Rational Secret Sharing

by

Alissa Natanovna Reyzin

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Master of Engineering in Computer Science and Engineering

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

August 2007

Author .....
Department of Electrical Engineering and Computer Science
August 10, 2007

Certified by .....
Silvio Micali
Professor
Thesis Supervisor

Accepted by .....
Arthur C. Smith
Professor of Electrical Engineering
Chairman, Department Committee on Graduate Students

Rational Secret Sharing

by

Alissa Natanovna Reyzin

Submitted to the Department of Electrical Engineering and Computer Science
on August 10, 2007, in partial fulfillment of the
requirements for the degree of
Master of Engineering in Computer Science and Engineering

Abstract

Recent work has attempted to bridge the fields of Cryptography and Game Theory in order to create more robust protocols that allow for a variety of player types. A keystone functionality used in cryptography is Secret Sharing. While there are several well known, efficient protocols that implement Secret Sharing in the Cryptographic model, designing a Rational Secret Sharing protocol which works in the Game Theoretic model has proved challenging.

In this thesis, we contrast several recently proposed protocols for Rational Secret Sharing based on their channel models, utility tolerances, equilibrium types, and efficiencies. We also discuss two more general results bridging Cryptography and Game Theory that can be used to construct Rational Secret Sharing protocols. Finally, we highlight several issues of context that influence the ways in which Rational Secret Sharing protocols might be used.

Thesis Supervisor: Silvio Micali
Title: Professor

Acknowledgments

I'd like to thank my adviser, Silvio Micali, for all of his guidance.

Contents

Chapter 1

Introduction

Cryptography and game theory are both fields dedicated to modeling and facilitating human interactions. Until recently, there was very little work that tried to bridge the models used in the two fields. We can learn a great deal about designing more robust and secure protocols by bringing together the strongest aspects of the Cryptographic and Game Theoretic models.

In this paper, we focus on a keystone cryptographic functionality called *secret sharing*. Protocols for secret sharing are widely used in distributed networking and are vital for implementing *secure function evaluation*. Several recent papers have proposed protocols for secret sharing that succeed in the game theoretic model, or in a combination of the two models. In this paper, we analyze the strengths and weaknesses of the proposed protocols and discuss possible directions for future research.

1.1 The Cryptographic and Game-Theoretic Adversarial Models

Traditional cryptographic protocols separate all agents into two categories - good agents and malicious agents. Proofs of security for such protocols are predicated on the assumption that agents classified as good will always follow the prescribed steps. Malicious agents, on the other hand, can act arbitrarily in order to disrupt the

protocol. Because malicious agents may communicate with each other to form secret coalitions, all the bad agents are often modeled as being controlled by a single entity called the *adversary*. A traditional cryptographic protocol guarantees privacy and correctness even when some percentage of the agents is controlled by the adversary.

In game theory, there is no distinction between good players and bad. Instead, each participant has utilities for every possible outcome of the game. The players wish to maximize their utilities and they act rationally in order to do so, based on their beliefs about the actions of the other players. In order for a rational player to act as a good player in the cryptographic model would, following the protocol must be a *dominant strategy* for her. That is, following the protocol must maximize her expected utility regardless of the actions of the remaining players.

1.2 The Weaknesses of the Two Models

Neither of the two models encompasses the other. The weakness of the cryptographic model is that it assumes some subset of the players to be unequivocally good. It is clear that this assumption is not always reasonable. One can argue that if the protocol contains no good players, then there is no reason for us to be concerned with its completion. While this is true in cases of functionalities such as encryption or signatures, it is not necessarily true for auctions or joint computations the completion of which accomplishes some social good.

There are several weaknesses of the game theoretic model as well. One weakness is the assumption that players are always rational. While in some cases an irrational player can be easily modeled as s rational player with different utilities, this is not necessarily true for all cases. An irrational player may choose actions randomly during play which can result in a sequence of actions that are not the rational path to *any* of the possible outcomes.

A second weakness of most game theoretic models is that they exclude the possibility of coalition formation. In most commonly used notions of equilibrium, the participants reason out their best action based on their beliefs about the actions of

others. There is no discussion as to where these beliefs originate, nor whether these beliefs can be changed through negotiation or conspiracy. There is also no discussion of what happens if several players are actually controlled by a single entity.

1.3 Harmonizing the Two Models

We would like to use tools from game theory in order to design cryptographic protocols that rational people will be motivated to follow. Although our eventual goal is to make possible a wide variety of protocols for rational agents, in this paper we choose to focus on a single task. Specifically, we compare and contrast several recently proposed protocols for *rational secret sharing* (which we abbreviate as RSS). We believe that secret sharing is a good representative protocol with which to begin our investigation.

Informally, a secret sharing protocol allows one agent, called the *dealer* to share a secret s amongst n other agents in such a way that any m of them can reconstruct the secret, but any group of fewer than m agents gains no knowledge about s.

There are several reasons why we choose to begin our investigation with secret sharing. Not only is secret sharing a useful functionality in its own right, it also enables secure function evaluation. Its many variants, such as proactive secret sharing and verifiable secret sharing, are often called upon in distributed computing. Finally, the traditional protocols for secret sharing are both practical and efficient, but it is easy to see that these protocols fail for rational participants. If we can create robust protocols for rational secret sharing, we will be well on our way to understanding general rational multiparty computation.

1.4 The State of the Art for Rational Secret Sharing

The name "rational secret sharing" was first introduced by Halpern and Teague [4] in 2004. Halpern and Teague pointed out that the reconstruction stage of the commonly used Shamir secret sharing protocol fails for players with certain very reasonable

incentives. Specifically, if all players prefer to learn the secret and also prefer that others not learn it, then no player will broadcast her share and the secret will never be reconstructed. Their goal, and the goal of several follow up papers ([14], [13], [15]), is to design a reconstruction protocol that incentivizes players to act in such a way that the secret is revealed to everybody.

Independently of Halpern and Teague's work, Lepinski, Micali, Peikert, and Shelat [7] described a protocol for fair SFE using envelopes. The property of fairness means that either all players receive the result of the computation, or none of the players do. This is exactly the property that is necessary for a successful RSS protocol. We discuss in detail how this result can be applied in order to create an RSS protocol in chapter 5.

In a later work, Izmalkov, Lepinski and Micali [5] described a method to convert any mediated normal form game into an unmediated game that uses the ballot box communication channel. Because rational secret sharing can be described as a mediated game, this result directly translates to an RSS protocol. We further discuss how to formalize rational secret sharing as a normal form game in chapter 4.

In contrast to [7] and [5], Halpern and Teague use simultaneous broadcast channels, which are not as powerful as either envelopes or the ballot box, in order to achieve RSS. However, their protocol has a less robust equilibrium as well as several other disadvantages. We describe Halpern and Teague's main results as well as the protocols of Gordon and Katz [14] and Abraham, Dolev, Gonen, and Halpern [13] which improved on [4] in chapter 6.

Chapter 2

Game Theoretic Definitions

2.1 Types of Games and Their Equilibria

2.2 Normal Form Games

A *normal form game* is a game in which all the players move only once and the moves are concurrent. Each player must choose her move without seeing the moves of others. There is a finite number of players, and a finite number of actions available to each one.

Definition 1 (Normal Form Game)

A normal form game is a triple $\{N, A, F\}$ *where:*

An action set A_i *is a non-empty set which represents the actions available to player* i.

The set Σ *is the set of* outcomes *of the game:* $\Sigma = \{A_1 \times A_2 \times ... \times A_n\}$

A payoff function F_i *maps an outcome to a number representing a player* i*'s utility:* $F_i = \Sigma \rightarrow \mathbb{R}$

$N = \{1, 2, 3...n\}$ *is the set of players.*

$A = \{A_1, A_2...A_n\}$ *is an n-tuple of action sets, one for each player.*

$F = \{F_1, F_2...F_n\}$ *is an n-tuple of payoff functions, one function per player.*

A normal form game is played as follows: Each player i independently selects an action a_i in A_i. After all the actions are chosen, i receives the payoff $F_i(a_1, a_2....a_n)$.

2.3 Extensive Form Games

An *extensive form game with perfect information* is a game in which players take turns making moves and the outcome of the game is not determined until all of the moves are complete. The term "perfect information" refers to the fact that players are aware of all the actions that occurred previously. In our definition, we allow every player to make a move at every discrete point in time. This is the most general notion, since it is always possible to restrict a player's action set to a single action if no moves are available to him at that time.

Definition 2 (Extensive Form Game with Perfect Information)

An extensive form game is a triple $\{N, H, F\}$ where:

A history H_i *is a sequence of vectors, where the components of each vector a^k are the actions of each player at time k. Each history represents a state that may occur during game play.*

$N = \{1, 2, 3...n\}$ *is the set of players.*

H is the set of possible histories of the game. The set H must contain the empty history e.

