Distributed Computing Meets Game Theory: Robust Mechanisms for Rational Secret Sharing and Multiparty Computation

Ittai Abraham[*]
Hebrew University
ittaia@cs.huji.ac.il

Danny Dolev[†]
Hebrew University
dolev@cs.huji.ac.il

Rica Gonen[‡]
Bell Labs,
Lucent Technologies
gonen@lucent.com

Joe Halpern[§]
Cornell University
halpern@cs.cornell.edu

ABSTRACT

We study *k-resilient Nash equilibria*, joint strategies where no member of a coalition C of size up to k can do better, even if the whole coalition defects. We show that such k-resilient Nash equilibria exist for secret sharing and multiparty computation, provided that players prefer to get the information than not to get it. Our results hold even if there are only 2 players, so we can do multiparty computation with only two rational agents. We extend our results so that they hold even in the presence of up to t players with "unexpected" utilities. Finally, we show that our techniques can be used to simulate games with mediators by games without mediators.

Categories and Subject Descriptors: F.0 [Theory of Computation]: General.

General Terms: Economics, Security, Theory.

Keywords: Distributed Computing, Game Theory, Secret Sharing, Secure Multiparty Computation.

1. INTRODUCTION

Traditionally, work on secret sharing and multiparty computation in the cryptography community, just like work in distributed computation, has divided the agents into "good guys" and "bad guys". The good guys follow the protocol; the bad guys do everything in their power to make sure it does not work. Then results are proved showing that if no more than a certain fraction of the agents are "bad", the protocol will succeed.

Halpern and Teague [10] studied secret sharing under the assumption that agents were *rational*: they would only do what was in their self-interest. For three or more players, under the assumption that a player prefers to get the secret over not getting it, they give a randomized protocol with constant expected running time in which all players learn the secret. They prove their protocol is a Nash equilibrium that survives iterated deletion of weakly dominated strategies.

Indeed, traditional results in game theory mostly consider the equilibrium notions (like Nash equilibrium) that tolerates the deviation of only one agent. That is, a joint strategy $(\sigma_1, \ldots, \sigma_n)$ is a Nash equilibrium if no agent can do better by unilaterally deviating (while all the other agents continue to play their part of the joint strategy). However, in practice, agents can form coalitions. It could well be that if three agents form a coalition and they all deviate from the protocol, then they could all do better.

We define an equilibrium to be *k-resilient* if it tolerates deviations by coalitions of size up to k. Roughly speaking, a joint strategy $(\sigma_1, \ldots, \sigma_n)$ is *k-resilient* if, for any coalition $|C| \leq k$ that deviates from the equilibrium, *none* of the agents in C do better than they do with $(\sigma_1, \ldots, \sigma_n)$. This is a very strong notion of resilience (much stronger than other notions in the literature). We will be interested in *k-resilient practical mechanisms* which, roughly speaking, are protocols that define a k-resilient Nash equilibrium that survives iterated deletion of weakly dominated strategies.

1.1 Our contributions

In this paper we significantly extend and improve the results of Halpern and Teague in several important ways. While continuing to use rationality so as to move away from the tradition "good guys"–"bad guys" adversary model

[*] Part of the work was done while the author visited Microsoft Research Silicon Valley.

[†] Part of the work was done while the author visited Cornell university. The work was funded in part by ISF, NSF, CCR, and AFOSR.

[‡] Supported in part by the IDA.

[§] Supported in part by NSF under grants CCR-0208535, ITR-0325453, and IIS-0534064, by ONR under grant N00014-01-10-511, by the DoD Multidisciplinary University Research Initiative (MURI) program administered by the ONR under grants N00014-01-1-0795 and N00014-04-1-0725, and by AFOSR under grant FA9550-05-1-0055.

