communication Models

Georg Fuchsbauer* Jonathan Katz† Eric Levieil*
David Naccache*

Abstract

We propose a new methodology for rational secret sharing leading to various instantiations that are simple and efficient in terms of computation, share size, and round complexity. Our protocols do not require physical assumptions or simultaneous channels, and can even be run over asynchronous, point-to-point networks.

Of additional interest, we propose new equilibrium notions for this setting (namely, computational versions of *strict Nash equilibrium* and *stability with respect to trembles*), show relations between them, and prove that our protocols satisfy them.

1 Introduction

The classical problem of *t-out-of-n secret sharing* [Sha79, Bla79] involves a dealer D who distributes shares of a secret s to a group of n players $P_1, \ldots, P_n$ so that (1) any group of t or more players can reconstruct the secret without further involvement of the dealer, yet (2) any group of fewer than t players cannot recover the secret. For example, in Shamir's scheme [Sha79] the secret s lies in a finite field $\mathbb{F}$, with $|\mathbb{F}| > n$. The dealer chooses a random polynomial $f(x)$ of degree at most $t-1$ with $f(0) = s$, and gives each player P_i the "share" $f(i)$. To reconstruct the secret s, any t players simply broadcast their shares and interpolate the polynomial. On the other hand, any set of fewer than t players has no information about s given their shares.

The implicit assumption in the original formulation of the problem is that each party is either honest or corrupt, and honest parties are all willing to cooperate when reconstruction of the secret is desired. Beginning with the work of Halpern and Teague [HT04], protocols for secret sharing and other cryptographic tasks have begun to be re-evaluated in a game-theoretic light (see [DR07, Kat08a] for an overview of work in this direction). In this context, parties are neither honest nor corrupt but are instead simply *rational* and are assumed (only) to act in their own self-interest.

Under natural assumptions regarding the utilities of the parties, standard secret-sharing schemes completely fail. For example, assume as in [HT04] that all players want to learn the secret above all else, but otherwise prefer that no other players learn the secret. (Later, we will treat the utilities of the players more precisely.) For a set of t parties to reconstruct the secret in Shamir's scheme, each party is supposed to broadcast their share. It is easy to see, however, that each player is better off withholding their share no matter what the other players do. Consider P_1: If fewer than $t-1$ other

*École Normale Supérieure, LIENS - CNRS - INRIA, Paris, France. Email: {georg.fuchsbauer, eric.levieil, david.naccache}@ens.fr

†University of Maryland, USA. Email: jkatz@cs.umd.edu. Work done while visiting ENS and IBM. Research supported by NSF CyberTrust grant #0830464 and NSF CAREER award #0447075.

players reveal their shares, then P_1 does not learn the secret regardless of whether P_1 reveals his share or not. If more than $t-1$ other players reveal their shares, then everyone learns the secret and P_1's actions again have no effect. On the other hand, if *exactly* $t-1$ other players reveal their shares, then P_1 learns the secret (using his share) but prevents other players from learning the secret by *not* publicly revealing his own share. The result is that if all players are rational then no one will broadcast their share and the secret will not be reconstructed.

A series of recent works [HT04, GK06, LT06, ADGH06, KN08a, KN08b, OPRV09, MS09] has focused on designing *rational* secret-sharing protocols immune to the above problem. Protocols for rational secret sharing also follow from the more general results of Lepinski et al. [LMPS04, LMS05a, IML05, ILM08]. Each of these works has some or all of the following disadvantages:

On-line dealer or trusted parties. Halpern and Teague [HT04] introduced a general approach to solving the problem that has been followed in most subsequent work. Their solution, however, requires the continual involvement of the dealer, even after the initial shares have been distributed. The solution proposed by Halpern and Teague also applies only when $t, n \geq 3$.

Recent work of [ILM08, MS09] requires the involvement of some (minimally trusted) external parties during the reconstruction phase.

Computational inefficiency. To eliminate the on-line dealer, several researchers [GK06, LT06, ADGH06, KN08a] have suggested solutions that rely on multiple invocations of protocols for generic secure multi-party computation. Because the function being computed by these protocols is complex, it is unclear whether computationally *efficient* protocols with suitable functionality can be designed. The solutions of [LMPS04, LMS05a, IML05, ILM08], though following a different high-level approach, also rely on generic secure multi-party computation.

Non-standard communication models. The solutions in [HT04, GK06, LT06, ADGH06] assume *simultaneous broadcast* which implies that parties must decide on what value (if any) to broadcast in a given round before observing the values broadcast by other parties. Kol and Naor [KN08a] show how to avoid simultaneous broadcast, at the cost of increasing the round complexity by a (multiplicative) factor linear in the size of the domain from which the secret is chosen; their approach thus has super-polynomial complexity for secrets of super-logarithmic length. Subsequent work by Kol and Naor [KN08b] shows how to avoid the assumption of simultaneous broadcast at the expense of increasing the round complexity by a (multiplicative) factor of t.

The solutions of [LMPS04, LMS05a, IML05] require *physical* assumptions such as secure envelopes and ballot boxes. Secure envelopes can be used to implement simultaneous broadcast (but not vice versa) and hence represent a strictly stronger class of assumptions.

Ong et al. [OPRV09] provide a solution without simultaneous broadcast, but under the assumption that sufficiently many parties are completely honest and never deviate from the protocol. Here, as in all other work mentioned above, we do not impose this assumption.

As far as we are aware, all prior schemes for $n > 2$ assume the existence of a broadcast channel (whether simultaneous or not).

1.1 Our Results

Our solutions do not suffer from any of the drawbacks mentioned above. (One of the schemes of Kol and Naor [KN08b] does not either; a more detailed comparison of our work with theirs is given in Section 1.2.1.) Our schemes follow the same high-level idea as in prior work, but implement this idea without resorting to generic secure multi-party computation. Our protocols are therefore

(arguably) simpler than previous solutions; more importantly, they are efficient in terms of round complexity, share size, and required computation.

Our protocols do not require simultaneous communication. Instead we can rely on synchronous (but *non*-simultaneous) point-to-point channels. To the best of our knowledge, all prior schemes for $n > 2$ assume broadcast (whether simultaneous or not); note that the obvious approach of simulating broadcast by running a broadcast protocol over a point-to-point network will not, in general, work in the rational setting. Moreover, we show that our protocol can be adapted to work even in *asynchronous* point-to-point networks. We thus answer a question that had been open since the work of Halpern and Teague [HT04].

As an independent contribution, we also introduce two new equilibrium notions and prove that our protocols satisfy them. (A discussion of game-theoretic equilibrium notions used in this and prior work is given in Section 2.2.) Specifically, we first introduce a *computational* version of strict Nash equilibrium. Using this equilibrium notion in the context of rational secret sharing was propounded by Kol and Naor [KN08b], but they used an *information-theoretic* notion of strict Nash and showed some inherent limitations of doing so. As in all of cryptography, we believe computational relaxations are meaningful and should be considered; this also allows us to circumvent the limitations that hold in the information-theoretic case.

Motivated by the suggestion in [Kat08a], we also formalize a notion of *stability with respect to trembles*. We then prove that this notion implies our definition of computational strict Nash.

An interesting feature of our definitions is that they effectively rule out "signalling" via subliminal channels in the protocol. In fact, at every point in our protocols *there is a* ***unique*** *legal message a party can send*. This prevents a party from outwardly appearing to follow the protocol while subliminally communicating (or trying to organize collusion) with other parties. Preventing subliminal communication was an explicit goal of some prior work (e.g., [IML05, LMS05a, ASV08]), which achieved it only by relying on non-standard communication models.

1.2 Overview of Our Approach

We follow the same high-level approach as in [HT04, GK06, LT06, ADGH06, KN08a, KN08b]. Our reconstruction protocol proceeds in a sequence of "fake" iterations followed by a single "real" iteration. Roughly speaking, these satisfy the following requirements:

- In the real iteration, everyone learns the secret (assuming everyone follows the protocol).
- In a fake iteration, no information about the secret is revealed.
- No party can tell, in advance, whether the next iteration will be real or fake.

The iteration number i^* of the real iteration is chosen according to a geometric distribution with parameter $\beta \in (0, 1)$ (where β depends on the players' utilities). To reconstruct the secret, parties run a sequence of iterations until the real iteration is identified, at which point all parties output the secret. If some party fails to follow the protocol, all parties abort. Intuitively, it is rational for P_i to follow the protocol as long as the expected gain of deviating, which is positive only if P_i aborts *exactly* in iteration i^*, is outweighed by the expected loss if P_i aborts before iteration i^*.

In most prior work [GK06, LT06, ADGH06, KN08a], a secure multi-party computation was performed in each iteration to determine whether the given iteration should be real or fake. Instead we use the following approach, described in the 2-out-of-2 case (we omit some technical details in order to focus on the main idea): The dealer D chooses i^* from the appropriate distribution *in advance*, at the time of sharing. The dealer then generates two key-pairs (vk_1, sk_1), (vk_2, sk_2) for a *verifiable random function* [MRV99] (VRF; see Section 2.4), where vk represents a verification key

and sk represents a secret key, and we denote by $\mathsf{VRF}_{sk}(x)$ the evaluation of the VRF on input x using secret key sk. The dealer gives the verification keys to both parties, gives sk_1 to P_1, and gives sk_2 to P_2. It also gives $s_1 = s \oplus \mathsf{VRF}_{sk_2}(i^*)$ to P_1, and $s_2 = s \oplus \mathsf{VRF}_{sk_1}(i^*)$ to P_2. Each iteration consists of one message from each party: in iteration i, party P_1 sends $\mathsf{VRF}_{sk_1}(i)$ while P_2 sends $\mathsf{VRF}_{sk_2}(i)$. Observe that a fake iteration reveals nothing about the secret, in a computational sense. Furthermore, neither party can identify the real iteration in advance. (The description above relies on VRFs. We show that, in fact, trapdoor permutations suffice.)

To complete the protocol, we need to provide a way for parties to identify the real iteration. Previous work allows parties to identify the real iteration *as soon as it occurs*. This suffices when simultaneous broadcast is available, since each party must decide on its message before it learns whether the current iteration is real. When simultaneous channels are not available, however, this approach does *not* work since it is vulnerable to an obvious rushing strategy. Kol and Naor [KN08a, KN08b] show two different ways to avoid simultaneous broadcast, but the first applies only for secrets from polynomial-size domains (and yields round complexity linear in the domain size), while the second yields round complexity linear in t.

Motivated by recent work on fairness (in the malicious setting) [GHKL08, GK08], we suggest the following, new approach: delay the signal indicating whether a given iteration is real or fake until the *following* iteration. As before, a party cannot risk aborting until it is sure that the real iteration has occurred; the difference is that now, by the time it is sure of this fact, the real iteration is over and all parties are able to reconstruct the secret. This eliminates the need for simultaneous channels, while increasing the (expected) round complexity by only a single round. This approach can be adapted to the case of t-out-of-n secret sharing for any $t \leq n$ and works even when the parties communicate over a asynchronous, point-to-point network.

1.2.1 Comparison to the Kol-Naor Scheme

The only prior rational secret-sharing scheme that avoids an on-line dealer, is computationally efficient, and does not require simultaneous broadcast or physical assumptions is that of Kol and Naor [KN08b]. They also use the strict Nash solution concept (though their protocol without simultaneous channels is only shown to be ϵ-Nash) and so their work provides an especially good point of comparison with ours. Our protocols have the following advantages with respect to theirs:

Share size. In the Kol-Naor scheme, the shares of the parties have *unbounded* length. While not a significant problem in its own right, this is problematic when rational secret sharing is used as a sub-routine for rational computation of general functions. (See [KN08a].) Moreover, the *expected* length of the parties' shares in the Kol-Naor scheme is large: in the 2-out-of-2 case, shares of a secret s have expected size $O(\beta^{-1} \cdot (|s| + k))$ in their scheme; shares in our scheme have size $|s| + O(k)$.