A terminal history *is a history $a^1, a^2...a^K \in H$ such that there is no other history $a^1, a^2...a^K, a^{K+1} \in H$. A game is over when a terminal history is reached. Let us call the set of terminal histories Z.*

A payoff function F_i maps a terminal history to a number representing player i's utility: $F_i = Z \rightarrow \mathbb{R}$

$F = \{F_1, F_2, ...F_n\}$ *is an n-tuple of payoff functions, one for each player.*

In order to play an extensive form game, each player must know the actions available to her at time t. The only way for a player to learn these actions from the game form above is to search through H for valid histories of length $t + 1$ that contain the current history as a subsequence. However, this process may take time exponential in

the length of the game. In order to avoid the problem, we define the functions $P_i : H Z \rightarrow A_i$ *which map histories to available actions for player* i.

An extensive form game with perfect information is played a follows: each player i *independently selects an action from* $P_i(e)$. *Each player is then informed of the current history* h. *If* h *is terminal, each player receives utility* $F_i(h)$. *If* h *is non-terminal, each player selects an action from* $P_i(h)$ *and the process repeats.*

Note that it is possible for an extensive form game to be infinite. *A game is infinite if and only if it has at least one branch that doesn't terminate in any finite number of steps.*

A variation of the extensive form game defined above is an extensive form game with imperfect information. In such a game, players may not be aware of the actions taken at prior times by other participants. Informally, each player knows himself to be in some information set, where an information set consists of one or more different histories. The player must choose her move with only the knowledge of the information set. She may not know the true history of the game so far.

2.4 Variations on Normal and Extensive Form Games

In both a normal form game and an extensive form game, it is possible that the players may not know the utility functions of the other participants. Games where this is the case are called games of *incomplete information.* In games of incomplete information, each player has a *type* that determines her utility function. The player's type is known only to herself and not to other players. The types are chosen from some distribution that is known a priori to all players.

Another common modification to both normal and extensive form games is to include nature as a player. Allowing nature to make moves represents the entropy that seeps into the game from the outside world. For example, if you are deciding which mode of transportation to use to get to work, this can be modeled as a game with nature: you choose whether to take the train or drive, and nature chooses the traffic conditions and train delays that you might encounter en route.

2.5 Nash Equilibrium

An equilibrium of a game is an outcome that rational players will be incentivized to play. The stronger the equilibrium, the more likely rational players are to play that equilibrium when told to do so by the designer.

The most commonly used equilibrium is called the Nash equilibrium. The essential intuition behind the Nash equilibrium is that it prevents single player deviations. If all players are told which Nash equilibrium is being played, and assuming that all other players stick to their prescribed strategies, it will be irrational for a single player to play anything other than her assigned strategy.

The notion of Nash equilibrium does not describe how the players agree on the specific equilibrium to play. Nor does it necessarily prevent other, possibly stronger, equilibria from being present in the game.

Definition 3 (Nash Equilibrium)

A Nash equilibrium of a normal form game $\{N, A, F\}$ *is an n-tuple of actions* σ^* *with the following property:* $\forall i \in N \; \forall \sigma_i \in A_i$

$$F_i(\sigma^*_{-i}, \sigma^*_i) \geq F_i(\sigma^*_{-i}, \sigma_i)$$

Similarly, it is possible to have a Nash equilibrium of an extensive form game. However, instead of an n-tuple of actions, a Nash equilibrium in an extensive form game consists of an n-tuple of *strategies.* A strategy for player i specifies which action i will take at every point in the game.

Let us call the set of all possible deterministic strategies for i S_i. *Note that for normal form games, the* $A_i = S_i$ *for all* i.

A Nash equilibrium of an extensive form game $\{N, H, F\}$ *is an n-tuple of actions* σ^* *with the following property:* $\forall i \in N \; \forall \sigma_i \in S_i$

$$F_i(\sigma^*_{-i}, \sigma^*_i) \geq F_i(\sigma^*_{-i}, \sigma_i)$$

2.6 ϵ-Nash Equilibrium

A slightly weaker variation on the Nash equilibrium useful for implementing games using standard cryptographic computational assumptions is the ϵ-Nash. Essentially, this equilibrium guarantees that no player will do more than ϵ better by deviating from the prescribed strategy. We define the remaining equilibria only for extensive form games, but it should be clear that normal form games are a subset of extensive form games, and therefore all equilibria apply to normal form games as well.

Definition 4 (ϵ-Nash Equilibrium) *For any $\epsilon > 0$ we define an ϵ-Nash equilibrium as a profile of strategies σ^* with the following property: $\forall i \in N \; \forall \sigma_i \in S_i$*

$$F_i(\sigma^*_{-i}, \sigma^*_i) + \epsilon \geq F_i(\sigma^*_{-i}, \sigma_i)$$

2.7 Iterated Deletion of Weakly Dominated Strategies

There are instances where a Nash equilibrium is not a sufficiently strong solution concept. A somewhat stronger solution concept is *Nash equilibrium with iterated deletion of weakly dominated strategies.* In order for a strategy τ of player i to be weakly dominated, there must exist another strategy that yields an outcome no worse than τ regardless of what strategies the other players use, and better in at least one case. Clearly, a rational player will never use a strategy that is weakly dominated, so we can remove all such strategies from consideration. The strategy profile of a Nash equilibrium with iterated deletion of weakly dominated only contains those strategies that survive iterated deletion.

Definition 5 (Iterated Deletion of Weakly Dominated Strategies)

A strategy $\sigma_i \in S_i$ is weakly dominated by strategy τ_i with respect to S if and only if $\exists \sigma_{-i} \in S_{-i}$

$$F_i(\sigma_{-i}, \tau_i) > F_i(\sigma_{-i}, \sigma_i)$$

and $\forall \sigma_{-i} \in S_{-i}$

$$F_i(\sigma_{-i}, \tau_i) \geq F_i(\sigma_{-i}, \sigma_i)$$

The strategies that survive iterated deletion are defined as follows:

Let S_i^0 *be the set of all strategies for i*

Let $S^k = \{S_1^k, S_2^k, ...S_n^k\}$

Let S_i^{k+1} *be the set of all strategies in* S_i^k *that are* not *weakly dominated with respect to* S^k.

Let S_i^∞ *be the intersection of all the* S_i^k.

S_i^∞ *is the set of strategies for i that survive any number of rounds of deletion.*

Note that iterated deletion of weakly dominated strategies is also applicable to both normal form and extensive form games.

2.8 k-resilient Equilibrium

A stronger notion of equilibrium which is resistant to deviations by coalitions of players was defined in [13]. Their notion of equilibrium prevents all deviations by coalitions of k or fewer players, even when the deviation would only benefit one of the members of the coalition.

Definition 6 (k-resilient Equilibrium) *For all* $|N| \geq k > 0$, *a k-resilient equilibrium of a game* $\{N, H, U\}$ *is an n-tuple of actions sigma* with the following property:* $\forall C \subset N, |C| \leq k \; \forall \sigma_C \in S_C \; \forall i \in C$

$$F_i(\sigma^*_{-C}, \sigma^*_C) \geq F_i(\sigma^*_{-C}, \sigma_C)$$

2.9 Dominant Strategy Equilibrium

The strongest possible notion of equilibrium is the dominant strategy equilibrium. A dominant strategy is a strategy that yields the best possible outcome no matter what the strategies of the other players.

Definition 7 (Dominant Strategy Equilibrium)

A dominant strategy equilibrium is a strategy profile σ^* with the following property: $\forall i \in N \; \forall \sigma \in S$

$$F_i(\sigma_i^*, \sigma_{-i}) \geq F_i(\sigma_i, \sigma_{-i})$$

Chapter 3

Basic Cryptographic Tools

In this chapter, we provide formal definitions of secret sharing, digital signatures and information checking. We also provide protocols for secret sharing and information checking. We informally sketch the concept of secure function evaluation and some known protocols for secure function evaluation. Finally, we provide a formalization of several traditional communication channels.

3.1 Shamir's Secret Sharing Protocol

Secret Sharing protocols were invented simultaneously by Shamir [10] and Blakley [16] in 1979. Both protocols allow a dealer to share a secret among n participants such that any m of them can reconstruct the secret, but it is information theoretically guaranteed that fewer than m participants cannot do so. Shamir's protocol, which is the more efficient and therefore more commonly used of the two, is described below. Note that a secret sharing protocol consists of two stages: the sharing of the secret by the dealer and the reconstruction by the participants. It is implicitly assumed that the dealer is unavailable when the reconstruction takes place; otherwise the dealer could simply broadcast the secret to all players when the need for reconstruction arises.