PODC'06, July 22-26, 2006, Denver, Colorado, USA.
Copyright 2006 ACM 1-59593-384-0/06/0007 ...$5.00.

that is standard in the distributed computing community, we mitigate the "fragility" of Nash equilibrium by allowing coalitions and tolerating a certain number of agents whose utilities may be unknown or nonstandard. Our specific contributions include the following:

1. While Halpern and Teague's results provide a 1-resilient equilibrium, our main result shows that we can achieve *optimal resilience* — an $(n-1)$-resilient practical mechanism — for the n out of n secret-sharing game. This is of particular interest in the context of secure multiparty computation. Replacing the use of n out of n secret sharing in standard multiparty computation protocols by our $(n-1)$-resilient rational secret sharing protocol, it follows that *any* multiparty computation that can be carried out with a trusted mediator can also be carried out without the trusted mediator, in a highly resilient way.
2. While Halpern and Teague's results are appropriate for three or more players, our results work even for two players. The $n = 2$ setting is of great interest. For example, consider Yao's [25] classic *millionaire's problem*, where two millionaires meet and want to learn which is richer, in a setting where both millionaires would like to learn the answer, but would prefer that the other millionaire does not. We provide the first fair solution to the problem in this setting. On a perhaps more practical note, consider rival companies that wish to securely compute aggregates statistics (e.g., medians or expectations) based on their combined private databases. Malkhi et al. [18] recently built a full distributed implementation of secure computation for two parties. One of the main open questions raised in the context of the real-world applicability of such systems is exactly the problem of fair termination. Our results solve this problem for any number of agents and arbitrary coalitions if agents are rational.
3. Halpern and Teague's randomized protocol includes a parameter that essentially determines the probability of terminating in any given round. To set this parameter, the protocol designer must know the utilities of all the agents (or, at least, a lower bound on the gap between the utility of everyone learning the secret and the utility of no one learning the secret). This is also the case for our general protocol. However, we can show that if $k < \lceil n/3 \rceil$, then we can give a single k-resilient practical mechanism that works for all choices of numerical utility, as long as agents do not strictly prefer not learning the secret to learning the secret. Moreover, this protocol does not require cryptographic assumptions. For the case that $k = 1$, this solves an open problem raised by Halpern and Teague [10].
4. A system designer may not be able to count on all agents having the utility she expects. For example, in our setting, especially if n is large, there may be some "altruists" who actually prefer it if more agents learn the secret. We extend our results to the case where there are t agents whose utility can be arbitrary, as long as the remaining $n - t$ agents have the utility that the designer expects.
5. Finally, our results lead to a deeper understanding of the power of *cheap talk* as a method to securely simulate an equilibrium without depending on honest mediators. In many cases of interest, it can be shown that a mechanism with desired properties exists if there is a trusted mediator. But a reliable trusted mediator is not always available. Izmalkov, Micali, and Lepinski [12] and Lepinski et al. [16] show, roughly speaking, that any equilibrium of a game with a mediator can be simulated by a game without a mediator. However, their construction relies on a very strong primitive they call an *envelope* and *ballot-box*; it is not clear that envelopes and ballot-boxes can be implemented in practice. Ben-Porath [4] shows that we can simulate a Nash equilibrium with a mediator provided that there is a "punishment strategy" which players can use to threaten potential deviators. Heller [11] strengthens Ben-Porath's result to allow coalitions. We show that, if we can assume private channels or if we make a standard cryptographic assumption (that imply oblivious transfer and computationally bounded players), we can simulate any equilibrium of a game with a mediator provided that there is a punishment strategy. We also show that if k is sufficiently small, then we can do the simulation even without the punishment strategy.

Perhaps the most significant issue that we have not yet addressed is asynchrony. All our results depend on the model being synchronous. We are currently working on the asynchronous case.

2. DEFINITIONS

We consider games in extensive form, described by a game tree whose leaves are labeled by the utilities of the players. We assume that at each node in the game tree player i is in some *local state* (intuitively, this local state describes player i's initial information and the messages sent and received by player i). With each run r (i.e., path that ends in a leaf) in the game tree and player i, we can associate i's utility, denoted $u_i(r)$, if that path is played. A *strategy* for player i is a (possibly randomized) function from i's local states to actions. Thus a strategy for player i tells player i what to do at each node in the game tree. A *joint strategy* $\vec{\sigma} = (\sigma_1, \ldots, \sigma_n)$ for the players determines a distribution over paths in the game tree. Let $u_i(\vec{\sigma})$ denote player i's expected utility if $\vec{\sigma}$ is played.