Round complexity. The version of the Kol-Naor scheme that does not rely on simultaneous broadcast [KN08b, Section 6] has expected round complexity $O(\beta^{-1} \cdot t)$, whereas our protocol has expected round complexity $O(\beta^{-1})$. (The value of β is roughly the same in both cases.)

Resistance to coalitions. For the case of t-out-of-n secret sharing, the Kol-Naor scheme is susceptible to coalitions of two or more players. We show t-out-of-n secret-sharing protocols resilient to coalitions of up to $(t-1)$ parties; see Section 4 for further details.

Avoiding broadcast. The Kol-Naor scheme for $n > 2$ assumes synchronous broadcast, whereas our protocols work even if parties communicate over an asynchronous, point-to-point network.

2 Model and Definitions

We denote the security parameter by k. Let $\epsilon : \mathbb{N} \to \mathbb{R}$ be a function which may take negative values. We say ϵ is *negligible* if for all $c > 0$ there is a $k_c > 0$ such that $\epsilon(k) < 1/k^c$ for all $k > k_c$, and let negl denote a generic negligible function. We say ϵ is *noticeable* if there exist c, k_c such that $\epsilon(k) > 1/k^c$ for all $k > k_c$. Note that it is possible for ϵ to be neither negligible nor noticeable.

2.1 Secret Sharing and Players' Utilities

We define our model and then describe the game-theoretic concepts used. Even readers familiar with prior work in this area should skim these sections, since we formalize certain aspects of the problem slightly differently from prior work, and define new equilibrium notions.

A *t-out-of-n secret-sharing scheme for domain $\mathcal{S}$* (with $|\mathcal{S}| > 1$) is a two-phase protocol carried out by a dealer D and a set of n parties $P_1, \ldots, P_n$. In the first phase (the *sharing phase*), the dealer chooses a secret $s \in \mathcal{S}$. Based on this secret and a security parameter 1^k, the dealer generates shares $s_1, \ldots, s_n$ and gives s_i to player P_i. In the second phase (the *reconstruction phase*), some set of $t^* \geq t$ parties jointly reconstruct s. We impose the following requirements:

Secrecy: The shares of any $t-1$ parties reveal nothing about s, in a computational sense. Formally, for any $s_0, s_1 \in \mathcal{S}$ and any $i_1, \ldots, i_{t-1}$ the following are computationally indistinguishable:

$$\left\{(s_1, \ldots, s_n) \leftarrow D(1^k, s_0) : (s_{i_1}, \ldots, s_{i_{t-1}})\right\} \quad \text{and} \quad \left\{(s_1, \ldots, s_n) \leftarrow D(1^k, s_1) : (s_{i_1}, \ldots, s_{i_{t-1}})\right\}.$$

Correctness: For any set of $t^* \geq t$ parties who run the reconstruction phase honestly, the correct secret s will be reconstructed, except possibly with probability negligible in k.

The above definition views parties as either malicious or honest. To model parties as *rational*, we first define players' utilities. Given some set of $t^* \geq t$ parties active during the reconstruction phase, let the outcome o of the reconstruction phase be a binary vector of length t^* with $o_i = 1$ iff the output of P_i is equal to the initial secret s (i.e., P_i "learned the secret"). In contrast to prior work, we consider a party to have learned the secret s if and only if it outputs s, and do not care whether that party "really knows" the secret or not. In particular, in our formulation a party who outputs a random value in $\mathcal{S}$ without running the reconstruction phase at all "learns" the secret with probability $1/|\mathcal{S}|$. We model the problem this way for at least two reasons:

1. Our formulation lets us model a player learning *partial* information about the secret, something not reflected in prior work. In particular, partial information that increases the probability with which a party outputs the correct secret increases that party's expected utility.
2. It is difficult, in general, to formally model what it means for a party to "really" learn the secret, especially when considering arbitrary protocols and behaviors. In contrast, in our definition it is easy to tell whether a player learns the secret by just looking at their output. Our notion also appears better suited for a computational setting, where a party might "know" the secret from an information-theoretic point of view, yet be unable to output it.

Let $\mu_i(o)$ denote the utility of player P_i for the outcome o. Following [HT04] and all subsequent work in this area, we make the following assumptions about the utility functions of the players:

- If $o_i > o'_i$, then $\mu_i(o) > \mu_i(o')$.
- If $o_i = o'_i$ and $\sum_i o_i < \sum_i o'_i$, then $\mu_i(o) > \mu_i(o')$.

That is, player P_i first prefers outcomes in which he learns the secret; otherwise, P_i prefers strategies in which the fewest number of other players learn the secret. For simplicity, in our analysis we distinguish three cases, described from the point of view of P_i (though we stress that we could also work with utilities satisfying the more general constraints above):

1. If o is an outcome in which P_i learns the secret and no other player does, then $\mu_i(o) \stackrel{\text{def}}{=} U^+$.
2. If o is an outcome in which everyone learns the secret, then $\mu_i(o) \stackrel{\text{def}}{=} U$.
3. If o is an outcome in which P_i does not learn the secret, then $\mu_i(o) \stackrel{\text{def}}{=} U^-$.

(Note that U^+, U, U^- are considered constants, and are independent of the security parameter.) Our conditions impose $U^+ > U > U^-$. Define

$$U_{\text{random}} \stackrel{\text{def}}{=} \frac{1}{|\mathcal{S}|} \cdot U^+ + \left(1 - \frac{1}{|\mathcal{S}|}\right) \cdot U^- \ ; \tag{1}$$

this is the expected utility of a party who outputs a random guess for the secret (assuming other parties abort without any output, or with the wrong output). We will also assume that $U > U_{\text{random}}$; otherwise, players have essentially no incentive to run the reconstruction phase at all.

Strategies in our context refer to interactive Turing machines executing some protocol. Given a vector of strategies $\vec{\sigma}$ for some set of t^* parties active in the reconstruction phase, we let $U_i(\vec{\sigma})$ denote the expected utility of P_i; note that this is a function of the security parameter k. The expectation is taken over the initial choice of s (which we will always assume[1] to be uniform), the dealer's randomness, and the randomness of the players' strategies. Following standard game-theoretic notation, define $\vec{\sigma}_{-i} \stackrel{\text{def}}{=} (\sigma_1, \ldots, \sigma_{i-1}, \sigma_{i+1}, \ldots, \sigma_{t^*})$ and $(\sigma_i', \vec{\sigma}_{-i}) \stackrel{\text{def}}{=} (\sigma_1, \ldots, \sigma_{i-1}, \sigma_i', \sigma_{i+1}, \ldots, \sigma_{t^*})$; that is, $(\sigma_i', \vec{\sigma}_{-i})$ denotes the strategy vector $\vec{\sigma}$ with P_i's strategy changed to σ_i'.

In the following sections, we motivate and define the equilibrium notions used in this work. We focus on single-player deviations here, and defer the case of coalitions to Appendix D.1.

2.2 Notions of Game-Theoretic Equilibria: A Discussion

The starting point for any discussion of game-theoretic equilibria is the *Nash equilibrium*. Roughly speaking, a protocol induces a Nash equilibrium if no party gains any advantage by deviating from the protocol, as long as all other parties follow the protocol. (In a *computational* Nash equilibrium, no *efficient* deviation confers any advantage.) As observed by Halpern and Teague [HT04], however, the Nash equilibrium concept is too weak for rational secret sharing. Halpern and Teague suggest, instead, to design protocols that induce a Nash equilibrium *surviving iterated deletion of weakly dominated strategies*; this notion was used in subsequent work of [GK06, LT06, ADGH06].

The notion of surviving iterated deletion, though, is also problematic in several respects. Kol and Naor [KN08b] show a secret-sharing protocol that is "intuitively bad" yet satisfies the definition because no strategy weakly dominates any other: for any strategies σ, σ', there exist (contrived) strategies of the remaining players for which σ is the better strategy, and vice versa. (See [Kat08a, Kat08b] for other arguments against this notion.) Also, a definition of surviving iterated deletion taking *computational* issues into account has not yet been given and appears difficult to do.[2]

[1] This can be relaxed to assuming merely that s has high min-entropy; we assume a uniform distribution only for simplicity. Note that the assumption that s has high min-entropy is implicit in all prior work; if not, a party could guess the secret with high probability without running the protocol at all!

[2] The formal definition allows for infinitely many iterations in which strategies are deleted.

Motivated by these drawbacks (and more), Kol and Naor propose other strengthenings of Nash equilibrium: *resistance to backward induction* [KN08a], *everlasting equilibrium*, and *strict Nash equilibrium* [KN08b]. The latter two notions are defined in an information-theoretic sense, and are overly conservative in that they rule out protocols using cryptography; indeed, Kol and Naor state [KN08b] that these equilibrium notions should be considered *sufficient* but not *necessary*.

Nevertheless, the notion of strict Nash equilibrium is appealing. A protocol is in Nash equilibrium if no deviations are advantageous; it is in *strict* Nash equilibrium if all deviations are *dis*advantageous. Put differently, in the case of a Nash equilibrium there is no incentive to deviate whereas in the case of a strict Nash equilibrium there is an incentive *not* to deviate. Furthermore, if a protocol is strict Nash then it also survives iterated deletion of weakly dominated strategies.

Another advantage of strict Nash is that protocols satisfying this notion deter subliminal communication in the following sense: since *any* detectable deviation from the protocol results in lower utility (assuming other parties are following the protocol), a party who tries to use the messages of the protocol as a covert channel risks a loss in utility as long as there is some reasonable probability (not necessarily 1!) that other players are following the protocol. In fact, our protocols satisfy the following, stronger condition: at every point in the protocol, there is a *unique* legal message that a party can send. Our protocols thus rule out subliminal communication in a strong sense; this was an explicit goal in work such as [LMPS04, LMS05b, LMS05a, ASV08].

We propose here a *computational* version of strict Nash equilibrium. We believe our definition retains the intuitive appeal of strict Nash, while also meaningfully taking computational limitations into account (and thus enabling the use of cryptography).

Motivated by the suggestion in [Kat08a], we also define a computational notion of *stability with respect to trembles*. Intuitively, stability with respect to trembles means that even if a party P_i believes that the other parties might play some arbitrary strategy with small probability δ (but will follow the protocol with probability $1-\delta$), there is still no better strategy for P_i than to follow the protocol. This turns out to imply our notion of computational strict Nash; see Appendix A.

As should be clear from the above, determining the "right" game-theoretic notions for rational secret sharing is the subject of ongoing research. We do not suggest that the definitions proposed here are the *only* ones to consider, but we do believe they increase our understanding of the problem.

2.3 Definitions of Game-Theoretic Equilibria

Here we focus on single-player deviations; coalitions are dealt with in Appendix D.1. Throughout this section, Π is a t-out-of-n secret-sharing scheme and σ_i denotes the prescribed actions of P_i in the reconstruction phase.

Definition 1 Π induces a *computational Nash equilibrium* if for any $t^* \geq t$, any set $I = \{i_1, \ldots, i_{t^*}\}$ of size t^*, any $i \in I$, and any PPT strategy σ_i' we have that $U_i(\sigma_i', \vec{\sigma}_{-i}) \leq U_i(\vec{\sigma}) + \mathsf{negl}(k)$. $\diamondsuit$

Our definitions of strict Nash and resistance to trembles require us to first define what it means to "follow a protocol". This is non-trivial since two *different* Turing machines σ, ρ might be "functionally identical" as far as a protocol is concerned; for example, ρ may be the same as σ except that it first performs some useless computation. Other subtleties are discussed below.