Definition 8 (Secret Sharing)

A secret sharing protocol consists of a pair of algorithms: Share *and* Reconstruct.

The Share algorithm takes a bit-string s and two integers n and m and outputs a vector of n bit-strings s_1 through s_n and some public information P.

$Share(s, n, m, k) = \langle s_1, s_2 .... s_n \rangle, P$

The Reconstruct algorithm takes an integer m' and a vector of bit strings S' and outputs either a bit-string s or the symbol bottom.

$Reconstruct(P, m', k, S') = s' \vee \perp$

The two algorithms satisfy the following properties: $\forall k \; \forall n \; \forall m < n$

$$Prob[s \leftarrow \{0,1\}^k; S, P \leftarrow Share(s, n, m, k); N \leftarrow \{R : R \subset \{1, 2..n\} \wedge |R| \geq m\};$$
$$S'_i \leftarrow S_i \text{ iff } i \in N \text{ and 0 otherwise } ; Reconstruct(P, k, m, S') = s] = 1$$

And $\forall$ *algorithms* A

$$Prob[s \leftarrow \{0,1\}^k; S, P \leftarrow Share(s, n, m, k); N \leftarrow \{R : R \subset \{1, 2..n\} \wedge |R| < m\};$$
$$S'_i \leftarrow S_i \text{ iff } i \in N \text{ and 0 otherwise}; A(P, m, n, S', k) = s] \leq 1/2^k$$

Protocol 1 (Shamir's Secret Sharing)

Share*(s, n, m):*

- *Choose a field F such that $|F| > n$ and $|F| > 2^k$.*
- *choose a random polynomial p over field F of degree $m - 1$ such that $p(0) = s$.*
- $S_i = p(i), P = F$

When the participants wish to reconstruct the secret, they broadcast all of their secret shares and then each one runs the Reconstruct algorithm.

Reconstruct*(P, m', k, S')*

- *Field $F = P$*

- *calculate* $p(0)$ *using the formula*

$$p(0) = \sum_{i=1}^{m} \prod_{j=1, j\neq i}^{m} \frac{x_j}{x_j - x_i} \cdot p(x_i)$$

and using the first m *entries of* S' *that are not* $\perp$.

Shamir's secret sharing protocol as presented above does not tolerate malicious players, since a malicious player could easily broadcast a false share during reconstruction and change the output of the interpolation. However, a trusted dealer whose key is well known can simply sign each share of the secret. The Reconstruct algorithm could then check that each share is properly signed before using it in the interpolation Since forging a cryptographic signature is extremely difficult, a computationally constrained adversary will not be able to send out an invalid share without being caught with overwhelmingly high probability.

3.2 Secure Function Evaluation (SFE)

The notion of Secure Function Evaluation (SFE) was introduced by Goldreich, Micali and Wigderson in [3] and was based on the two-party work of Yao ([12]). Since then SFE has become one of the central primitives in cryptography. It allows a set of n players to jointly evaluate a function on their private inputs without compromising either privacy or correctness. Essentially, an SFE protocol allows the participants to simulate a trusted party when no such party exists.

We do not present the formal notion of an SFE protocol here, but instead describe the key requirements in an intuitive manner. We also sketch two commonly used types of SFE protocols, one based on computational assumptions, the other based on the availability of perfectly secure communication between each pair of players as well as a broadcast channel.

Consider a randomized function $f(q_1, q_2 ... q_n) = \{t_1, t_2 ... t_n\}$ where each player i inputs q_i and receives the output t_i. An SFE protocol for function f must satisfy two main properties: privacy and correctness.

- Correctness: Correctness guarantees that the joint distribution of the outputs $\{t_1, t_2 ... t_n\}$ will be identical to the distribution that would occur if the function were evaluated by a trusted party.

- Privacy: Privacy guarantees that no player i can learn more about the inputs of the other players than she would have learned simply from seeing q_i and t_i. Privacy can be demonstrated by providing a simulator which can reconstruct something that looks like i's view of the protocol when given only q_i and t_i as inputs.

Another way to formalize an SFE protocol is to view it as emulating an ideal function evaluation. And ideal function evaluation is one in which each player privately sends her input to a trusted party, the trusted party evaluates f, and privately sends each player her output. This evaluation clearly provides the best possible privacy and security properties we can hope to achieve.

The GMW Protocol

The first SFE protocols were discovered in 1987 by Goldreich, Micali and Wigderson. Their protocols were based on computational assumptions, but did not rely on either private channels or a simultaneous broadcast channel. GMW actually described several closely related protocols with different properties. We list them below:

- The first protocol is secure only against *honest-but-curious* adversaries. Honest-but-curious adversaries always follow the stated protocol. However, they may try to extract information from the communications they are privy to, even when they are not told to do so.

  The protocol is secure against an arbitrary number (up to $n - 1$) of honest-but-curious players. This is the case even when all honest-but-curious players act together under the control of a single adversary. This protocol also has the property called *fairness*. Informally stated, fairness is the property that if a single player finds out her output, all players will do so.

- The second protocol is secure against any number (up to $n-1$) of malicious players. However a malicious player can abort the protocol at any time. This abort may occur even after some of the malicious players have received their outputs. Therefore, the protocol does not guarantee fairness. It is possible for the adversary to receive the function outputs of the players she controls, while the good players do not receive any information.

- The third protocol is secure against malicious players, but only as long as the majority of players are honest. This protocol guarantees fairness in addition to privacy and correctness.

The essential idea behind the implementation of all three of the GMW protocols is that the players all get shares of the inputs handed to them. They then simulate a circuit for evaluating the required function on those inputs. At every step in the evaluation, each player provides a proof that she is acting in accordance with the protocol. Finally, the participants send their shares of the circuit's outputs to the respective players.

The BGW Protocol

The BGW SFE protocols are secure without relying on any computational assumptions. Instead, the BGW protocols rely on the existence of perfectly secure private channels between every pair of protocol participants. Due to this assumption, the BGW protocols are perfectly, rather than computationally, secure with respect to computationally unbounded parties. The two main protocols described by BGW are summarized below.

- The first BGW protocol is secure only against a minority of honest-but-curious players. The honest-but-curious players can all work together in order to extract additional information from the protocol and the protocol is guaranteed to remain secure in spite of their efforts. This protocol is trivially fair, since honest-but-curious players can not abort the protocol.

- The second BGW protocol is secure against an arbitrarily malicious adversary as long as the adversary controls fewer than $n/3$ of the participants. In addition, the protocol is fair, in the sense that if any player receives her function output, all players must receive their shares.

For implementation details of the original BGW protocol see [1].

3.3 Digital Signatures

Digital signatures are a tool for data authentication. In order for player A to be able to authenticate a piece of information using digital signatures, she must publish something called her *public key*, to which she holds the corresponding secret key. Any player who has A's public key can verify that the signature is hers, but no one can forge the signature, except with a probability that is exponentially small in the security parameter.

All digital signatures rely on computational assumptions. For example, a common assumption that is sufficient in order to construct digital signatures is the existence of trapdoor permutations. The security of digital signatures is also dependent on the existence of some known upper bound on the computational power of the forger.

The now standard definition of digital signatures was first introduced by Goldwasser, Micali and Rivest [18] in 1988.

Definition 9 (Digital Signatures)

We denote a sequence of k ones by 1^k.

*A digital signature is a triple of probabilistic polynomial-time algorithms (*Generate, Sign, Verify*).*

The Generate algorithm takes a positive number k in unary notation and returns two strings a public key Pk, and a secret key Sk.

Generate$(1^k) = (Pk, Sk)$

The Sign algorithm takes a message m to be signed and a secret key Sk and outputs a signature s.

$Sign(Sk, m) = s$

The verify algorithm takes a message m, a signature s and a public key Pk and outputs either a true or false value.

$Verify(m, s, Pk) = bool$

The three algorithms satisfy the following properties: $\forall k$

$$Prob\{(Pk, Sk) \leftarrow Generate(1^k); m \leftarrow \{0,1\}^k; s \leftarrow Sign(Sk, m) : Verify(m, s, Pk) = true\} = 1$$

and $\forall c \ \exists k_0 \ \forall k \geq k_0 \ \forall$ *probabilistic polynomial-time algorithms* A

$$Prob\{(Pk, Sk) \leftarrow Generate(1^k); m \leftarrow \{0,1\}^k; s \leftarrow A(Pk, m, 1^k) : Verify(m, s, Pk) = true\} \leq 1/k^c$$

In order to see examples of digital signatures satisfying the definition above, see [19], [20], and [17]. In the rest of this paper, we will refer simply to digital signatures without specifying the protocol we are using. Any signature which satisfies the definition above will be sufficient.