Let $N = \{1, \ldots, n\}$ be the set of players. Let $\mathcal{S}_i$ denote the set of possible strategies for player i, and let $\mathcal{S} = \mathcal{S}_1 \times \ldots \times \mathcal{S}_n$. Given a space $A = A_1 \times \cdots \times A_n$ and a set $I \subset N$ let $A_I = \prod_{i \in I} A_i$ and $A_{-I} = \prod_{i \notin I} A_i$. Thus, $\mathcal{S}_I$ is the strategy set of players in I. Given $x \in A_I$ and $y \in A_{-I}$, let (x, y) be the tuple in A such that $(x, y)_i = x_i$ if $i \in I$ and $(x, y)_i = y_i$ otherwise.

A joint strategy is a Nash equilibrium if no player can gain any advantage by using a different strategy, given that all the other players do not change their strategies. We want to define a notion of *k-resilient equilibrium* that generalizes Nash equilibrium, but allows a coalition of up to k players to change their strategies. There has been a great deal of work on dealing with deviations by coalitions of players, going back to Aumann [2]. Perhaps most relevant to our work is that of Bernheim, Peleg, and Whinston [5]. They define a notion of *coalition-proof Nash equilibrium* that, roughly speaking, attempts to capture the intuition that $\vec{\sigma}$ is a coalition-proof equilibrium if there is no deviation that allows all players to do better. However, they argue that this is too strong a requirement, in that some deviations are not *viable*: they are not immune from further deviations. Thus, they give a rather complicated definition that tries to capture the intuition of a deviation that is immune from further deviations. This work is extended by Moreno and Wooders [19] to allow correlated strategies.

Since we want to prove possibility results, we are willing to consider a notion of equilibrium that is even *stronger* then those considered earlier in the literature.

Definition 1 (k-resilient equilibrium). *Given a nonempty set $C \subseteq N$, $\sigma_C \in \mathcal{S}_C$ is a* group best response for C to $\sigma_{-C} \in \mathcal{S}_{-C}$ *if, for all $\tau_C \in \mathcal{S}_C$ and all $i \in C$, we have*

$$u_i(\sigma_C, \sigma_{-C}) \geq u_i(\tau_C, \sigma_{-C}).$$

A joint strategy $\vec{\sigma} \in S$ is a k-resilient equilibrium if, for all $C \subseteq N$ with $|C| \leq k$, σ_C is a group best response for C to σ_{-C}. A strategy is strongly resilient *if it is k resilient for all $k \leq n-1$.*

Given some desired functionality $\mathcal{F}$, we say that $(\Gamma, \vec{\sigma})$ is a *k-resilient mechanism for $\mathcal{F}$* if $\vec{\sigma}$ is a k-resilient equilibrium of Γ and the outcome of $(\Gamma, \vec{\sigma})$ satisfies $\mathcal{F}$. For example, if $\vec{\sigma}$ is a k-resilient mechanism for secret sharing, then in all runs of $\vec{\sigma}$, everyone would get the secret.

Observe that a 1-resilient equilibrium is just a Nash equilibrium; thus, this notion generalizes Nash equilibrium. The notion of k-resilience strengthens Moreno and Wooder's notion of coalition-proof equilibrium by tolerating arbitrary deviations, not just ones that are viable (in that the deviations themselves are immune to further deviations). Moreover, k-resilience implies tolerance of deviations where only a single player in the coalition does better; coalition-proof equilibria tolerate only deviations where everyone in the coalition does better. By considering deviations where only one player does better, we allow situations where one player effectively controls the coalition. This can happen in practice in a network if one player can "hijack" a number of nodes in the network. It could also happen if one player can threaten others, or does so much better as a result of the deviation that he persuades other players to go along, perhaps by the promise of side payments. [1] We do restrict to coalitions of size at most k, where k is a parameter. Such a restriction seems reasonable; it is difficult in practice to form and coordinate large coalitions.

We take a *mechanism* to be a pair $(\Gamma, \vec{\sigma})$ consisting of a game and a joint strategy for that game. Intuitively, a mechanism designer designs the game Γ and recommends that player i follow σ_i in that game. The expectation is that a "good" outcome will arise if all the players play the recommended strategy in the game. Designing a mechanism essentially amounts to designing a protocol; the recommended strategy is the protocol, and the game is defined by all possible deviations from the protocol. $(\Gamma, \vec{\sigma})$ is a *practical mechanism* if $\vec{\sigma}$ is a Nash equilibrium of the game Γ that survives iterated deletion of weakly-dominated strategies.[2] Similarly a *k-resilient practical mechanism* is a practical mechanism where $\vec{\sigma}$ is k-resilient.