Definition 2 Strategies σ_i, ρ_i *yield equivalent play with respect to* Π, denoted $\sigma_i \overset{\Pi}{\approx} \rho_i$, if for all PPT distinguishers D we have

$$\left|\Pr[D(1^k, \mathsf{view}_{-i}(\sigma_i, \vec{\sigma}_{-i})) = 1] - \Pr[D(1^k, \mathsf{view}_{-i}(\rho_i, \vec{\sigma}_{-i})) = 1]\right| \leq \mathsf{negl}(k),$$

where view_{-i} denotes the view of P_{-i} up to the point where the protocol ends. $\diamondsuit$

We write $\sigma_i \overset{\Pi}{\not\approx} \rho_i$ if σ_i, ρ_i do not yield equivalent play with respect to Π. Note that σ_i, ρ_i can yield equivalent play with respect to Π even if (1) they differ when interacting with some *other* set of strategies $\vec{\sigma}'_{-i}$ (we only care about their behavior when other parties run Π); (2) they differ in their local computation or output; and (3) they differ in messages sent *after* the protocol ends (once the protocol ends, there is no way to force players to behave one way or the other).

We now define the notion that *detectable* deviations from the protocol decrease a player's utility.

Definition 3 Π induces a *computational strict Nash equilibrium* if

1. Π induces a computational Nash equilibrium;
2. For any $t^* \geq t$, any set $I = \{i_1, \ldots, i_{t^*}\}$, any $i \in I$, and any PPT strategy σ'_i for which $\sigma'_i \overset{\Pi}{\not\approx} \sigma_i$, there is a $c > 0$ such that $U_i(\vec{\sigma}) \geq U_i(\sigma'_i, \vec{\sigma}_{-i}) + 1/k^c$ for infinitely many values of k. $\diamondsuit$

We now turn to defining stability with respect to trembles. We say that $\vec{\rho}_{-i}$ *is δ-close to* $\vec{\sigma}_{-i}$ if $\vec{\rho}_{-i}$ takes the following form: with probability $1 - \delta$ the parties play according to $\vec{\sigma}_{-i}$, while with probability δ they follow an arbitrary PPT strategy $\vec{\sigma}'_{-i}$. (In other words, $\vec{\rho}_{-i}$ differs from $\vec{\sigma}_{-i}$ with probability at most δ.) In this case, we refer to $\vec{\sigma}'_{-i}$ as the *residual strategy* of $\vec{\rho}_{-i}$.

Definition 4 Π induces a *computational Nash equilibrium that is stable with respect to trembles* if

1. Π induces a computational Nash equilibrium;
2. There exists a noticeable function δ such that for any $t^* \geq t$, any set $I = \{i_1, \ldots, i_{t^*}\}$, any $i \in I$, any vector of PPT strategies $\vec{\rho}_{-i}$ that is δ-close to $\vec{\sigma}_{-i}$, and any PPT strategy ρ_i, there exists a PPT strategy $\sigma'_i \overset{\Pi}{\approx} \sigma_i$ such that $U_i(\rho_i, \vec{\rho}_{-i}) \leq U_i(\sigma'_i, \vec{\rho}_{-i}) + \mathsf{negl}(k)$. $\diamondsuit$

Intuitively, the definition means that even if a party P_i believes that the other parties might play some different strategy with some small probability δ, there is still no better strategy for P_i than to outwardly follow the protocol[3] (while possibly performing some additional local computation). Moreover, if Π induces a computational Nash equilibrium then any (polynomial-time) local computation performed by P_i will not help as long as other parties follow the protocol.

In Appendix A, we show that if Π is stable with respect to trembles then it induces a computational strict Nash equilibrium. In our proofs, we show that our protocols satisfy the former, possibly stronger definition.

2.4 Verifiable Random Functions (VRFs)

A VRF is a keyed function whose output is "random-looking" but can still be verified as correct, given an associated proof. The notion was introduced by Micali, Rabin, and Vadhan [MRV99], and various constructions in the standard model are known [MRV99, Dod02, Lys02, DY05, CL07]. The definition we use (see Appendix B) is stronger than the "standard" one in that it includes an uniqueness requirement on the *proof* as well, but the constructions of [Dod02, DY05] achieve it. (Also, we use VRFs only as a stepping stone to our construction based on trapdoor permutations.)

[3]Specifically, for any strategy ρ_i that does *not* yield equivalent play w.r.t. Π, there exists a strategy σ'_i that *does* yield equivalent play w.r.t. Π and performs essentially as well.

Reconstruction phase

At the outset of this phase, P_1 chooses $s_1^{(0)}$ uniformly from $\mathcal{S} = \{0,1\}^\ell$ and P_2 chooses $s_2^{(0)}$ the same way. Then in each iteration i, the parties do the following:

(**P_2 sends message to P_1:**) P_2 computes $y_2^{(i)} := \mathsf{Eval}_{sk_2}(i)$, $\pi_2^{(i)} := \mathsf{Prove}_{sk_2}(i)$ and $z_2^{(i)} := \mathsf{Eval}'_{sk'_2}(i)$, $\bar{\pi}_2^{(i)} := \mathsf{Prove}'_{sk'_2}(i)$. It sends $(y_2^{(i)}, \pi_2^{(i)}, z_2^{(i)}, \bar{\pi}_2^{(i)})$ to P_1.

(**P_1 receives message from P_2:**) P_1 receives $(y_2^{(i)}, \pi_2^{(i)}, z_2^{(i)}, \bar{\pi}_2^{(i)})$ from P_2. If P_2 does not send anything, or if $\mathsf{Vrfy}_{pk_2}(i, y_2^{(i)}, \pi_2^{(i)}) = 0$ or $\mathsf{Vrfy}'_{pk'_2}(i, z_2^{(i)}, \bar{\pi}_2^{(i)}) = 0$, then P_1 outputs $s_1^{(i-1)}$ and halts.

(**P_1 sends message to P_2:**) P_1 computes $y_1^{(i)} := \mathsf{Eval}_{sk_1}(i)$, $\pi_1^{(i)} := \mathsf{Prove}_{sk_1}(i)$ and $z_1^{(i)} := \mathsf{Eval}'_{sk'_1}(i)$, $\bar{\pi}_1^{(i)} := \mathsf{Prove}'_{sk'_1}(i)$. It sends $(y_1^{(i)}, \pi_1^{(i)}, z_1^{(i)}, \bar{\pi}_1^{(i)})$ to P_2.

If $\mathsf{signal}_1 = z_2^{(i)}$ then P_1 outputs $s_1^{(i-1)}$ and halts. Otherwise, it sets $s_1^{(i)} := \mathsf{share}_1 \oplus y_2^{(i)}$ and continues.

(**P_2 receives message from P_1:**) P_2 receives $(y_1^{(i)}, \pi_1^{(i)}, z_1^{(i)}, \bar{\pi}_1^{(i)})$ from P_1. If P_1 does not send anything, or if $\mathsf{Vrfy}_{pk_1}(i, y_1^{(i)}, \pi_1^{(i)}) = 0$ or $\mathsf{Vrfy}'_{pk'_1}(i, z_1^{(i)}, \bar{\pi}_1^{(i)}) = 0$, then P_2 outputs $s_2^{(i-1)}$ and halts.

If $\mathsf{signal}_2 = z_1^{(i)}$ then P_2 outputs $s_2^{(i-1)}$ and halts. Otherwise, it sets $s_2^{(i)} := \mathsf{share}_2 \oplus y_1^{(i)}$ and continues.

Figure 1: The reconstruction phase of secret-sharing protocol Π.

3 2-out-of-2 Rational Secret Sharing using VRFs

We give a protocol Π for 2-out-of-2 secret sharing, and show that Π induces a computational Nash equilibrium stable with respect to trembles. Let $\mathcal{S} = \{0,1\}^\ell$ be the domain of the secret, where ℓ may depend on k. Let $(\mathsf{Gen}, \mathsf{Eval}, \mathsf{Prove}, \mathsf{Vrfy})$ be a VRF with range $\{0,1\}^\ell$, and let $(\mathsf{Gen}', \mathsf{Eval}', \mathsf{Prove}', \mathsf{Vrfy}')$ be a VRF with range $\{0,1\}^k$ (cf. Appendix B). The protocol is defined as follows:

Sharing phase: Let s denote the secret. The dealer chooses an integer $i^* \in \mathbb{N}$ according to a geometric distribution with parameter[4] β, where β is a constant that depends on the players' utilities but is independent of the security parameter; we discuss how to set β below. We assume $i^* < 2^k - 1$ since this occurs with all but negligible probability. (Technically, if $i^* \geq 2^k - 1$ the dealer can just send a special error message to each party.)

The dealer computes $(pk_1, sk_1), (pk_2, sk_2) \leftarrow \mathsf{Gen}(1^k)$ and $(pk'_1, sk'_1), (pk'_2, sk'_2) \leftarrow \mathsf{Gen}'(1^k)$, and:

- $\mathsf{share}_1 := \mathsf{Eval}_{sk_2}(i^*) \oplus s$ and $\mathsf{share}_2 := \mathsf{Eval}_{sk_1}(i^*) \oplus s$;
- $\mathsf{signal}_1 := \mathsf{Eval}'_{sk'_2}(i^* + 1)$ and $\mathsf{signal}_2 := \mathsf{Eval}'_{sk'_1}(i^* + 1)$.

Finally, the dealer gives to P_1 the values $(sk_1, sk'_1, pk_2, pk'_2, \mathsf{share}_1, \mathsf{signal}_1)$, and gives to P_2 the values $(sk_2, sk'_2, pk_1, pk'_1, \mathsf{share}_2, \mathsf{signal}_2)$.

Reconstruction phase: A high-level overview of the protocol was given in Section 1.1, and we give the formal specification in Figure 1. The reconstruction phase proceeds in a series of iterations, where each iteration consists of one message sent by each party. Although these messages could be sent at the same time (since they do not depend on each other), we do not want to assume synchronous communication and therefore simply require P_2 to communicate first in each iteration.

We give some intuition as to why the reconstruction phase of Π is a computational Nash equilibrium for an appropriate choice of β. Assume P_2 follows the protocol, and consider possible

[4]That is, the dealer flips a biased coin whose probability of landing heads is β and sets i^* to be the number of coin flips until the first head occurs.

deviations by P_1. (Deviations by P_2 can be analyzed symmetrically. In fact, that case is easier to analyze since P_2 goes first in every iteration.) There are essentially two things P_1 can do: it can abort in iteration $i = i^* + 1$ (i.e., as soon as it receives $z_2^{(i)} = \mathsf{signal}_1$), or it can abort in some iteration $i < i^* + 1$. In the first case, when P_1 aborts in iteration $i^* + 1$, party P_1 "knows" that it learned the dealer's secret in the preceding iteration (that is, in iteration i^*) and can thus output the correct secret; however, P_2 will output $s_2^{(i^*)} = s$ and so learns the secret as well (cf. Section 2.1). Thus, in this case, P_1 does not increase its utility beyond what it would achieve by following the protocol. In the second case, when P_1 aborts in some iteration $i < i^* + 1$, party P_1 does not know whether $i = i^*$ or not. The best strategy it can adopt is to output $s_1^{(i)}$ and hope that $i = i^*$. The expected utility that P_1 obtains by following this strategy can be calculated as follows:

- The probability that P_1 aborts exactly in iteration $i = i^*$ is roughly β.
- When $i < i^*$, player P_1 has "no information" about s and so the best it can do is guess. The expected utility of P_1 in this case is thus at most U_{random} (cf. Equation (1)).
- U^+ is always an upper bound for the utility of P_1, even when $i = i^*$.