3.4 Information Checking

Rabin and Ben-Or's [9] protocol for information checking was invented as an alternative to digital signatures for use in secret sharing protocols. This protocol allows several participants to verify the validity of another participant's data.

An information checking protocol allows a dealer to give data d to player A and verification data to player B in such a way that player A can prove to B that she received d from the dealer. However, B does not have any information about d until A chooses to reveal it. Information checking does not have the versatility of digital signatures, but is information-theoretically secure instead of being computationally secure. In addition, Rabin and Ben-Or's protocol is far simpler and more efficient

than any known digital signature.

Definition 10 (Information Checking)

*An information checking protocol consists of a triple of algorithms (*Generate, Share, Verify*).*

Generate takes in the positive integer k in unary notation and output some common information I.

$Generate(1^k) = I$

Share takes in a value between 1 and 2^k and the common information I and outputs a some check information h and verification information v.

$Share(d, I) = h, v$

Verify takes in a value between 1 and 2^k d, common information I, check information h and verification information v and outputs either true or false.

$Verify(d, I, h, v) = bool$

The algorithms must satisfy the following properties: $\forall k$

$$Prob\{I \leftarrow 1^k; d \leftarrow 0, 1^k; h, v \leftarrow Share(d, I) : Verify(d, I, h, v) = true\} = 1;$$

and $\forall k$ $\forall$ algorithms A

$$Prob\{I \leftarrow 1^k; d \leftarrow 0, 1^k; h, v \leftarrow Share(d, I) : A(I, v, 1^k) = d\} \leq 1/2^k$$

Protocol 2 (Rabin and Ben-Or's Protocol for Information Checking)

Generate:

- *Generate a field F that has at least 2^k elements. $I = F$*

Share:

- *Choose two random numbers $b \neq 0$ and h in the field described by I*
- *compute $d + bh = c$.*
- *the verification information $v = (b, c)$*

Verify:

- *get b and c from verification information v.*
- *if $d + bh = c$ output true. Otherwise output false.*

Note that because A does not know which verification data B has, and the verification data is chosen randomly and uniformly, the probability of A being able to lie about d and not get caught is indeed exponentially small in k.

3.5 Communication Channels

In real life, people have many ways to communicate with one another including on the phone, by email, face to face, or through letters. Each of these communication channels has different properties. In order to be able to discuss the relative strengths and weaknesses of several communication channels, we must formalize the notion of such a channel.

A communication channel is a trusted party that implements a small set of functionalities. Each channel is capable of carrying out one or more functionalities. Some of the functionalities are invoked at a player's request and others occur regardless of player actions at every discrete time t.

A functionality's input and output is described by a set of player-input pairs (i, d). An input pair (i, d) represents information d privately handed to the trusted party by i. An output pair (i, d) represents data d handed privately to player i. The functionality may also store certain information, as well as computing on the information it receives.

We denote the set of participants by N.

Definition 11 (Broadcast Channel) *The* broadcast *communication channel is a party that carries out the single functionality cast.*

$cast\{(i,p)\}$: *output* $\{(j,(i,p))|j \in N\}$

The functionality cast is carried out when any player invokes it. Intuitively the broadcast communication channel is identical to a player standing up in a room with all the other participants and making an announcement.

Definition 12 (Private Channels) *The private channel model also consists of a single functionality that can be invoked by any player at any time. We call the functionality tell.*

$tell\{(i,(p,j))\}$: *output* $\{(j,(i,p))\}$

Intuitively, this functionality is equivalent to player i passing a note to player j out of sight of all other players.

Definition 13 (Simultaneous Broadcast Channel) *The* simultaneous broadcast *communication channel carries out two functionalities. The first functionality is store, which is carried out whenever a player invokes it. The second functionality is castAll which is carried out at every discrete time step t. The trusted party maintains a set S which is empty at time 0.*

$S.add(e)$ *adds the element e to the set S*

$S.clear()$ *makes S into $\emptyset$*

$store\{(i,p)\}$: $S.add((i,p))$, *output* $\emptyset$

$castAll$: *output*$\{(j,S)|j \in N\}$, $S.clear()$

Intuitively, this channel is equivalent to every player independently placing information into an envelope marked with her name and a trusted party opening all of the envelopes.

Definition 14 (Envelope Channel)

An envelope communication model consists of the functionalities store, retrievePrivately, retrievePublicly and pass. The trusted party maintains a counter C and a set S of pairs $((i,p),l)$ where i is a player, p is some data and l is a numerical identifier which is empty at time 0.

$e.getData()$ *outputs the first element of entry e.*

$e.changeOwner(j)$; *changes* $e = ((i,p),l)$ *to* $((j,p),l)$

$S.add(e)$ *adds the element e to the set S*

S.clear() *makes* S *into* $\emptyset$

S.get(l) returns the element S *in which the second entry is* l

S.remove(l) removes the element of S *in which the second entry is* l

$store\{(i,p)\}:$ *S.add*$(((i,p),C))$, $C = C + 1$ *output* $\{(j,$ "*i created envelope* $C-1$"$)|$ $j \in N\}$

Intuitively, this functionality corresponds to sealing some secret data into a specially marked envelope in plain view.

$retrievePrivately(i,l):$ *output* $\{S.get(l).getData, (j,$ "*i privately opened envelope* l"$)$ $|j \in N)\}$; *S.remove*(l)

Intuitively, this functionality corresponds to opening an envelope in such a way that everyone knows you opened it, but only you see the data inside.

$retrievePublicly(i,l):$ *output* $\{(j,(S.get(l),$ "*i opened envelope* l"$))|j \in N\}$; *S.remove*(l)

Intuitively, this functionality corresponds to opening an envelope publicly.

$pass(i,(l,j)):$ *output* $\{(k,$ " *player* i *passed envelope* l *to player* j "$)|k \in N\}$; $S.get(l).changeOwner(j)$

Intuitively, this functionality corresponds to passing an envelope across the table to another player in plain view.

Definition 15 (Ballot-Box Channel)

We do not formally describe the Ballot-Box channel here. A technical definition is given in [5].

Intuitively, a Ballot-Box channel model consists of envelopes, which are formalized above, and an envelope randomizer called the ballot box. In this model, envelopes come in two sizes, small and large. A player may:

- *Place some secret data into an envelope.*
- *Open an envelope privately*
- *Open an envelope publicly*
- *Pass an envelope to another player*
- *Place up to 5 small envelopes into a large envelope. The envelopes will remain stacked in the order in which they were originally placed.*
- *Randomize at most 5 envelopes of the same size in such a way that all players know the envelopes have been randomized, but no one knows the resulting permutation.*

Moreover, the ballot box model makes the strong assumption that no communication channels other than the specified, publicly viewable ones are available to the players during the protocol.

We say that communication channel A is at least as strong as communication channel B if A can be used to simulate B. Clearly, the ballot box channel is at least as strong as the envelope channel. Similarly, the envelope channel is at least as strong as the simultaneous broadcast channel.

Chapter 4

An RSS Scheme Using the Ballot Box Channel

In this chapter, we describe how the result of Izmalkov, Lepinski and Micali [5] allows us to implement a protocol for rational secret sharing. Using the Izmalkov et al. result, we can transform any mediated normal form game into an unmediated game which relies on the ballot box channel. Although the Izmalkov et al. result appeared after Halpern and Teague's original paper on rational secret sharing, we choose to describe it first because it significantly simplifies the task of designing as RSS protocol.

4.1 The Mediated Game for RSS

In order to create a protocol for rational secret sharing we must first fully understand the mechanism that an RSS protocol implements and the equilibrium strength guaranteed by that mechanism. In addition, we must decide which assumptions we are willing to make on the strength of the communication channels, the utilities of the players, and the information available to the protocol designer. In this section, we discuss the mechanism that we are attempting to implement and the assumptions that will allow us to do so. We also discuss issues of context that may arise when the protocol is executed by real people in the real world.

In phase one of secret sharing, the dealer distributes shares to all the players. In

this phase, the dealer is the only one who acts, and she clearly acts in her own self interest. Thus, as long as we assume the existence of private channels between the dealer and each of the players in the share phase of the protocol, we can remove this phase from consideration and focus on the reconstruction phase. From now on, when we refer to an RSS protocol, we are referring specifically to the reconstruction phase of the protocol and not to the share phase.

Because the reconstruction phase requires a single decision from each player, we can formalize the mechanism for this phase as a normal form game with a mediator. In the reconstruction phase, each player holds a share of the secret. She has three choices - she may send the correct share of the secret to the mediator, an incorrect share of the secret, or nothing at all. The mediator must be able to verify whether the share of the secret is correct. Note that the mediator is *not* the same entity as the dealer and does not know the secret herself. If she did, she could simply send out the secret to all of the participants without the need to run a multiparty computation.