3. GAMES WITH MEDIATORS

To prove our results, it is useful to consider games with mediators. We can think of a mediator as a trusted party. We will show that if there is an equilibrium in a game using a mediator, then we can get a similar equilibrium even without the mediator, provided utilities satisfy certain properties.

Following Forges [6], we view a mediator as a communication device. Consider a multistage game with T stages with complete recall. Formally, a *mediator* is a tuple $\langle \mathcal{I}_i^t, \mathcal{O}_i^t, \mathcal{P}^t \rangle$ for $i \in N$ and $t \in T$, where $\mathcal{I}_i^t$ is a set of inputs that player i can send at stage t, $\mathcal{O}_i^t$ is a set of outputs for player i at stage t, and $\mathcal{P}^t$ is a function from $\prod_{i \in N, r \leq t} \mathcal{I}_i^t$ and (possibly some random bits r) to $\prod_{i \in N} \mathcal{O}_i^t$. Less formally, at each stage t, each player sends a message (input) and the mediator computes a function (which is possibly random) of all the messages ever sent and sends each player a piece of advice (output).

Given a multistage game Γ and a mediator d, we can define a game Γ_d, the extension of Γ with d. Each *stage* t of Γ_d can consists of three *phases*: in the first phase, each player, i, sends some input in $\mathcal{I}_i^t$ to d; in the second phase, d sends each player i the appropriate output in $\mathcal{O}_i^t$ according to $\mathcal{P}^t$; and in the third phase, each player makes a move in Γ or no move at all. The utilities of the players depend only on the moves made in Γ.

4. RESILIENT SECRET SHARING

In this section we focus on a specific game: m out of n secret sharing based on Shamir's scheme [23]. The secret is $f(0)$, where f is a degree $m-1$ (random) polynomial over a field F such that $|F| > n$. Agent i is given $f(i)$, for $i = 1, \ldots, n$; $f(i)$ is agent i's "share" of the secret. We assume, without loss of generality, that $f(0) \neq 0$ (and that this fact is common knowledge among the players). For ease of exposition, we assume that the secret is equally likely to be any nonzero field value (and that this fact is common knowledge among the players).

For most of this section we assume for ease of exposition that the initial shares given to each player are *signed* by the issuer, in such a way that each other player can verify the signature of the issuer and no player can forge the signature. We remove the assumption at the end of the section.

To prove our results, we must formalize the intuition that players prefer getting the secret to not getting it. Halpern and Teague [10] did this by assuming that it was possible to determine whether or not a player learned a secret just by looking at a run (complete path) in a game tree. To avoid this assumption, we assume that players must output the secret (or their best guess at the secret) at the end of the game. Of course, if they guess wrong, the output will be incorrect. This approach has the additional advantage that we can model situations where players get partial information about the secret (so that they can guess the value with high probability).

Given a run r in the game tree, let $out(r)$ be a tuple where $out_i(r) = 1$ if player i correctly outputs the secret and $out_i(r) = 0$ if player i outputs a value that is not $f(0)$. We can now model the fact that players prefer to get the secret to not getting it using two assumptions that are identical to those made by Halpern and Teague, except that we talk about what a player outputs rather than what a player learns. The following assumption says that player i's utility

[1] Of course, in a more refined model of the game that took the threats or side-payments into account, everyone in the coalition would do better.

[2] We assume the reader is familiar with the concept of iterated deletion; see [21, 10] for details.

depends just on the outputs:

U1. If $out(r) = out(r')$ then $u_i(r) = u_i(r')$.

The next assumption is that each player strictly prefers learning the secret to not learning it.

U2. If $out_i(r) = 1$ and $out_i(r') = 0$ then $u_i(r) > u_i(r')$.

In some of our results we require a weaker assumption, namely, that each player never prefers not to learn the secret.

U2$'$. If $out_i(r) = 1$ and $out_i(r') = 0$ then $u_i(r) \geq u_i(r')$.

Halpern and Teague [10] made an additional assumption that was needed for their impossibility result, which captures the intuition that players prefer that as few as possible of the other players will learn the secret. This property was not used by Halpern and Teague for their possibility result. We do not need it for ours either, so we do not make it here.