Putting everything together, the expected utility of P_1 following this strategy is at most

$$\beta \times U^+ + (1 - \beta) \times U_{\text{random}} .$$

Since $U_{\text{random}} < U$ by assumption, it is possible to set β so that the entire expression above is strictly less than U; in that case, P_1 has no incentive to deviate. This is formalized in the following theorem, whose proof is given in Appendix C.

Theorem 1 *Let β be such that $U > \beta \cdot U^+ + (1 - \beta) \cdot U_{\text{random}}$. Then Π, using this value of β, induces a computational Nash equilibrium stable with respect to trembles.*

3.1 Using Trapdoor Permutations Instead of VRFs

The protocol from the previous section can be adapted rather easily to use trapdoor permutations rather than VRFs. The key observation is that the VRFs in the previous protocol were used only in a very specific way: the VRF was applied to inputs *in order*. What we can do instead is to use a trapdoor permutation f with associated hardcore bit h to instantiate the VRF in our scheme in the following way: The public key will be a description of f and a random element y in the domain of f; the secret key will be the trapdoor enabling inversion of f. In iteration i, the "evaluation" of the VRF on "input" i will be the ℓ-bit sequence

$$h\left(f^{-(i-1)\ell-1}(y)\right), h\left(f^{-(i-1)\ell-2}(y)\right), \ldots, h\left(f^{-(i-1)\ell-\ell}(y)\right) ,$$

and the "proof" will be $\pi_i = f^{-(i-1)\ell-\ell}(y)$. Verification can be done with respect to the original point y, but can also be done in time independent of i by using the previous proof π_{i-1} (namely, by checking that $f^\ell(\pi_i) = \pi_{i-1}$).

The key point is that the essential properties we need still hold: the properties of verifiability and uniqueness of proofs are easy to see, and pseudorandomness still holds but with respect to a modified game where the adversary queries $\mathsf{Eval}_{sk}(1), \ldots, \mathsf{Eval}_{sk}(i)$ and then the adversary has to guess whether it is given $\mathsf{Eval}_{sk}(i+1)$ or a random string. We omit further details.

Sharing Phase

To share a secret $s \in \{0,1\}^\ell$, the dealer does the following:

- Choose $r^* \in \mathbb{N}$ according to a geometric distribution with parameter β.
- Generate[a] $(pk_1, sk_1), \ldots, (pk_n, sk_n) \leftarrow \mathsf{Gen}(1^k)$ and $(pk'_1, sk'_1), \ldots, (pk'_n, sk'_n) \leftarrow \mathsf{Gen}'(1^k)$.
- Choose random $(t-1)$-degree polynomials $G \in \mathbb{F}_{2^\ell}[x]$ and $H \in \mathbb{F}_{2^k}[x]$ such that $G(0) = s$ and $H(0) = 0$.
- Send sk_i, sk'_i to player P_i, and send to all parties the following values:
  1. $\{(pk_j, pk'_j)\}_{1 \le j \le n}$
  2. $\{g_j := G(j) \oplus \mathsf{Eval}_{sk_j}(r^*)\}_{1 \le j \le n}$
  3. $\{h_j := H(j) \oplus \mathsf{Eval}'_{sk'_j}(r^* + 1)\}_{1 \le j \le n}$

Reconstruction Phase

Let I denote the set of the indices of the t active players. Each party P_i (for $i \in I$) chooses $s_i^{(0)}$ uniformly from $\{0,1\}^\ell$. In each iteration r, the parties do:

- For all $i \in I$ (in ascending order, say), P_i sends the following to all players:

$$\big(y_i^{(r)} := \mathsf{Eval}_{sk_i}(r),\ z_i^{(r)} := \mathsf{Eval}'_{sk'_i}(r),\ \mathsf{Prove}_{sk_i}(r),\ \mathsf{Prove}'_{sk'_i}(r) \big).$$

- If some party P_i receives an incorrect proof (or nothing) from some other party P_j, then P_i terminates and outputs $s_i^{(r-1)}$. Otherwise, P_i does as follows:
  - Set $h_j^{(r)} := h_j \oplus z_j^{(r)}$ for all $j \in I$. Interpolate a degree-$(t-1)$ polynomial $H^{(r)}$ through the t points $\{h_j^{(r)}\}_{j \in I}$. If $H^{(r)}(0) = 0$ then output $s_i^{(r-1)}$ and terminate.
  - Otherwise, compute $s_i^{(r)}$ as follows: set $g_j^{(r)} := g_j \oplus y_j^{(r)}$ for all $j \in I$. Interpolate a degree-$(t-1)$ polynomial $G^{(r)}$ through the points $\{g_j^{(r)}\}_{j \in I}$, and set $s_i^{(r)} := G^{(r)}(0)$.

[a] Gen outputs VRF keys with range $\{0,1\}^\ell$, and Gen' outputs VRF keys with range $\{0,1\}^k$.

Figure 2: Protocol $\Pi_{t,n}$ for "exactly t-out-of-n" secret sharing.

4 Extension to the t-out-of-n Case

In this section we describe extensions of our protocol to the t-out-of-n case, where we consider deviations by coalitions of up to $t-1$ parties. For formal definitions of our game-theoretic equilibrium notions in the case of coalitions, see Appendix D.1; these definitions are fairly straightforward extensions of the definitions given in Section 2.3.

In describing our protocols we use VRFs for notational simplicity, but all the protocols given here can be instantiated using trapdoor permutations using the ideas from the previous section.

A protocol for "exactly t-out-of-n" secret sharing. We begin by describing a protocol $\Pi_{t,n}$ for t-out-of-n secret sharing that assumes *exactly* t parties are active during the reconstruction phase, and is resilient to coalitions of up to $t-1$ of these parties. For now, we assume communication is over a synchronous (but not simultaneous) point-to-point network.

As in the 2-out-of-2 case, every party is associated with two keys for a VRF. The dealer chooses an iteration r^* according to a geometric distribution, and also chooses two random polynomials

G and H of degree $t-1$ such that $G(0) = s$ and $H(0) = 0$. Each party receives *blinded* versions of all n points $G(j)$ and $H(j)$: each $G(j)$ is blinded by the value of P_j's VRF on the point r^*, and each $H(j)$ is blinded by the value of P_j's VRF on the point $r^* + 1$. In each iteration r, all parties are supposed to send to all parties the value of their VRFs evaluated on the current iteration number r; once this is done, every party can interpolate a polynomial to obtain candidate values for $G(0)$ and $H(0)$. When $H(0) = 0$ parties know the protocol is over, and output the $G(0)$ value reconstructed in the *previous* iteration. See Figure 2 for details.

Theorem 2 *For appropriate choice of β, protocol $\Pi_{t,n}$ induces a $(t-1)$-resilient computational Nash equilibrium stable with respect to trembles as long as exactly t parties are active during the reconstruction phase.*

The proof is very similar to the proof of Theorem 1, and is given in Appendix D.2

Handling the general case. The prior solution assumes exactly t parties are active during reconstruction. If $t^* > t$ parties are active, the "natural" modification of the protocol — where the lowest-indexed t parties run $\Pi_{t,n}$ and all other parties remain silent — does not work. To see why, assume the active parties are $I = \{1, \ldots t+1\}$ and let $\mathcal{C} = \{3, \ldots, t+1\}$ be a coalition of $t-1$ parties. In each iteration r, as soon as P_1 and P_2 send their values, the parties in $\mathcal{C}$ can compute $t+1$ points $\{g_j^{(r)}\}_{j \in I}$. Because of the way these points are constructed, they are guaranteed to lie on a $(t-1)$-degree polynomial when $r = r^*$, but are unlikely to lie on a $(t-1)$-degree polynomial when $r < r^*$. This gives the parties in $\mathcal{C}$ a way to determine r^* in advance, at which point they can abort and learn the secret while preventing P_1 and P_2 from doing the same.

Nevertheless, a relatively simple modification works: simply have the dealer run independent instances $\Pi_{t,n}, \Pi_{t+1,n}, \ldots, \Pi_{n,n}$. Then, in the reconstruction phase, the parties run $\Pi_{t^*,n}$ where t^* denotes the number of active players. It follows as an easy corollary of Theorem 2 that this induces a $(t-1)$-resilient computational Nash equilibrium stable with respect to trembles regardless of how many parties are active during the reconstruction phase.

One remaining problem is with regard to coalitions $\mathcal{C}$ (with $|\mathcal{C}| > 1$) where some members of $\mathcal{C}$ are *not* active during the reconstruction phase; in this case, the same attack described above applies. (We remark that all prior work appears to assume implicitly that all players in a coalition are active during reconstruction.) This can be addressed by having the dealer run independent instances of $\Pi_{t,n}$ for all $\binom{n}{t}$ subsets of size t. Then any t^* active players can reconstruct the secret by having the t lowest-indexed players run the instance corresponding to their subset while the remaining players are silent. Obviously, this is only efficient for small values of t and we leave it as an open question to deal with this, more challenging case for general t, n.

Asynchronous networks. Our protocol $\Pi_{t,n}$ can be adapted fairly easily to work even when the parties communicate over an *asynchronous* point-to-point network. (In the asynchronous model, messages can be delayed arbitrarily and delivered out of order, but any message that is sent is eventually delivered.) In the asynchronous case, parties cannot distinguish an abort from a message that is delayed and so we modify the protocol as follows: each party proceeds to the next iteration r as soon as it has received $t-1$ valid messages for the previous iteration, and only halts if it receives an invalid message from some party.

As before, we can handle the general case by having the dealer run independent instances of the "exactly t^*-out-of-n" protocol just described for all values of $t^* \in \{t, \ldots, n\}$. (We assume that all parties are aware of how many other parties are active during the reconstruction phase.) This is resilient against any coalition $\mathcal{C}$ of size at most $t-1$ as long as all members of the coalition are active during reconstruction. The inefficient solution mentioned earlier, where $\binom{n}{t}$ independent

instances of the basic protocol are run, applies here as well to ensure resilience in the more general case when members of $\mathcal{C}$ may not be active during reconstruction.

More formal treatment of the asynchronous case, including a discussion of definitions in this setting, is deferred to Appendix D.3.

References

[ADGH06] I. Abraham, D. Dolev, R. Gonen, and J. Halpern. Distributed computing meets game theory: Robust mechanisms for rational secret sharing and multiparty computation. In *ACM Conf. on Principles of Distributed Computing (PODC) 2006*, pages 53–62. ACM Press, 2006.

[ASV08] J. Alwen, A. Shelat, and I. Visconti. Collusion-free protocols in the mediated model. In *Advances in Cryptology — Crypto 2008*, volume 5157 of *LNCS*, pages 497–514. Springer, 2008.

[Bla79] G.R. Blakley. Safeguarding cryptographic keys. *National Computer Conference*, 48:313–317, 1979.

[CL07] M. Chase and A. Lysyanskaya. Simulatable VRFs with applications to multi-theorem NIZK. In *Advances in Cryptology — Crypto 2007*, volume 4622 of *LNCS*, pages 303–322. Springer, 2007.

[Dod02] Y. Dodis. Efficient constructions of (distributed) verifiable random functions. In *Public-Key Cryptography (PKC) 2003*, volume 2567 of *LNCS*, pages 1–17. Springer, 2002.

[DR07] Y. Dodis and T. Rabin. Cryptography and game theory. In N. Nisan, T. Roughgarden, É. Tardos, and V.V. Vazirani, editors, *Algorithmic Game Theory*, pages 181–207. Cambridge University Press, 2007.