What should the trusted mediator do? Clearly, if she receives m or more valid shares, she should compute the secret and broadcast it to all the players. If she receives fewer than m valid shares, she should not disclose the secret to any of the players. But how can the mediator distinguish valid shares from invalid ones? We postpone this question for the next several paragraphs and instead assume that the mediator is capable of doing so.

We would like to guarantee that every player find out the secret. In order to make this guarantee, we must restrict the set of possible utilities of the players. For example, it must be the case that at least m players prefer learning the secret to not learning it. If this is not the case, fewer than m players will send their secret shares, and no one will learn anything. This assumption is quite natural; if fewer than m players want to learn the secret, the secret reconstruction should not be taking place.

If we had a guaranteed way of distinguishing valid shares from invalid ones, the above assumption would guarantee that sending out their true shares would be the dominant strategy for at least m players. Secret reconstruction would then be the dominant strategy equilibrium. However, no guaranteed way of verifying the validity

of a share exists. We discuss several possible imperfect methods below.

4.1.1 Modifying the Utilities to Prevent Lying

The easiest, and least satisfying, way to resolve the problem is to simply modify player utilities so that no player has an incentive to send a false share. This is the case if we assume that *all* players prefer to learn the secret regardless of what other players learn. This assumption is extremely strong as it does not allow for even one player with unexpected utilities.

Making this assumption yields a normal form mechanism for m out of n secret sharing where it is a dominant strategy equilibrium for each player to send out her real share.

4.1.2 Using Digital Signatures to Prevent Lying

Consider the following modification to the protocol: the dealer uses digital signatures to sign the shares when she hands them out, and each player sends her signed share to the mediator. The mediator will only consider a share valid if the signature is correct.

Because the security of digital signatures is based on computational assumptions, the use of digital signatures immediately implies that the players must be computationally bounded.

Even a computationally bounded player has a small probability of succeeding in forging the dealer's signature. When will a player want to forge a signature? She will attempt to forge the signature if and only if she wants to disrupt the protocol more than she wants to learn the secret. Therefore, sending the true secret share will still be dominant strategy for at least those m players who prefer to learn the secret. We can now view the the game in the following way: first, each player who has the incentives to do so uses her bounded resources to attempt to break the digital signature. Then the normal form game for RSS is played. Depending on whether any of the players succeed in breaking the signature, the normal form game may end in

everyone learning the secret, in no one learning the secret, or in everyone learning a false secret.

Everyone can learn the false secret only if someone succeeds in forging the digital signature. This will happen with probability less than any polynomial in the security parameter.

So, the dominant strategy for the m players who prefer to learn the secret is to send their secret share. The dominant strategy for those players who wish to disrupt the protocol more than they wish to learn the secret is to attempt to forge the signature. The dominant strategy for all other players is to do nothing. Therefore, there is a dominant strategy equilibrium which results in all players learning the secret with extremely high probability.

4.1.3 Using Information Checking to Prevent Lying

Another idea is to use Rabin and Ben-Or's [9] information checking protocol described in the previous chapter. Instead of receiving a signed secret share, every player will receive a share d and a verification parameter c as well as verification pairs (b_i, y_i) for every other player i.

What happens when a player tries to lie about her share of the secret? With very high probability, she will be unable to do so, and sending an incorrect share will be equivalent to sending nothing. With a small probability, sending an incorrect share may result in the players not reconstructing the secret, or reconstructing an incorrect secret. Even worse, a coalition of players could lie about their verification pairs. This would make it impossible for the mediator to tell which player is sending invalid information.

The easiest way to deal with this problem is to make assumptions about the utilities of the players. For example, if we assume that at most $m - 1$ players prefer messing up the protocol for everyone to learning the secret, then, with high probability, the mediator can figure out which $m - 1$ players have sent incorrect shares and ignore those shares.

The game then becomes a normal form game with nature, in which nature may

choose whether a player's invalid share actually looks valid to the mediator. (In reality, nature's choices are made by the dealer when she chooses the verification information.) The probability of this occurrence is exponentially small in the security parameter chosen by the dealer. This also does not change the fact that sending the correct shares is the dominant strategy for at least m players. Therefore, the secret recovery has an extremely high probability of success.

The dominant strategy equilibrium in this mechanism is for the m players who prefer to learn the secret to send their correct shares. The other players may prefer to lie about their shares, depending on their utilities.

4.2 Turning the Mediated Game into a Protocol

Izmalkov, Lepinski ,and Micali [5] define rational secure computation as a way to securely and privately simulate any normal form game without using a mediator. They provide a way to translate any normal form game which relies on a trusted party into a ballot box protocol which will have exactly the same equilibria. We can use the Izmalkov et al. result and the normal form game(s) that we just defined in order to create a ballot box protocol for secret sharing.

4.3 Putting the Game in Context

We have succeeded in formalizing the reconstruction phase of rational secret sharing as a normal form game (with nature, in one case). However, it is not clear that this formalization is valid in reality.

For example, one of our unspoken assumptions is that the players must all meet in the same room at the same time in order to run the protocol. But there is nothing that prevents m of the players from getting together secretly from the others and running a secret reconstruction protocol on their own. In this way, it is trivially possible for m colluding players to find out the secret without sharing it with the others.

Similarly, while a normal form game is only played one time, there is nothing that prevents players from running the protocol many times with different subsets of the players present.

There are two ways that we can avoid these problems of context. One way to deal with the problem is to make an extremely stronger assumption on the communication channels. For example, we can assume that our channels are only available by appointment, and that all the players must be present in order to use the channels. This would guarantee that the secret can only be reconstructed if all the players are present. However, this solution seems to go against the spirit of secret sharing. After all, the main purpose of secret sharing is to allow m players to reconstruct the secret even if the other players are unavailable or disinterested in the reconstruction.

A similar, but slightly more effective solution is to say that the channels are only available at a single point in time and that the game takes place regardless of which players are there at the announced time. We argue that all the players interested in reconstructing the secret will be present at the announced time, and that if at least m players are interested, the secret will be reconstructed. All players not present during the protocol will not learn the secret, but we argue that this is no different from the original protocol - if a player does not want to learn the secret, he can simply close his eyes and refuse to look at the output of the protocol.

It is important to realize that these problems of context will be present regardless of the specific RSS protocol. Unless the players all prefer to share the secret with everyone (in which case designing a protocol becomes unnecessary) or we make our communication channels extremely restrictive, m colluding players will always be able to learn the secret without sharing it with the others.

Chapter 5

An RSS Scheme using Envelopes

In the previous chapter, we saw an implementation of rational secret sharing using the Izmalkov, Lepinski, Micali result for Rational SFE. We now describe how a perfectly fair SFE can also be used to implement RSS. Specifically we discuss the perfectly fair SFE protocol of Lepinski, Micali, Peikert, and Shelat [7] which relies on envelope channels, and several ways that it can be transformed into an RSS protocol. Note that this result appeared independently from, and about a month after Halpern and Teague's original paper on RSS.

5.1 Using Fair SFE as a Black Box

Assume that we have a perfectly secure and fair SFE protocol. Let us use it to evaluate the Shamir's secret sharing Reconstruct function. Clearly, either all players will receive the output, or no players will. We need not worry about the possibility that a few players will see the secret while others are prevented from seeing it. Therefore, our only worry is whether any player will be motivated and able to lie about her shares of the secret, or to interrupt the protocol.

We can deal with the issue of lying by either modifying utilities, using digital signatures or using information checking, as described in the previous chapter.

Unfortunately, the fair SFE protocol of [7] allows a single bad player to abort the protocol. There are two ways to deal with this problem. The first is to make strong

assumptions on the utilities of the players - if every player is motivated to find out the secret, then no player will ever abort. The second way is to dig deeper into the [7] implementation of fair SFE and to use the ideas in a non-black box manner in order to create an RSS protocol that makes fewer assumptions on the player utilities.

5.2 RSS Using Envelopes

Rational secret sharing using envelopes has at its core the *secret data testing protocol* described originally in [21] and formalized in [7]. This protocol allows a prover to prove any public predicate Q about a string s known only to herself without revealing any other information about the string. The existence of the secret data testing protocol means that each player can create envelopes containing his secret share which is digitally signed by the dealer, and prove that the share inside the envelopes is in fact correctly signed. Once at least m players have submitted their envelopes and proved that the shares in the envelopes are correct, all envelopes are opened and the secret reconstructed.