We provide a strategy for an augmented game with a mediator that has a k-resilient equilibrium where all agents learn the secret, and then show how this strategy can be implemented without a mediator in a way that survives iterated deletion.

Suppose that the game is augmented with a mediator that uses the following protocol.

- In stage $t \geq 0$, the mediator waits to receive an appropriate phase 1 message from each player. In stage 0, this will be a share in polynomial f correctly signed by the issuer; in stage $t > 0$, this will be an *ack* message. If it does not receive an appropriate message from each player, the mediator stops playing. If the mediator is still playing at stage t, then after receiving a phase 1 (of stage t) message from each player, it chooses a binary random variable c^t with $Pr[c^t = 1] = \alpha$ (we specify α below) and a random degree $m-1$ polynomial g^t over F such that $g^t(0) = 0$. It computes the polynomial $h^t = f \cdot c^t + g^t$ and, in phase 2, sends each player i the value $h^t(i)$. Note that if $c^t = 0$, then $h^t(0) = 0$; if $c^t = 1$, $h^t(0) = f(0)$. Thus, if $c^t = 1$, h^t encodes the same secret as f; if $c^t = 0$, then h^t does not encode a secret (and if the players get all the shares of h^t, they will realize that h^t does not encode a secret, since $f(0) \neq 0$, by assumption).

Consider the following strategy σ_i for player i.

1. In phase 1 of stage 0, send your share of the secret to the mediator and set *ok* := *true*. In phase 1 of stage $t > 0$, if *ok* = *true*, send *ack* to the mediator.
2. If the mediator sends the message a_i^t in phase 2 of stage $t \geq 0$ and *ok* = *true*, then in phase 3 of stage t send a_i^t to all the other players. If not all players sent a message in phase 3, set *ok* := *false*. Otherwise, construct the polynomial h^t by interpolating the received shares. If there is no such polynomial, set *ok* := *false*. Otherwise, if $h^t(0) \neq 0$, then *stop*, set *ok* := *false*, and output $h^t(0)$. If $h^t(0) = 0$, continue to the next stage.

We want to show that $(\sigma_1, \ldots, \sigma_n)$ is a k-resilient equilibrium, provided that α is chosen appropriately. If $A \subseteq N$, let $u_i(A)$ be i's best-case expected utility if exactly the players in A learn the secret. To understand why we say "best-case expected utility" here, note that if the players in A learn the secret, we assume that their output is the secret. But the utility of player i also depends on what the players not in A output. Since the players not in A have no information about the secret, the probability that a player not in A will guess the secret correctly is $1/(|F|-1)$. However, this probability alone does not determine i's expected payoff, since the payoff may depend in part on how the players not in A correlate their guesses. For example, if the players not in A agree to all guess the same value, then with probability $1/(|F|-1)$ they all guess the secret, and with probability $(|F|-2)/(|F|-1)$ none do. On the other hand, if they always make different guesses, then with probability $(n-|A|)/(|F|-1)$, exactly one player not in A guesses the secret, and with probability $(|F|-1-n+|A|)/(|F|-1)$, no player in A guesses the secret. Let $u_i(A)$ be i's payoff if the players not in A correlate their guesses in a way that gives i the best expected payoff, and let $m_i = \max_{A \subseteq N} u_i(A)$.

Proposition 1. *If* $\alpha \leq \min_{i \in N} \frac{u_i(N) - u_i(\emptyset)}{m_i - u_i(\emptyset)}$ *and* $k < m$, *then* $(\sigma_1, \ldots, \sigma_n)$ *is a k-resilient practical mechanism for m out of n secret sharing that has expected running time* $O(1/\alpha)$.

Proof. Clearly if everyone follows the strategy, then, with probability 1, everyone will eventually learn (and thus output) the secret, and this will happen in expected time $O(1/\alpha)$. Consider what happens if some coalition $C \subseteq N$ of players does not follow the strategy. This could happen if either (1) some players in C lie about their initial share in their phase 1 message to the mediator or do not send a message at all, (2) some players in C do not send an *ack* message to the mediator in phase 1 of some round $t > 0$, or (3) some players in C either lie or do not send their shares (received from the mediator in phase 2) to the other players in phase 3 of some round t.