[DY05] Y. Dodis and A. Yampolskiy. A verifiable random function with short proofs and keys. In *Public Key Cryptography (PKC 2005)*, volume 3386 of *LNCS*, pages 416–431. Springer, 2005.

[GHKL08] D. Gordon, C. Hazay, J. Katz, and Y. Lindell. Complete fairness in secure two-party computation. In *40th Annual ACM Symposium on Theory of Computing (STOC)*, pages 413–422. ACM Press, 2008.

[GK06] S.D. Gordon and J. Katz. Rational secret sharing, revisited. In *Security and Cryptography for Networks (SCN)*, volume 4116 of *LNCS*, pages 229–241. Springer, 2006.

[GK08] D. Gordon and J. Katz. Partial fairness in secure two-party computation, 2008. Available at `http://eprint.iacr.org/2008/206`.

[HT04] J. Halpern and V. Teague. Rational secret sharing and multiparty computation. In *36th Annual ACM Symposium on Theory of Computing (STOC)*, pages 623–632. ACM Press, 2004.

[ILM08] S. Izmalkov, M. Lepinski, and S. Micali. Verifiably secure devices. In *5th Theory of Cryptography Conference (TCC)*, volume 4948 of *LNCS*, pages 273–301, 2008.

[IML05] S. Izmalkov, S. Micali, and M. Lepinski. Rational secure computation and ideal mechanism design. In *FOCS*, pages 585–595. IEEE Computer Society, 2005.

[Kat08a] J. Katz. Bridging game theory and cryptography: Recent results and future directions. In *Theory of Cryptography Conference (TCC)*, volume 4948 of *LNCS*, pages 251–272. Springer, 2008.

[Kat08b] J. Katz. Ruminations on defining rational MPC, 2008. Talk given at SSoRC, Bertinoro, Italy. Slides available at `http://www.daimi.au.dk/~jbn/SSoRC2008/program`.

[KN08a] G. Kol and M. Naor. Cryptography and game theory: Designing protocols for exchanging information. In *Theory of Cryptography Conference (TCC)*, volume 4948 of *LNCS*, pages 320–339. Springer, 2008.

[KN08b] G. Kol and M. Naor. Games for exchanging information. In *40th Annual ACM Symposium on Theory of Computing (STOC)*, pages 423–432. ACM Press, 2008.

[LMPS04] M. Lepinski, S. Micali, C. Peikert, and A. Shelat. Completely fair SFE and coalition-safe cheap talk. In *ACM Symposium on Principles of Distributed Computing (PODC)*, pages 1–10. ACM Press, 2004.

[LMS05a] M. Lepinski, S. Micali, and A. Shelat. Collusion-free protocols. In *37th ACM Symposium in Theory of Computing (STOC)*, pages 543–552. ACM Press, 2005.

[LMS05b] M. Lepinski, S. Micali, and A. Shelat. Fair-zero knowledge. In *2nd Theory of Cryptography Conference (TCC)*, volume 3378 of *LNCS*, pages 245–263, 2005.

[LT06] A. Lysyanskaya and N. Triandopoulos. Rationality and adversarial behavior in multi-party computation. In *Advances in Cryptology — Crypto 2006*, volume 4117 of *LNCS*, pages 180–197. Springer, 2006.

[Lys02] A. Lysyanskaya. Unique signatures and verifiable random functions from the DH-DDH separation. In *Advances in Cryptology — Crypto 2002*, volume 2442 of *LNCS*, pages 597–612. Springer, 2002.

[MRV99] S. Micali, M. Rabin, and S. Vadhan. Verifiable random functions. In *40th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 120–130. IEEE, 1999.

[MS09] S. Miclai and A. Shelat. Truly rational secret sharing. In *6th Theory of Cryptography Conference (TCC)*, 2009.

[OPRV09] S. J. Ong, D. Parkes, A. Rosen, and S. Vadhan. Fairness with an honest minority and a rational majority. In *6th Theory of Cryptography Conference (TCC)*, 2009. Available at `http://eprint.iacr.org/2008/097`.

[Sha79] A. Shamir. How to share a secret. *Communications of the ACM*, 22(11):612–613, 1979.

A Relating Two Equilibrium Notions

The following claim shows that Definition 4 implies Definition 3. We prove the claim for 2-out-of-2 secret sharing, but it extends easily to the general case.

Claim 1 *Let* Π *be a 2-out-of-2 secret-sharing protocol, where* σ_i *denotes the prescribed strategy of player* P_i*. If* Π *induces a computational Nash equilibrium that is stable with respect to trembles, then it induces a computational strict Nash equilibrium.*

Proof Assume toward a contradiction that Π does not induce a computational strict Nash equilibrium. Then P_1 (say) can deviate from the protocol without decreasing his utility. Formally, there exists a PPT strategy ρ_1 for which $\rho_1 \overset{\Pi}{\not\approx} \sigma_1$, yet $U_1(\sigma_1, \sigma_2) \leq U_1(\rho_1, \sigma_2) + \mathsf{negl}(k)$. Since $U_1(\sigma_1, \sigma_2) = U$, it must be that case that P_1 outputs the correct secret with all but negligible probability in an execution of ρ_1 against σ_2.

Since $\rho_1 \overset{\Pi}{\not\approx} \sigma_1$, there exists a PPT distinguisher D and a constant c such that

$$\Pr[D(1^k, \mathsf{view}_2(\rho_1, \sigma_2)) = 1] - \Pr[D(1^k, \mathsf{view}_2(\sigma_1, \sigma_2)) = 1] > 1/k^c$$

for infinitely many values of k. Define

$$p \overset{\text{def}}{=} \Pr[D(1^k, \mathsf{view}_2(\rho_1, \sigma_2)) = 1], \quad p' \overset{\text{def}}{=} \Pr[D(1^k, \mathsf{view}_2(\sigma_1, \sigma_2)) = 1].$$

Consider the following strategy ρ_2 for player P_2, parameterized by δ: with probability $(1-\delta)$, play σ_2; with probability δ, run σ_2 until the last protocol message is sent and then run D on the view. If D outputs 1 then output $\perp$; otherwise, output whatever is dictated by the protocol.

We have

$$\begin{aligned} U_1(\rho_1, \rho_2) &= (1-\delta) \cdot U_1(\rho_1, \sigma_2) + \delta p \cdot U^+ + \delta \cdot (1-p) \cdot U_1(\rho_1, \sigma_2) \\ &= (1 - \delta p) \cdot U_1(\rho_1, \sigma_2) + \delta p \cdot U^+ \\ &\geq (1 - \delta p) \cdot U + \delta p \cdot U^+ - \mathsf{negl}(k). \end{aligned}$$

On the other hand, let σ_1' be any PPT strategy with $\sigma_1' \overset{\Pi}{\approx} \sigma_1$. Then

$$\begin{aligned} U_1(\sigma_1', \rho_2) &\leq (1-\delta) \cdot U_1(\sigma_1', \sigma_2) + \delta \cdot (1-p') \cdot U_1(\sigma_1', \sigma_2) + \delta \cdot p' \cdot U^+ \\ &\leq \big(1 - \delta p'\big) \cdot U + \delta p' \cdot U^+. \end{aligned}$$

So

$$U_1(\rho_1, \rho_2) - U_1(\sigma_1', \rho_2) = \delta \cdot (p - p') \cdot (U^+ - U) - \mathsf{negl}(k),$$

and for any noticeable function δ the difference $U_1(\rho_1, \rho_2) - U_1(\sigma_1', \rho_2)$ is not negligible. This contradicts the assumption that Π induces a Nash equilibrium that is stable with respect to trembles. ■

B Verifiable Random Functions

Definition 5 A *verifiable random function* (VRF) with range $\mathcal{R} = \{\mathcal{R}_k\}$ is a tuple of probabilistic polynomial-time algorithms (Gen, Eval, Prove, Vrfy) such that the following hold:

Correctness: For all k, any (pk, sk) output by $\mathsf{Gen}(1^k)$, the algorithm Eval_{sk} maps k-bit inputs to the set $\mathcal{R}_k$. Furthermore, for any $x \in \{0,1\}^k$ we have $\mathsf{Vrfy}_{pk}(x, \mathsf{Eval}_{sk}(x), \mathsf{Prove}_{sk}(x)) = 1$.

Verifiability: For all (pk, sk) output by $\mathsf{Gen}(1^k)$, there does not exist a tuple (x, y, y', π, π') with $y \neq y'$ and $\mathsf{Vrfy}_{pk}(x, y, \pi) = 1 = \mathsf{Vrfy}_{pk}(x, y', \pi')$.

Unique proofs: For all (pk, sk) output by $\mathsf{Gen}(1^k)$, there does not exist a tuple (x, y, π, π') with $\pi \neq \pi'$ and $\mathsf{Vrfy}_{pk}(x, y, \pi) = 1 = \mathsf{Vrfy}_{pk}(x, y, \pi')$.

Pseudorandomness: Consider the following experiment involving an adversary $\mathcal{A}$:

1. Generate $(pk, sk) \leftarrow \mathsf{Gen}(1^k)$ and give pk to $\mathcal{A}$.
2. $\mathcal{A}$ adaptively queries a sequence of strings $x_1, \ldots, x_\ell \in \{0,1\}^k$ and is given $y_i = \mathsf{Eval}_{sk}(x_i)$ and $\pi_i = \mathsf{Prove}_{sk}(x_i)$ in response to each such query x_i.
3. $\mathcal{A}$ outputs a string $x \in \{0,1\}^k$ subject to the restriction $x \notin \{x_1, \ldots, x_\ell\}$.
4. A random bit $b \leftarrow \{0,1\}$ is chosen. If $b = 0$ then $\mathcal{A}$ is given $y = \mathsf{Eval}_{sk}(x)$; if $b = 1$ then $\mathcal{A}$ is given a random $y \leftarrow \mathcal{R}_k$.
5. $\mathcal{A}$ makes more queries as in step 2, as long as none of these queries is equal to x.
6. At the end of the experiment, $\mathcal{A}$ outputs a bit b'. We say $\mathcal{A}$ *succeeds* if $b' = b$.

We require that for any PPT adversary $\mathcal{A}$, the success probability of $\mathcal{A}$ is $\frac{1}{2} + \mathsf{negl}(k)$. $\diamondsuit$

C Proof of Theorem 1

We must first show that Π is a valid secret-sharing scheme. The secrecy property will follow from the proof that the reconstruction phase is a computational Nash equilibrium: if secrecy did not hold then computing the secret locally and not participating in the reconstruction phase at all would be a profitable deviation. We therefore focus on correctness. Assuming both parties run the protocol honestly, the correct secret is reconstructed unless:

- $i^* \geq 2^k - 1$.
- For some $i < i^* + 1$, either $\mathsf{signal}_1 = \mathsf{Eval}'_{sk'_2}(i)$ or $\mathsf{signal}_2 = \mathsf{Eval}'_{sk'_1}(i)$.

The first event occurs with negligible probability. Pseudorandomness of the VRF easily implies that the latter two events happen with only negligible probability as well.