There are several subtleties in the protocol described above. First, it relies on the fact that players are computationally bounded and cannot forge digital signatures. Second, the secret data testing protocol has some small probability of error that depends on the security parameter. If the security parameter is small and the advantage that a player gets from submitting a fake secret share is large, then a rational player will take the bet and try submit an incorrect share.

More formally, a player will try to submit an incorrect share if and only if this will increase his expected utility. Let us call the utility that player i receives when everyone gets the secret U_i^{all}, the utility she receives when nobody gets the secret U_i^{none} and the maximum possible utility she receives when she gets the secret but all others either do not get the secret or get an incorrect secret U_i^i. Then we simply need to make sure that prob (lying successfully)$\cdot U_i^i + 1-$ prob (lying successfully) $\cdot U_i^{none} < U_i^{all}$ for all players. Since the probability of cheating successfully is exponentially small in the security parameter, it is only necessary that the security parameter be polynomial in

the largest difference between U_i^i and U_i^{all}, which is a reasonable requirement.

Note that for this protocol to have an equilibrium that results in secret reconstruction, the protocol designer must properly set the security parameters based on both the computational abilities of all the players, and some limits on the differences between each player's utilities. This means that the protocol designer must be aware to at least some small extent of the players' types. In addition, even if the parameters are securely set, there is still a practically negligible, but theoretically important chance that some player will be able to forge a digital signature.

What kind of equilibrium does the protocol sketched above provide? We would like to say that it provides a dominant strategy equilibrium as long as m players are interested in learning the secret. This would trivially be the case if we disregarded the subtleties above. However, if only m players are interested in sharing and they suspect that some player has forged a digital signature, it may be better for the other players to keep quiet, in order to prevent her from learning the secret while they do not.

If the players trust the protocol designer to have properly set the security parameters, they should believe that the probability of forgery is exponentially small, as is the probability of tricking the SDT protocol. Therefore, in expectation, each of the m players will still be better off if they send out a valid share. Instead of simply sending their share, the dominant strategy for some of the m players may be: first try to forge a digital signature. If you succeed, send the false share. Otherwise, send your actual share.

Chapter 6

RSS Protocols using Simultaneous Broadcast and Private Channels

Halpern and Teague [4] were the first to identify the problem of rational secret sharing and to describe a protocol for it. Their work, as well as the follow up works of Gordon and Katz [14] and Abraham, Dolev, Gonen and Halpern [13] used simultaneous broadcast and private channels in order to implement RSS.

6.1 The Contributions of Halpern and Teague

One of Halpern and Teague's main contributions was to identify rational secret sharing as a functionality that can provide a great deal of insight into combining the cryptographic and game-theoretic realms. They also contributed two main technical results concerning RSS. The first result is a proof that there does not exist any protocol for RSS that has a fixed running time and uses simultaneous broadcast communication channels. The second is a basic randomized protocol for RSS in the simultaneous broadcast channel model.

6.1.1 Impossibility of Fixed Running Time RSS Protocols

Halpern and Teague show that certain types of rational secret sharing protocols do not exist. Their impossibility result demonstrates that there are no RSS protocols which satisfy the following requirements:

- The set of strategies prescribed by the dealer is an equilibrium at least as strong as a Nash equilibrium with iterated deletion of weakly dominated strategies.
- The channels used by the players during reconstruction are no stronger than simultaneous broadcast and private channels.
- There is a commonly known upper-bound on the number of rounds in the protocol.
- The range of incentive structures for which the protocol is an equilibrium includes at least one structure in which all players prefer to learn the secret (no matter what else occurs) and prefer to prevent others from learning it.

They also claim that their impossibility result applies to all two-player protocols, but Gordon and Katz [14] demonstrate that this is not the case.

We do not present the formal proof of the impossibility result in this thesis, but the intuition behind the proof is as follows:

- Assume that every player prefers all outcomes in which she learns the secret to all outcomes in which she does not. Further assume that every player prefers that as few as possible of the other players learn the secret.
- Consider any strategy that instructs players to send out valid information during the last round of the protocol. Clearly, this is weakly dominated by not sending anything during the last round of the protocol, since sending out information can only help other players learn the secret.
- Because all strategies in which useful information is sent out in the last round are weakly dominated, every player knows that no player will send any information

in the last round. Therefore, the next to last round effectively becomes the last round.

- Every round of iterated deletion of weakly dominated strategies removes those strategies in which information is sent during the last round, and therefore effectively removes the last round itself. After many rounds of iterated deletion of weakly dominated strategies, the only remaining strategy for any player will be to send nothing in any round. This clearly cannot lead to every player reconstructing the secret.

6.1.2 The Outline of the Protocol

In order to demonstrate that RSS is still possible, Halpern and Teague propose a randomized protocol that allows m out of n secret sharing for any $m, n > 2$. We choose not to describe the details of Halpern and Teague's protocol, since later protocols achieve results which are significantly stronger by using similar methods. The protocol's main properties are summarized below.

Halpern and Teague first describe a 3-out-of-3 RSS protocol and then use it to create any m-out-of-n protocol where m is at least 3. They do this by splitting the players into 3 groups and choosing one player in each group as a leader. All players in the group use private channels in order to send their shares to the group leader. The three group leaders then conduct the reconstruction protocol. The last stage of the protocol is one in which the group leaders broadcast their shares, and therefore all players learn the secret.

The 3-out-of-3 RSS protocol proceeds in rounds. Each round begins with the three players in possession of three valid secret shares signed by the dealer. With some probability, the secret is shared. If the secret is not shared in a given round, the dealer must send out new secret shares and a new round begins.

6.1.3 Limitations of the Halpern and Teague Protocol

We list the main limitations of the Halpern and Teague protocol.

1. The Halpern and Teague RSS protocol requires the dealer's presence during secret reconstruction. This begs the question as to why we are bothering to run an RSS protocol at all. If the dealer is present at all times, why should she not simply broadcast the secret to everyone whenever she wants the players to learn it? Halpern and Teague do not provide a satisfactory answer.

   Moreover, the protocol requires that the dealer have a private channel to every player during the course of the protocol, and that the dealer distribute valid shares to all the players. These assumptions are stronger than the basic broadcast channel assumption necessary in order for the dealer to simply tell everyone the secret whenever she wishes to do so.

2. The equilibrium provided by the protocol is a Nash equilibrium with iterated deletion of weakly dominated strategies. However, it is not the only Nash equilibrium in the game. While the protocol designer can instruct players to play specific strategies, there is no guarantee that rational players will not deliberate and then settle on a different equilibrium.

3. The protocol strategies are an equilibrium only if all players prefer learning the secret to not learning it. This means that the protocol cannot tolerate even a single player who is uninterested in learning the secret, is irrational, or has unexpected utilities. More formally, the protocol makes the following assumptions:

   - A player's utility function depends only on which players learn the secret. Let us denote by S the set of players that learn the correct secret. Then if u and u' are two outcomes such that $S(u) = S(u')$, $F_i(u) = F_i(u')$ for all i
   - Every player prefers any outcome in which she learns the secret to one in which she does not. If $i \in S(u)$ and $i \notin S(u')$, $F(u) > F(u')$.

4. The protocol is not resistant to coalitions. Recall that in an m-out-of-n secret reconstruction protocol, the ;players are divided into three groups, and each player sends her share to her group leader. Consider what would happen if the

three group leaders were in coalition. Clearly, once they have m secret shares, they could simply forgo the protocol entirely and reconstruct the secret without giving it to any of the other participants.

5. The protocol designer must be aware of the incentives of the players in order to choose parameter α, which determines the probability with which the secret will be reconstructed at each round. α is chosen such that for all players i if $S(u) = i$ and $S(u') = \emptyset$ and $S(u'') = N$

$$\frac{\alpha^2}{\alpha^2 + (1-\alpha)^2} \cdot F_i(u) + \frac{(1-\alpha^2)}{\alpha^2 + (1-\alpha)^2} \cdot F_i(u') \leq F(u'').$$

   Not only must the dealer know enough about the incentives of all the players in order to choose a sufficiently small α, she must also choose a sufficiently large α in order for the protocol to be practical. Because the protocol will run in expected $5/\alpha^3$ rounds, the protocol designer must choose an α that both guarantees the prescribed strategies to be an equilibrium and results in a protocol with a reasonably quick expected running time.

6. Halpern and Teague's protocol relies on secure digital signatures in order for the players to know that the reconstructed secret is the same secret that was distributed by the dealer. There are several troubling implications of using digital signatures that Halpern and Teauge do not make explicit.