We next prove that Π induces a computational Nash equilibrium. Assume P_2 follows the strategy σ_2 as prescribed by the protocol, and let σ'_1 denote any probabilistic polynomial-time strategy followed by P_1. (The other case, where P_1 follows the protocol and we look at deviations by P_2, follows similarly with an even simpler proof.) In a given execution of the reconstruction phase, let i denote the iteration, if any, in which P_1 aborts; if P_1 never aborts then we set $i = \infty$. Let early be the event that $i < i^*$; let exact be the event that $i = i^*$; and let late be the event that $i > i^*$. Let $\mathsf{correct}$ be the event that P_1 outputs the correct secret s. We will consider the probabilities of these events in two experiments: the experiment defined by running the actual secret-sharing scheme, and a second experiment where P_1 is given $\mathsf{share}_1, \mathsf{signal}_1$ chosen uniformly at random from the appropriate ranges. Probabilities in the first experiment will be denoted by $\Pr_{\text{real}}[\cdot]$, and probabilities in the second experiment will be denoted by $\Pr_{\text{ideal}}[\cdot]$. We have

$$\begin{aligned} & U_1(\sigma'_1, \sigma_2) \qquad (2) \\ & \leq \; U^+ \cdot \Pr_{\text{real}}[\mathsf{exact}] + U^+ \cdot \Pr_{\text{real}}[\mathsf{correct} \wedge \mathsf{early}] + U^- \cdot \Pr_{\text{real}}[\overline{\mathsf{correct}} \wedge \mathsf{early}] + U \cdot \Pr_{\text{real}}[\mathsf{late}], \end{aligned}$$

using the fact (as discussed in the intuition preceding the theorem) that whenever late occurs P_2 outputs the correct secret s. Since when both parties follow the protocol P_1 gets utility U, we need to show that there exists a negligible function ϵ such that $U_1(\sigma'_1, \sigma_2) \leq U + \epsilon(k)$. This is proved by the following claims.

Claim 2 *There exists a negligible function ϵ such that*

$$\begin{aligned}
|\Pr_{\text{real}}[\mathsf{exact}] - \Pr_{\text{ideal}}[\mathsf{exact}]| &\leq \epsilon(k) \\
|\Pr_{\text{real}}[\mathsf{late}] - \Pr_{\text{ideal}}[\mathsf{late}]| &\leq \epsilon(k) \\
|\Pr_{\text{real}}[\mathsf{correct} \wedge \mathsf{early}] - \Pr_{\text{ideal}}[\mathsf{correct} \wedge \mathsf{early}]| &\leq \epsilon(k) \\
|\Pr_{\text{real}}[\overline{\mathsf{correct}} \wedge \mathsf{early}] - \Pr_{\text{ideal}}[\overline{\mathsf{correct}} \wedge \mathsf{early}]| &\leq \epsilon(k).
\end{aligned}$$

Proof This follows easily from the pseudorandomness of the VRFs. ■

Let abort be the event that P_1 aborts before iteration i^*+1 (so $\mathsf{abort} = \mathsf{exact} \vee \mathsf{early}$). Define

$$\begin{aligned}
U_{\text{ideal}} \stackrel{\text{def}}{=} \; & U^+ \cdot \Pr_{\text{ideal}}[\mathsf{exact} \wedge \mathsf{abort}] + U^+ \cdot \Pr_{\text{ideal}}[\mathsf{correct} \wedge \mathsf{early} \wedge \mathsf{abort}] \\
& + U^- \cdot \Pr_{\text{ideal}}[\overline{\mathsf{correct}} \wedge \mathsf{early} \wedge \mathsf{abort}] + U \cdot \Pr_{\text{ideal}}[\mathsf{late}].
\end{aligned}$$

Claim 2 shows that $|U_1(\sigma_1', \sigma_2) - U_{\text{ideal}}| \leq \epsilon(k)$ for some negligible function ϵ. It remains to bound U_{ideal}. Information-theoretically, we have

$$\Pr_{\text{ideal}}[\mathsf{exact} \mid \mathsf{abort}] = \beta \quad \text{and} \quad \Pr_{\text{ideal}}[\mathsf{correct} \mid \mathsf{early}] = \frac{1}{|\mathcal{S}|};$$

therefore,

$$\begin{aligned}
U_{\text{ideal}} &= U^+ \cdot \left(\beta \cdot \Pr_{\text{ideal}}[\mathsf{abort}] + \frac{1}{|\mathcal{S}|} \cdot (1-\beta) \cdot \Pr_{\text{ideal}}[\mathsf{abort}]\right) \\
&\quad + U^- \cdot \left(1 - \frac{1}{|\mathcal{S}|}\right)(1-\beta) \cdot \Pr_{\text{ideal}}[\mathsf{abort}] + U \cdot (1 - \Pr_{\text{ideal}}[\mathsf{abort}]) \\
&= U + \left(U^+ \cdot \left(\beta + \frac{1}{|\mathcal{S}|} \cdot (1-\beta)\right) + U^- \cdot \left(1 - \frac{1}{|\mathcal{S}|}\right)(1-\beta) - U\right) \cdot \Pr_{\text{ideal}}[\mathsf{abort}] \\
&= U + \left(\beta \cdot U^+ + (1-\beta) \cdot U_{\text{random}} - U\right) \cdot \Pr_{\text{ideal}}[\mathsf{abort}] \;\leq\; U \qquad (3)
\end{aligned}$$

using the fact that $\beta \cdot U^+ + (1-\beta) \cdot U_{\text{random}} - U < 0$. This completes the proof that Π induces a computational Nash equilibrium.

To complete the proof, we show that Π induces a computational Nash equilibrium stable with respect to trembles. This part of the proof relies heavily on the uniqueness property of the VRFs. In particular, the uniqueness property implies that in each iteration each party has a *unique* message it can send that will not be treated as an abort.

Let δ be a parameter we will fix later. Let ρ_2 denote any PPT strategy for P_2 that is δ-close to σ_2, and let ρ_1 be an arbitrary PPT strategy for P_1. We show the existence of a PPT strategy σ_1' satisfying the requirements of Definition 2. (Once again we focus on deviations by P_1 but the case of P_2 is analogous.)

Say an iteration-i message $(y_2^{(i)}, \pi_2^{(i)}, z_2^{(i)}, \bar{\pi}_2^{(i)})$ from P_2 *represents an abort* if $\mathsf{Vrfy}_{pk_2}(i, y_2^{(i)}, \pi_2^{(i)}) = 0$ or $\mathsf{Vrfy}'_{pk_2'}(i, z_2^{(i)}, \bar{\pi}_2^{(i)}) = 0$. Say P_2 *aborts* if it sends nothing or sends a message representing an abort. (These notions are defined analogously for P_1.) Strategy σ_1' is defined as follows:

1. Given input $(sk_1, sk_1', pk_2, pk_2', \mathsf{share}_1, \mathsf{signal}_1)$, run ρ_1 on this input. Set $\mathsf{aborted} := 0$.
2. In each iteration i:
   (a) Receive the iteration-i message m_i from P_2. If P_2 aborts, then set $\mathsf{aborted} := 1$.

(b) Give m_i to ρ_1 and get in response some message m'_i.

(c) If $\mathsf{aborted} = 1$ then forward m'_i to P_2; otherwise, compute the response $(y_1^{(i)}, \pi_1^{(i)}, z_1^{(i)}, \bar{\pi}_1^{(i)})$ as prescribed by Π and send that to P_2.

3. If $\mathsf{aborted} = 0$ then output whatever is prescribed by Π; otherwise, output whatever ρ_1 outputs.

When σ'_1 interacts with σ_2, then $\mathsf{aborted}$ is never set to 1; thus, σ'_1 and σ_1 yield equivalent play with respect to σ_2. It remains to show that $U_1(\rho_1, \rho_2) \leq U_1(\sigma'_1, \rho_2) + \epsilon(k)$ for some negligible function ϵ. Let $\hat{\rho}_2$ denote the "residual strategy" of ρ_2; i.e., $\hat{\rho}_2$ is run only with probability δ by ρ_2. In an interaction where P_1 follows strategy ρ_1, let abort denote the event that P_1 aborts before P_2 aborts, and let $p_{\text{abort}}(\sigma)$ be the probability of this event when P_2 follows strategy σ. We first claim that the only "advantage" to P_1 of playing ρ_1 rather than σ'_1 arises due to ρ_1 aborting first.

Claim 3 $U_1(\rho_1, \hat{\rho}_2) - U_1(\sigma'_1, \hat{\rho}_2) \leq p_{\text{abort}}(\hat{\rho}_2) \cdot (U^+ - U^-)$.

Proof Note that abort is well-defined in the interaction of σ'_1 with $\hat{\rho}_2$ since σ'_1 runs a copy of ρ_1 as a sub-routine. When abort does *not* occur, there are two possibilities: neither ρ_1 nor P_2 ever aborts, or P_2 aborts first. We consider these in turn:

- When neither ρ_1 nor P_2 aborts, the output of P_2 is unchanged whether P_1 is running σ'_1 or ρ_1. Furthermore, the output of P_1 when running σ'_1 is equal to the correct secret. Thus, the utility of P_1 when running σ'_1 is at least the utility of P_1 when running ρ_1.
- If P_2 aborts first, the outputs of both P_1 and P_2 will be identical regardless of whether P_1 runs σ'_1 or ρ_1; this follows because as soon as P_2 aborts, strategy σ'_1 "switches" to playing strategy ρ_1.

So, the utility obtained by playing σ'_1 can only possibly be less than the utility obtained by playing ρ_1 when abort occurs. The maximum difference in the utilities in this case is $U^+ - U^-$. ■

The next claim shows that abort occurs at least as often when ρ_1 interacts with σ_2 as when ρ_1 interacts with $\hat{\rho}_2$.

Claim 4 $p_{\text{abort}}(\sigma_2) \geq p_{\text{abort}}(\hat{\rho}_2)$.

Proof To see this, consider some view of ρ_1 on which it aborts first when interacting with $\hat{\rho}_2$. (The view includes both the information d_1 given to ρ_1 by the dealer as well as the messages from P_2.) Since ρ_1 aborts first and, in every iteration, there is a *unique* non-aborting message that P_2 can send, it follows that ρ_1 will also abort when interacting with σ_2 (who never aborts first) whenever ρ_1 is given d_1 from the dealer. The claim follows. ■

Define $U^* \stackrel{\text{def}}{=} \beta \cdot U^+ + (1-\beta) \cdot U_{\text{random}}$ and recall that $U^* < U$ by assumption. Now,

$$\begin{aligned} U_1(\rho_1, \rho_2) &= (1-\delta) \cdot U_1(\rho_1, \sigma_2) + \delta \cdot U_1(\rho_1, \hat{\rho}_2) \\ &\leq (1-\delta) \cdot \big(U + (U^* - U) \cdot p_{\text{abort}}(\sigma_2)\big) + \delta \cdot U_1(\rho_1, \hat{\rho}_2) + \mathsf{negl}(k), \end{aligned}$$

using Equation (3). Also,

$$\begin{aligned} U_1(\sigma'_1, \rho_2) &= (1-\delta) \cdot U_1(\sigma'_1, \sigma_2) + \delta \cdot U_1(\sigma'_1, \hat{\rho}_2) \\ &= (1-\delta) \cdot U + \delta \cdot U_1(\sigma'_1, \hat{\rho}_2)\,. \end{aligned}$$

It follows that

$$\begin{aligned} & U_1(\rho_1,\rho_2) - U_1(\sigma_1',\rho_2) \\ &= (1-\delta)\cdot(U^*-U)\cdot p_{\text{abort}}(\sigma_2) + \delta\cdot\big(U_1(\rho_1,\hat{\rho}_2) - U_1(\sigma_1',\hat{\rho}_2)\big) + \mathsf{negl}(k) \\ &\leq (1-\delta)\cdot(U^*-U)\cdot p_{\text{abort}}(\hat{\rho}_2) + \delta\cdot(U^+-U^-)\cdot p_{\text{abort}}(\hat{\rho}_2) + \mathsf{negl}(k)\,, \end{aligned}$$

using Claims 3 and 4. Since $U^* - U$ is strictly negative, $\delta > 0$ can be found for which the above expression is negligible for k large enough. This completes the proof.

D The t-out-of-n Case

D.1 Game-Theoretic Definitions for the Case of Coalitions

We view a coalition $\mathcal{C}$ as a set of parties who may coordinate their strategies in an arbitrary way. Since the coalition acts in unison, we treat the utility of the coalition as a whole and, in particular, view the coalition as having only a single output value (rather than viewing each member of the coalition as potentially outputting a different value). Let $\mu_{\mathcal{C}}(\cdot)$ denote the utility of the coalition $\mathcal{C}$. As before, we assume the following utilities:

1. If o is an outcome in which $\mathcal{C}$ learns the secret and no player outside $\mathcal{C}$ does, then $\mu_{\mathcal{C}}(o) = U^+$.
2. If o is an outcome in which all parties active during the reconstruction phase (including $\mathcal{C}$) learn the secret, then $\mu_{\mathcal{C}}(o) = U$.
3. If o is an outcome in which $\mathcal{C}$ does not learn the secret, then $\mu_{\mathcal{C}}(o) = U^-$.

If $\vec{\sigma} = (\sigma_{\mathcal{C}}, \vec{\sigma}_{-\mathcal{C}})$ then $U_{\mathcal{C}}(\vec{\sigma})$ denotes the expected utility of $\mathcal{C}$ when parties in $\mathcal{C}$ follow $\sigma_{\mathcal{C}}$ and every other party P_i follows σ_i.

As for the 2-out-of-2 case, we assume that a coalition has an incentive to run the protocol rather than just output a random value. Thus, we continue to assume that $U_{\text{rand}} < U$, where U_{rand} is defined in (1). This implies that there exists a $\beta > 0$ such that

$$\beta\cdot U^+ + (1-\beta)\cdot U_{\text{rand}} < U\,. \tag{4}$$

We adapt the definitions from Section 2.3, specialized to the case of coalitions. Throughout this section, Π denotes a t-out-of-n secret-sharing scheme and σ_i denotes the prescribed actions of P_i during the reconstruction phase.

Definition 6 Π induces an *r-resilient computational Nash equilibrium* if for any $t^* \geq t$, any set $I = \{i_1,\ldots,i_{t^*}\}$ of size t^*, any $\mathcal{C} \subset I$ with $|\mathcal{C}| \leq r$, and any PPT strategy $\sigma_{\mathcal{C}}'$ we have that $U_i(\sigma_{\mathcal{C}}', \vec{\sigma}_{-\mathcal{C}}) \leq U_i(\vec{\sigma}) + \mathsf{negl}(k)$. ♢

Definition 7 Π induces an *r-resilient computational strict Nash equilibrium* if

1. Π induces an r-resilient computational Nash equilibrium;
2. For any $t^* \geq t$, any set $I = \{i_1,\ldots,i_{t^*}\}$, any $\mathcal{C} \subset I$ with $|\mathcal{C}| \leq r$, and any PPT strategy $\sigma_{\mathcal{C}}'$ for which $\sigma_{\mathcal{C}}' \overset{\Pi}{\not\approx} \{\sigma_i\}_{i\in\mathcal{C}}$, there is a $c > 0$ such that $U_i(\vec{\sigma}) \geq U_i(\sigma_{\mathcal{C}}', \vec{\sigma}_{-\mathcal{C}}) + 1/k^c$ for infinitely many values of k.

$\diamondsuit$

Definition 8 Π induces an *r-resilient computational Nash equilibrium stable with respect to trembles* if

1. Π induces an r-resilient computational Nash equilibrium;
2. There is a noticeable function δ such that for any $t^* \geq t$, any set $I = \{i_1, \ldots, i_{t^*}\}$, any $\mathcal{C} \subset I$ with $|\mathcal{C}| \leq r$, any vector of PPT strategies $\vec{\rho}_{-\mathcal{C}}$ that is δ-close to $\vec{\sigma}_{-\mathcal{C}}$, and any PPT strategy $\rho_{\mathcal{C}}$, there exists a PPT strategy $\sigma'_{\mathcal{C}} \overset{\Pi}{\approx} \{\sigma_i\}_{i\in\mathcal{C}}$ such that $U_i(\rho_{\mathcal{C}}, \vec{\rho}_{-\mathcal{C}}) \leq U_i(\sigma'_{\mathcal{C}}, \vec{\rho}_{-\mathcal{C}}) + \mathsf{negl}(k)$.$\diamondsuit$

D.2 Proof of Theorem 2

We re-state the theorem for convenience.

Theorem 3 *Protocol $\Pi_{t,n}$ (cf. Figure 2) with β satisfying (4) induces a $(t-1)$-resilient computational Nash equilibrium stable with respect to trembles, as long as exactly t parties are active during the reconstruction phase. (That is, it satisfies Definition 8 when $t^* = t$.)*

Proof Assume there are t active players $I := \{i_1, \ldots, i_t\}$. Let $\mathcal{C} \subset I$ with $|\mathcal{C}| \leq t-1$.

We say that a player sends a *bad* message to another player if either he does not send anything or at least one of the proofs is invalid. Let $\sigma'_{\mathcal{C}}$ be an arbitrary PPT strategy, and in an execution of the protocol let r be the round where the *first* player in $\mathcal{C}$ sends a bad message to a player in $\overline{\mathcal{C}} = I \setminus C$. Let $\Pr_{\text{real}}[\cdot]$ refer to the probability of events in an execution of the actual secret-sharing scheme. Let early be the event that $r < r^*$, let exact be the event that $r = r^*$, and let late be the event that $r > r^*$. Let $\mathsf{correct}$ be the event that $\mathcal{C}$ outputs the correct secret. We have

$$\begin{aligned} &U_{\mathcal{C}}(\sigma'_{\mathcal{C}}, \vec{\sigma}_{-\mathcal{C}}) \\ &\quad\leq\ U^+ \cdot \Pr_{\text{real}}[\mathsf{exact}] + U^+ \cdot \Pr_{\text{real}}[\mathsf{correct} \wedge \mathsf{early}] + U^- \cdot \Pr_{\text{real}}[\overline{\mathsf{correct}} \wedge \mathsf{early}] + U \cdot \Pr_{\text{real}}[\mathsf{late}], \end{aligned}$$

using the fact that when $r > r^*$ all players learn the secret.

Let $\Pr_{\text{ideal}}[\cdot]$ denote the probabilities in an experiment in which the coalition is given random values for $\{g_j, h_j\}_{1\leq j\leq n}$.

Claim 5 *There exists a negligible function ϵ such that:*

$$\begin{aligned} |\Pr_{\text{real}}[\mathsf{exact}] - \Pr_{\text{ideal}}[\mathsf{exact}]| &\leq \epsilon(k) \\ |\Pr_{\text{real}}[\mathsf{late}] - \Pr_{\text{ideal}}[\mathsf{late}]| &\leq \epsilon(k) \\ |\Pr_{\text{real}}[\mathsf{correct} \wedge \mathsf{early}] - \Pr_{\text{ideal}}[\mathsf{correct} \wedge \mathsf{early}]| &\leq \epsilon(k) \\ |\Pr_{\text{real}}[\overline{\mathsf{correct}} \wedge \mathsf{early}] - \Pr_{\text{ideal}}[\overline{\mathsf{correct}} \wedge \mathsf{early}]| &\leq \epsilon(k). \end{aligned}$$

Proof We consider the coalition's knowledge after receiving all other parties' messages in some iteration $r \leq r^*$ (with $y_i^{(r)} = \mathsf{Eval}_{sk_i}(r)$ and $z_i^{(r)} = \mathsf{Eval}'_{sk'_i}(r)$):

$$\begin{array}{rlllll} \text{Their shares (for } j \in [n]) & g_j = G(j) \oplus y_j^{(r^*)}, & & h_j = H(j) \oplus z_j^{(r^*+1)} & & \\ \text{Their own VRF values } (i \in \mathcal{C}) & y_i^{(1)}, z_i^{(1)}, & \ldots, & y_i^{(r)}, z_i^{(r)}, y_i^{(r+1)}, z_i^{(r+1)}, & \ldots & \\ \text{The received values } (j \in I \setminus \mathcal{C}) & y_j^{(1)}, z_j^{(1)}, & \ldots, & y_j^{(r)}, z_j^{(r)} & & \end{array}$$

The claim follows from the fact that G is a random polynomial (since s is chosen at random) and pseudorandomness of the VRF. ■

Define $\mathsf{abort} \stackrel{\text{def}}{=} \mathsf{exact} \vee \mathsf{early}$. Define

$$\begin{aligned} U_{\text{ideal}} \stackrel{\text{def}}{=} \; & U^+ \cdot \Pr_{\text{ideal}}[\mathsf{exact} \wedge \mathsf{abort}] + U^+ \cdot \Pr_{\text{ideal}}[\mathsf{correct} \wedge \mathsf{early} \wedge \mathsf{abort}] \\ & + U^- \cdot \Pr_{\text{ideal}}[\overline{\mathsf{correct}} \wedge \mathsf{early} \wedge \mathsf{abort}] + U \cdot \Pr_{\text{ideal}}[\mathsf{late}]. \end{aligned}$$

In the ideal setting, the coalition has (information theoretically) no information on the secret, so if the coalition aborts prior to round r^* it correctly guesses the secret with probability at most $\frac{1}{|\mathcal{S}|}$. Moreover $\Pr_{\text{ideal}}[\mathsf{exact} \mid \mathsf{abort}] = \beta$. We thus have

$$\begin{aligned} U_{\text{ideal}} &\leq U^+ \cdot \beta \cdot \Pr_{\text{ideal}}[\mathsf{abort}] + U_{\text{rand}} \cdot (1-\beta) \cdot \Pr_{\text{ideal}}[\mathsf{abort}] + U \cdot (1 - \Pr_{\text{ideal}}[\mathsf{abort}]) \\ &= U + \big(\beta \cdot U^+ + (1-\beta) \cdot U_{\text{rand}} - U\big) \cdot \Pr_{\text{ideal}}[\mathsf{abort}] \qquad (5) \\ &\leq U \end{aligned}$$

by the choice of β satisfying (4). This completes the proof that Π induces a $(t-1)$-resilient computational Nash equilibrium.

To prove resistance to trembles, we show that there is a constant $\delta > 0$ such that

$$\forall\, I = \{i_1, \ldots, i_t\}\ \forall\, \mathcal{C} \subset I\ (|\mathcal{C}| \leq t-1)\ \forall\, \vec{\sigma}'_{-\mathcal{C}}\ (\delta\text{-close to } \vec{\sigma}_{-\mathcal{C}})\ \forall\, \rho_{\mathcal{C}}\ \exists\, \sigma'_{\mathcal{C}}\ :$$
$$\sigma'_{\mathcal{C}} \stackrel{\Pi}{\approx} \sigma_{\mathcal{C}} \quad \wedge \quad U_{\mathcal{C}}(\rho_{\mathcal{C}}, \vec{\sigma}'_{-\mathcal{C}}) \leq U_{\mathcal{C}}(\sigma'_{\mathcal{C}}, \vec{\sigma}'_{-\mathcal{C}}) + \mathsf{negl}(k).$$

Let $\rho_{\mathcal{C}}$ be an arbitrary coordinated strategy of $\mathcal{C}$. Let $\vec{\sigma}'_{-\mathcal{C}}$ be a strategy for $\overline{\mathcal{C}} = I \setminus \mathcal{C}$ that is δ-close to the prescribed strategy $\vec{\sigma}_{-\mathcal{C}}$; this means that with probability $(1-\delta)$, all players P_i with $i \in \overline{\mathcal{C}}$ run σ_i. Let $\vec{\rho}_{-\mathcal{C}}$ be their "residual strategies"; i.e., the strategies that the play (with probability δ) when not running $\sigma_{-\mathcal{C}}$.