   - In order for the players to verify the dealer's signature, they must know the dealer's public key in advance, or must receive the public key from the dealer when they receive their shares.
   - Because the security of digital signatures is based on computational assumptions, the use of digital signatures immediately implies that the players must be computationally bounded. This is an assumption that is not needed for any other aspect of the protocol.
   - Digital signatures are not perfectly secure. Even a computationally bounded player may, with a very small probability, be able to forge the dealer's dig-

ital signature and therefore send out a false share of the secret. This could cause all other players to reconstruct an incorrect secret, while the player who commits forgery learns the true secret. Therefore, it is quite plausible that rational players with the the types of utilities described above would attempt to forge the signature. The equilibrium strategy for every player motivated to forge the signature then becomes to first attempt to forge the signature, and only follow the protocol if forging does not succeed.

We can now view the the game in the following way: first, each player who has the incentives to do so uses her bounded resources to attempt to break the digital signature. Then the players attempt to execute the protocol. Depending on whether any of the players succeed in breaking the signature, the protocol may result in everyone learning the secret, in no one learning the secret, or in everyone learning a false secret.

Everyone can learn the false secret only if someone succeeds in forging the digital signature. This will happen with probability less than any polynomial in the security parameter. However, it is still a possibility that must be kept in mind. Moreover, it means that even if players reconstruct the secret, they can never be perfectly sure that the secret they have reconstructed is the correct one.

6.2 The Contributions of Gordon and Katz

The main contributions of Gordon and Katz's paper were the following:

- demonstrated that a two out of two rational secret sharing protocol is possible despite Halpern and Teague's apparent proof to the contrary
- described a significantly simplified protocol for RSS that had stronger properties than the Halpern and Teague protocol.
- suggested a method for removing the dealer from the reconstruction phase of the protocol.

We summarize the Gordon and Katz protocol below.

Protocol 3 *Gordon and Katz RSS Protocol*

We make the assumption that the secret s lies in a commonly known subset G of some field F. We also assume that there is some parameter β chosen by the protocol designer that is known to the dealer. β is chosen based on the utilities of the players in such a way that for all players i if $S(u) = i$ and $S(u') = \emptyset$ and $S(u'') = N$

$$\beta \cdot F_i(u) + (1 - B) \cdot F_i(u') \leq F_i(u'')$$

1. *With probability β the dealer distributes Shamir secret shares of s. Otherwise, she distributes shares of some $s' \in F/G$.*

2. *Each player has a flag* ***all-honest*** *which is set to true at the beginning of the protocol. If* ***all-honest*** *is true, each player broadcasts the share she received.*

3. *If at least m shares were broadcast, each player reconstructs the secret. If the secret is in G, then everyone knows the secret and the protocol is complete.*

4. *If all n shares were broadcast, and the secret can be reconstructed but the reconstructed secret is not in G, then the protocol repeats starting from the beginning.*

5. *Otherwise each player sets her* ***all-honest*** *to false and stops playing.*

Similarly to Halpern and Teague's protocol, this protocol relies on the fact that all players prefer to learn the secret regardless of anything else that occurs during the protocol. The equilibrium it provides is a coalition resistant Nash with iterated deletion of weakly dominated strategies.

6.2.1 Replacing the Dealer

Gordon and Katz notice that dealer involvement during the reconstruction phase is illogical, since a dealer who is present can always directly send the secret to all other players. To solve this problem, they suggest getting rid of the dealer and replacing

her with a secure function evaluation protocol. This SFE protocol takes the true shares and, with probability β, re-share the secret or, with probability $1 - \beta$ shares some s' not in G. It also signs the new shares appropriately. (This requires that all the players receive shares of the dealer's secret key in addition to the shares of the s).

How does replacing the dealer with an SFE protocol affect the protocol's properties? Gordon and Katz do not go into detail, but note that there are two options: the secure function evaluation can be either perfectly or computationally secure.

If it is perfectly secure, then the equilibrium remains the same. If anyone deviates from her assigned strategy, all the other players stop playing. Therefore, if every player is motivated to find out the secret, no player will deviate. The only caveat is that perfect SFE requires secure private channels in addition to broadcast channels. This is still a major improvement on the protocol of Halpern an Teague, however, since that protocol requires private channels as well as dealer presence.

Abraham, Dolev, Gonen, Halpern [13] offer a solution that uses computational SFE which we describe in the next section.

6.2.2 Limitations of the Gordon and Katz Protocol

We consider how the limitations of Gordon and Katz's protocol compare to the limitations of the Halpern and Teague protocol listed above.

1. Dealer presence: this limitation is removed as described above.

2. Equilibrium type: this problem remains unchanged. There are still other equilibria present in the game.

3. Utility assumptions: the assumptions on utilities are exactly the same as in Halpern and Teague's protocol.

4. Lack of resistance to coalitions: Although Gordon and Katz do not define coalition resistance in their paper, the original protocol they provide is a k-resilient Nash for all $k < m$. The protocol which does not rely on the dealer's presence

must rely on the BGW SFE protocol described in chapter 2. The BGW SFE can only handle up to 1/3 of the players being in a coalition.

5. Prior Awareness of Incentives: Although the parameter used in the protocol changes, the problem of having to know players' incentives in advance remains. In the case of Gordon and Katz protocol, the designer must choose a parameter β described above such that for all players

$$\beta \cdot F_i(\text{u}) + (1 - B) \cdot F_i(\text{u'}) \leq F_i(\text{u"})$$

β also cannot be chosen to be extremely small, because the protocol will take an expected $1/\beta$ rounds.

6. Reliance on computational assumptions: This limitation remains, since players still need to use digital signatures in order to verify the validity of shares sent by others.

6.3 The Contributions of Abraham, Dolev, Gonen, and Halpern

Abraham et al. wrote their paper concurrently with Gordon and Katz and their contributions were similar. Their main contributions were:

- demonstrated that two out of two secret sharing was possible
- defined k-resistant equilibria and demonstrated that their protocol was resistant to coalitions
- Described how to remove the dealer using computationally secure SFE and ϵ-Nash equilibria.
- used information checking to create a protocol with constant expected running time for $k < m \leq n - k$

- discussed how to deal with players who have unexpected utilities
- described an RSS protocol that doesn't require the dealer to be aware of player utilities for $k < m \leq n - 2k$.

The basic Abraham et al. protocol is almost equivalent to the Gordon and Katz protocol. The main difference is that they distinguish the trusted mediator used during the protocol reconstruction from the dealer. This distinction makes it easier to replace the mediator by an SFE protocol, since the SFE protocol need not have any special knowledge. They use digitally signed shares so that the mediator can confirm she is receiving a valid share.

Protocol 4 *Abraham, Dolev, Gonen and Haleprn RSS Protocol*

The protocol assumes that the secret $s \neq 0$ lies in a field F. It also assumes that there is some parameter β chosen by the protocol designer based on participating players' utilities that is known to the mediator.

1. *The dealer distributes digitally signed Shamir secret shares of s.*
2. *Each player has a flag* ***all-honest*** *which is set to true.*
3. *Each player sends his share of the secret to the mediator.*
4. *If at any point in the game the mediator does not receive all the appropriate messages from all other players, she stops playing.*
5. *With probability β the dealer re-shares the secret. Otherwise, she sends out secret shares of 0.*
6. *All players who have* ***all-honest*** *set to true broadcast the secret.*
7. *If at least m shares were broadcast, each player reconstructs the secret. If the secret can be reconstructed and it's not 0, then everyone knows the true secret and the protocol is complete.*

8. *If all n shares were broadcast, and the secret can be reconstructed but the reconstructed secret is 0, then each player sends an acknowledgement to the mediator, and we return to step 5.*

   Otherwise each player sets her ***all-honest*** *to false and stops playing.*

The protocol above is resistant to coalitions of size $m-1$ and runs in time linear in $1/\beta$.

Abraham et. al. do not mention the possibility of some player forging a digital signature. However, as long as $n > m$ a forgery would result in the polynomial being impossible to interpolate, and no-one learning the secret. And if $n = m$, then it will result in everyone learning the incorrect secret. Because the original secret shares are never broadcast, player i cannot learn s and prevent others from learning it even if she forges a digital signature.

6.3.1 Constant Round RSS

Let k be the size of the coalitions we want to be able to tolerate. Assume that $k < m \leq n - k$. Then it is possible to use information checking in order to create a two round protocol for RSS.

We modify the protocol as follows:

- The mediator chooses a field F from which the polynomial's coefficients are drawn so that $|F| > 1/\beta$ where β is a security parameter dependent on the players' utilities.
- The mediator re-shares the real secret with probability $1/2$
- In addition to distributing the re-shares of the secret, the mediator sends out a check information for every player to be able to check every other player's share.