We say that a player *aborts* if he sends a *bad message* to any other player.

Claim 6 *Given the players' inputs from the dealer, there is only one possible execution of the reconstruction protocol in which no player aborts.*

Proof This follows immediately from the uniqueness property of the VRFs. ■

We define strategy $\sigma'_{\mathcal{C}}$ as follows:

- $\sigma'_{\mathcal{C}}$ simulates $\rho_{\mathcal{C}}$ in the background; i.e., every message received from outside the coalition is given to it.
- $\mathcal{C}$ follows the prescribed strategy $\sigma_{\mathcal{C}}$ until the point it receives the first bad message from a player in $\overline{\mathcal{C}}$. $\sigma'_{\mathcal{C}}$ then switches to $\rho_{\mathcal{C}}$.

Observe that when interacting with $\vec{\sigma}_{-\mathcal{C}}$, strategies $\sigma'_{\mathcal{C}}$ and $\sigma_{\mathcal{C}}$ yield equivalent play; i.e., $\sigma'_{\mathcal{C}} \stackrel{\Pi}{\approx} \sigma_{\mathcal{C}}$.

For any strategy $\vec{\tau}_{-\mathcal{C}}$, let $\mathsf{abort}(\vec{\tau}_{-\mathcal{C}})$ denote the event following event: when $\mathcal{C}$ plays $\rho_{\mathcal{C}}$ and $\overline{\mathcal{C}}$ plays $\vec{\tau}_{-\mathcal{C}}$, then a player from $\mathcal{C}$ sends a bad message to someone in $\overline{\mathcal{C}}$ before $\mathcal{C}$ receives a bad message; let $p_{\mathsf{abort}}(\vec{\tau}_{-\mathcal{C}})$ be the probability of $\mathsf{abort}(\vec{\tau}_{-\mathcal{C}})$.

Claim 7 $U_{\mathcal{C}}(\rho_{\mathcal{C}}, \vec{\rho}_{-\mathcal{C}}) - U_{\mathcal{C}}(\sigma'_{\mathcal{C}}, \vec{\rho}_{-\mathcal{C}}) \leq p_{\mathsf{abort}}(\vec{\rho}_{-\mathcal{C}}) \cdot \big(U^+ - U_{\min}\big)$, *where* $U_{\min}$ *is the minimum value* $\mu_{\mathcal{C}'}$ *takes for any* $\mathcal{C}'$.

Proof U^+ is the maximum utility, so $U^+ - U_{\min}$ is an upper bound for any difference of utilities. It suffices thus to prove that in the event of $\overline{\mathsf{abort}(\vec{\rho}_{-\mathcal{C}})}$, we have

$$U_{\mathcal{C}}(\rho_{\mathcal{C}}, \vec{\rho}_{-\mathcal{C}}) \;\leq\; U_{\mathcal{C}}(\sigma'_{\mathcal{C}}, \vec{\rho}_{-\mathcal{C}})$$

Consider the case where $\mathcal{C}$ never sends or receives a bad message. It follows from Claim 6 that $\rho_{\mathcal{C}}$ and $\sigma'_{\mathcal{C}}$ look the same to $\overline{\mathcal{C}}$. Moreover in an execution without abort, running $\sigma'_{\mathcal{C}}$ makes $\mathcal{C}$ output the secret, which is optimal.

Assume that a player in $\overline{\mathcal{C}}$ sends a bad message to $\mathcal{C}$ before $\rho_{\mathcal{C}}$ would send one. In this case, $\sigma'_{\mathcal{C}}$ switches to $\rho_{\mathcal{C}}$. Since there is only one possible set of messages sent so far ($\rho_{\mathcal{C}}$ did not send bad messages yet!), this means that the execution is equivalent to one where $\mathcal{C}$ played $\rho_{\mathcal{C}}$ from start. The two utilities are thus equivalent. ■

Claim 8 $p_{\mathsf{abort}}(\vec{\rho}_{-\mathcal{C}}) \;\leq\; p_{\mathsf{abort}}(\vec{\sigma}_{-\mathcal{C}})$

Proof We show that each time $\rho_{\mathcal{C}}$ makes a party in $\mathcal{C}$ send the first bad message when playing against $\vec{\rho}_{-\mathcal{C}}$, this would also have happened against $\vec{\sigma}_{-\mathcal{C}}$. Due to Claim 6, and the fact that no bad messages were sent from $\overline{\mathcal{C}}$ to $\mathcal{C}$ yet, $\vec{\rho}_{-\mathcal{C}}$ must have sent the same messages to $\mathcal{C}$ as $\vec{\sigma}_{-\mathcal{C}}$, which never aborts first. ■

Let $U^* := \beta \cdot U^+ + (1-\beta) \cdot U_{\text{rand}}$, which is strictly greater than U by (4). From (5) we have

$$U_{\mathcal{C}}(\rho_{\mathcal{C}}, \vec{\sigma}_{-\mathcal{C}}) \;\leq\; U + (U^* - U) \cdot p_{\mathsf{abort}}(\vec{\sigma}_{-\mathcal{C}}) + \epsilon(k)\,.$$

Thus,

$$U_{\mathcal{C}}(\rho_{\mathcal{C}}, \vec{\sigma}'_{-\mathcal{C}}) \;\leq\; (1-\delta) \cdot \Big(U + (U^* - U) \cdot p_{\mathsf{abort}}(\vec{\sigma}_{-\mathcal{C}}) + \epsilon(k)\Big) + \delta \cdot U_{\mathcal{C}}(\rho_{\mathcal{C}}, \vec{\rho}_{-\mathcal{C}}).$$

Moreover, $U_{\mathcal{C}}(\sigma'_{\mathcal{C}}, \vec{\sigma}_{-\mathcal{C}}) = U_{\mathcal{C}}(\sigma_{\mathcal{C}}, \vec{\sigma}_{-\mathcal{C}}) = U$ implies

$$U_{\mathcal{C}}(\sigma'_{\mathcal{C}}, \vec{\sigma}'_{-\mathcal{C}}) \;=\; (1-\delta) \cdot U + \delta \cdot U_{\mathcal{C}}(\sigma'_{\mathcal{C}}, \vec{\rho}_{-\mathcal{C}}).$$

Putting it all together, we get

$$\begin{aligned} U_{\mathcal{C}}(\rho_{\mathcal{C}}, \vec{\sigma}'_{-\mathcal{C}}) - U_{\mathcal{C}}(\sigma'_{\mathcal{C}}, \vec{\sigma}'_{-\mathcal{C}}) & \\ &\leq\; (1-\delta) \cdot (U^* - U) \cdot p_{\mathsf{abort}}(\vec{\sigma}_{-\mathcal{C}}) + \delta \cdot \big(U_{\mathcal{C}}(\rho_{\mathcal{C}}, \vec{\rho}_{-\mathcal{C}}) - U_{\mathcal{C}}(\sigma'_{\mathcal{C}}, \vec{\rho}_{-\mathcal{C}})\big) + \epsilon(k) \\ &\leq\; (1-\delta) \cdot (U^* - U) \cdot p_{\mathsf{abort}}(\vec{\rho}_{-\mathcal{C}}) + \delta \cdot (U^+ - U_{\min}) \cdot p_{\mathsf{abort}}(\vec{\rho}_{-\mathcal{C}}) + \epsilon(k), \end{aligned}$$

using Claims 7 and 8. Setting $\delta = \frac{-(U^*-U)}{U^+ - U_{\min} - (U^*-U)}$, which is strictly positive as $U^* - U$ is strictly negative, the above is negligible. ■

D.3 The Asynchronous Case

We begin with a few technical notes as to how we model the asynchronous setting:

1. As is standard in the asynchronous setting, we allow messages to be delayed and to be delivered in arbitrary order, but we assume eventual message delivery. (I.e., a message sent from one party to another will be received by time $t = \infty$.)

Sharing Phase

The sharing phase is identical to protocol $\Pi_{t,n}$ in Figure 2.

Reconstruction Phase

Let I denote the set of the indices of the t active players. Each party P_i (for $i \in I$) chooses $s_i^{(0)}$ uniformly from $\{0,1\}^\ell$ and writes it on its output tape. For $r = 1, \ldots,$ party P_i does:

- P_i sends the following to all players:

$$\big(y_i^{(r)} := \mathsf{Eval}_{sk_i}(r),\ z_i^{(r)} := \mathsf{Eval}'_{sk'_i}(r),\ \mathsf{Prove}_{sk_i}(r),\ \mathsf{Prove}'_{sk'_i}(r) \big).$$

- If P_i receives an incorrect proof from some other party P_j, then P_i terminates. (Note that if this occurs then the value $s_i^{(r-1)}$ is written on its output tape.) Otherwise, as soon as P_i receives $t-1$ valid messages for iteration r it does:
  - P_i sets $h_j^{(r)} := h_j \oplus z_j^{(r)}$ for all $j \in I$, and interpolates a degree-$(t-1)$ polynomial $H^{(r)}$ through the t points $\{h_j^{(r)}\}_{j \in I}$. If $H^{(r)}(0) = 0$ then P_i writes the value $s_i^{(r-1)}$ to its output tape and terminates.
  - Otherwise, P_i sets $g_j^{(r)} := g_j \oplus y_j^{(r)}$ for all $j \in I$, and interpolates a degree-$(t-1)$ polynomial $G^{(r)}$ through the points $\{g_j^{(r)}\}_{j \in I}$. It writes the value $s_i^{(r)} := G^{(r)}(0)$ to its output tape and proceeds to the next iteration.

Figure 3: Protocol $\Pi'_{t,n}$ for "exactly t-out-of-n" secret sharing in the asynchronous case.

2. We define an outcome o by the values recorded on the output tapes of the players at time $t = \infty$. In particular, it does not matter whether a party has halted or not; all that matters is the value written on its output tape (see the next item). This is essential, as even parties who follow the protocol honestly will not output the correct secret in any fixed time bound, and a party who deviates from the protocol can cause an honest party to never halt.

3. Due to the above, we allow parties to write a value to their output tape multiple times. Again, though, we stress that the value that "counts" as far as defining the outcome o is the value on a party's output tape at time $t = \infty$)

4. When considering possible deviations by a coalition $\mathcal{C}$, we allow $\mathcal{C}$ to schedule message delivery in the network (subject to constraint in item 1, above). The definition of "yielding equivalent play" (cf. Definition 2), however, still refers only to protocol messages sent by $\mathcal{C}$ and not to the way message delivery is scheduled. (We cannot hope to claim that if $\mathcal{C}$ changes the order of message delivery then its utility decreases.)

With this in mind, we present protocol $\Pi'_{t,n}$ for the asynchronous case in Figure 3. The protocol is mostly identical to protocol $\Pi_{t,n}$ (cf. Figure 2) except with regard to how aborts are handled.

We now sketch the proof that this protocol induces a $(t-1)$-resilient computational Nash equilibrium whenever exactly t parties are active in the reconstruction phase. (A formal proof of this fact, as well as a proof that the protocol induces a $(t-1)$-resilient computational Nash equilibrium stable with respect to trembles, will appear in the full version.) Assume some set of t parties I running the reconstruction phase, and consider some coalition $\mathcal{C} \subset I$ of size at most $t-1$. Let P^* denote the player in I who is not in $\mathcal{C}$. As usual, the best strategy for $\mathcal{C}$ is to wait until it can definitively identify iteration r^*, which occurs only after it receives the iteration-(r^*+1) message

from P^*. But P^* only sends its iteration-(r^*+1) message after it has received (valid) iteration-r messages from all the parties in $\mathcal{C}$. By this point, no matter what the parties in $\mathcal{C}$ do, P^* has the correct secret s written on its output tape.