Assume that some coalition of k players tried to send incorrect shares. Because $k < n - m$ there will always be sufficiently many correct shares sent out to guarantee

that secret reconstruction is still possible. The only question is whether the players will be able to distinguish correct shares from incorrect ones. Our information checking protocol guarantees that this will be the case with extremely high probability. Therefore if β is set correctly, no coalition will have incentive to deviate from the prescribed protocol.

6.3.2 RSS Without Knowledge of Utilities

Let k be the size of the coalitions we want to be able to tolerate. Assume that $k < m \leq n - 2k$. Then it is possible to use Reed-Solomon unique decoding (ref) in order to create a two round protocol for RSS without relying on the protocol designer's knowledge of player utilities.

Specifically, the protocol remains identical except that the mediator re-shares the secret with probability 1/2, and the players are instructed to use Reed-Solomon unique decoding when reconstructing the polynomial.

6.3.3 Replacing the Mediator

Abraham et al suggest replacing the mediator with an SFE protocol. They argue that if the SFE protocol is perfectly secure, the players will have the same incentives to follow the correct protocol as they did earlier: any deviation will either not affect the outcome, or will cause everyone to find out an incorrect secret, or no secret at all. Therefore, coalition members will have no incentive to deviate, but will act in an honest-but-curious manner.

There are several well known SFE protocols that can tolerate some proportion of honest-but-curious members. The BGW protocol is perfectly secure and can handle $k < n/2$ honest but curious members. However, in order to be able to handle a coalition of more than half the players, it is necessary to make some cryptographic assumptions.

The GMW protocol can handle up to $n - 1$ honest but curious players. However the GMW protocol is based on computational assumptions and is therefore only

computationally secure. This means that players have a small probability of being able to act in a way different from that specified by the protocol and not get caught. Notice that this is essentially the same problem as limitation 6 in both the Halpern and Teague and Gordon and Katz protocol.

Although it is impossible to remove the probability of error when cryptographic assumptions are used in the protocol, Abraham et al. succeed in formalizing the way in which the equilibrium is effected by the imperfection of cryptography. Instead of a k-resilient Nash equilibrium, they claim their protocol provides a k-resilient ϵ-Nash for any ϵ, where ϵ depends on the security parameter of the cryptography and on the utilities of the players.

Consider a player who attempts to break a computational assumption. If the player doesn't succeed, clearly she can not do better than to follow the prescribed protocol. If she does succeed, she might be able to do something different and receive a significantly higher utility. Let us assume that we know an upper-bound b on the utility that any player receives for any outcome in which she breaks the cryptography. Then if a player's probability of breaking the cryptography is ϵ', the total expected utility she gains from deviating is at most $b\dot{\epsilon}'$. Similarly, if a coalition of k players attempts to break the cryptography, their probability of doing so will be at most $k\dot{b}\dot{e}psilon'$.

6.3.4 Dealing with Players with Unexpected Utilities

Just as this protocol allows us to handle coalitions of players, it also allows us to handle a small number of players with unexpected utilities. These players are not irrational, but they may, for example, prefer not to learn the secret, or prefer that only some other player j learn the secret.

Because we now wish for the protocol to be able to tolerate a coalition of k players as well as t players with arbitrary utilities, and because the t players could always choose to send their shares to the coalitions members, it is clear that $t + k$ must replace k in all bounds for the protocol variations described above. The mediator (or SFE) can then be instructed to continue play as long as at least $n - t$ players send

their shares.

We do not describe the details of the results pertaining to players with unexpected utilities. See the original paper [13] for more information.

6.3.5 Limitations of the Abraham, Dolev, Gonen, and Halpern. Protocol

We consider how the limitations of the Abraham et al. protocol relate the limitations of the two protocols presented earlier in this chapter.

1. Dealer presence: this limitation is removed as described above.

2. Equilibrium type: although the type of equilibrium prescribed by the protocol differs from a basic Nash as described above, the problem remains - there are other equilibria in the game that are at least as strong as the equilibrium prescribed.

3. Utility assumptions: this protocol can tolerate a small number of players with unexpected utilities, as described above. The utilities of the remaining players are required to have the same structure as in the earlier protocols.

4. Lack of resistance to coalitions: This protocol is resistant to coalitions of size up to $n - 1$.

5. Prior Awareness of Incentives: The protocol designer still needs to be aware of the incentives in the original protocol in order to properly choose β. In the two-round protocol, awareness of incentives is used to choose the size of the field for the information checking protocol. Finally, some limited awareness of incentives is necessary in order to choose appropriate security parameters for the cryptography in the GMW SFE protocol.

6. Reliance on computational assumptions: The protocol which uses the GMW SFE in order to eliminate the dealer still relies on computational assumptions, as well as on the players being computationally bounded. In contrast to prior

papers, the equilibrium is correctly formalized as an ϵ-Nash. The protocol that uses information checking instead of digital signatures and the BGW SFE is less powerful, but avoids computational assumptions.

Bibliography

[1] M. Ben-Or, S. Goldwasser, and A. Wigderson. Completeness theorems for non-cryptographic fault-tolerant distributed computation. In *Proc. 20th ACM Symp. Theory of Computing*, pages 110, 1988.

[2] O. Goldreich. *Foundations of Cryptography, Vol. 2.* Cambridge University Press, 2004.

[3] O. Goldreich, S. Micali, and A. Wigderson. How to play any mental game. In *Proc. 19th ACM Symp. Theory of Computing*, pages 218229, 1987.

[4] J. Y. Halpern and V. Teague. Rational secret sharing and multiparty computation: extended abstract. In *Proc. 36th ACM Symp. Theory of Computing*, pages 623632, June 2004.

[5] S. Izmalkov, M. Lepinski, and S. Micali. Rational secure computation and ideal mechanism design. In *Proc. 46th IEEE Symp. Foundations of Computer Science*, pages 585595, October 2005.

[6] M. Lepinksi, S. Micali, and A. Shelat. Collusion-free protocols. In *Proc. 37th ACM Symp. Theory of Computing*, pages 543552, 2005.

[7] M. Lepinski, S. Micali, C. Peikert, and A. Shelat. Completely fair SFE and coalition-safe cheap talk. In *Proc. 23rd ACM Symp. Principles of Distributed Computing*, pages 110, July 2004.

[8] M. J. Osborne and A. Rubinstein. *A Course in Game Theory.* MIT Press, Cambridge, Mass., 1994.

[9] T. Rabin and M. Ben-Or. Verifiable secret sharing and multiparty protocols with honest majority. In *Proc. 21st ACM Symp. Theory of Computing*, pages 7385, 1989.

[10] A. Shamir. How to share a secret. *Commun. ACM*, 22(11):612613, 1979.

[11] Y. Shoham and M. Tennenholtz. Non-cooperative computing: Boolean functions with correctness and exclusivity. *Theoretical Computer Science*, 343(12):97113, 2005.

[12] A. Yao. Protocols for secure computation (extended abstract). In *Proc. 23rd IEEE Symp. Foundations of Computer Science*, pages 160164, 1982.

[13] I. Abraham, D. Dolev, R. Gonen, and J. Halpern. Distributed computing meets game theory: Robust mechanisms for rational secret sharing and multiparty computation. In *Proc. 25th ACM PODC*, pages 53-62, July 2006.

[14] D. Gordon and J. Katz. Rational secret sharing, revisited. in *Proc Fifth Conference on Security and Cryptography for Networks*, pages 229-241, July 2006.

[15] A. Lysyanskaya and N. Triandopoulos. Rationality and adversarial behavior in multi-party computation. In *Proc 26th International Cryptology Conference*, pages 180-197, 2006.

[16] G. R. Blakley. Safeguarding cryptographic keys. In *Proc National Computer Conference 48*, pages 313317, 1979.

[17] Oded Goldreich. Two remarks concerning the Goldwasser-Micali-Rivest signature scheme. In *Advances in CryptologyCRYPTO 86*, pages 104110, 1987.

[18] Shafi Goldwasser, Silvio Micali, and Ronald L. Rivest. A digital signature scheme secure against adaptive chosen-message attacks. *SIAM Journal on Computing* pages 281308, April 1988.

[19] Rosario Gennaro, Shai Halevi, and Tal Rabin. Secure hash-and-sign signatures without the random oracle. In *Advances in CryptologyEUROCRYPT*, pages 123139, 1999.

[20] Ronald Cramer and Victor Shoup. Signature schemes based on the strong RSA assumption. In *ACM Transactions on Information and System Security*, pages 161185, 2000.

[21] M. Ben-Or, O. Goldreich, S. Goldwasser, J. Hastad, J. K. S. Micali, and P. Rogaway. Everything provable is provable in zero-knowledge. In *Proc. CRYPTO 88*, pages 3756. Springer Verlag, 1988.