Since we have assumed that shares are signed by the issuer, if some $i \in C$ lies about his initial share, then i will be caught and the mediator will not continue. This clearly results in a worse outcome for i. Similarly, if some $i \in C$ does not send a share in stage 0 or does not send an *ack* message in stage $t > 0$, then the game stops and no player learns the secret. This is also a worse outcome for i.

Now suppose that some $i \in C$ does not send a share in phase 3 of round t. With probability $1-\alpha$, the game will stop and no player will learn the secret. If the game does not stop (which happens with probability α), the best outcome player i can hope for is to gain m_i. Thus, $i \in C$ gains from sending nothing only if $\alpha m_i + (1-\alpha)u_i(\emptyset) > u_i(N)$. But we have assumed that this is not the case. Thus, i will send something at phase 3.

Now suppose that players in C decide to send $X = \{x_i \mid i \in C\}$ instead of $\{h^t(i) \mid i \in C\}$. We say that X is a *compatible response* if and only if there exists a degree $m-1$ polynomial q such that

$$q(i) = \begin{cases} x_i - h^t(i) & \text{if } i \in C \\ 0 & \text{if } i \in \{0, 1, \ldots, n\} - C. \end{cases}$$

Note that this condition can be checked by simple interpolation. There are two cases to consider. If X is a compatible response, then sending X has the same result as sending $\{h^t(i) \mid i \in C\}$. Essentially, sending a compatible response X at stage t will cause the players to reconstruct the polynomial $h^t + q$ instead of polynomial h^t. However, since $[h^t + q](0) = h^t(0) + q(0) = h^t(0)$, this does not change the outcome, i.e. either with probability α all players learn the secret or with probability $1 - \alpha$ all players interpolate 0 as the secret and continue to the next stage. On the other hand, if X is not a compatible response, then with probability α player $i \in C$ learns the secret and gains utility at most m_i, and with probability $1 - \alpha$ player i does not learn the secret and the game ends, since other players will attempt to interpolate $[h^t + q]$. This attempt will either fail or will result in some $[h^t + q](0) \neq 0$. Thus, $i \in C$ gains from sending a incompatible response only if $\alpha m_i + (1 - \alpha)u_i(\emptyset) > u_i(N)$, which we assume is not the case.

Finally, the fact the $\vec{\sigma}$ survives iterated deletion is immediate from the sufficient condition given by Halpern and Teague [10, Theorem 3.2] for a randomized protocol with unbounded running time to survive the iterated deletion process. □

If in the protocol above we use n out of n secret sharing (that is, if the mediator chooses a polynomial of degree $n - 1 = m - 1$), then we have a strongly resilient protocol, provided that α is chosen appropriately. That is, we can essentially tolerate arbitrary coalitions.

Note that the choice of α here depends on the utilities of the agents. Implicitly, we are assuming that the mediator knows these utilities, so that it can choose α appropriately. Moreover, the expected running time of the protocol depends on α. We now show that if $k < m \leq n - k$, then we can modify the mediator's algorithm slightly so that the expected running time is 2 rounds. However, the modification still depends on the mediator knowing the players' utilities. We then show that if $k < m \leq n - 2k$, then it is not even necessary for the mediator to know the utilities at all.

The key observation is that if $k \leq n-m$ (i.e., if $m \leq n-k$), then the players have sufficiently many shares that they can reconstruct the secret even if the coalition members do not send their shares in phase 3. If the coalition members actually send incorrect values of a_i^t rather than no value at all, then the non-coalition members may have a problem. They will realize that there is no polynomial that interpolates the values that were sent; moreover, they will know that there exist $n - k \geq m$ "good" values, all of which lie on the correct polynomial h^t. However, there may be multiple subsets of size m through which different polynomials can be interpolated. This problem can be avoided if the mediator sends each player some information they can use to verify the truth of the other player's statements. The verification process uses Rabin and Ben-Or's [22] *information checking protocol* (ICP).

We modify the mediator's protocol as follows:

- F (the field from which the coefficients of the polynomial are taken) is chosen such that $|F| > 1/\beta$, where β is a security parameter determined below.
- c is chosen so that $\Pr(c = 1) = 1/2$, rather than α.
- In addition to sending each player i the value $h^t(i)$ at round t, it also sends i a random element y_{ij}^t in F, one for each player $j \neq i$. Finally, it sends j a pair (b_{ij}^t, c_{ij}^t) of field elements such that $c_{ij}^t = b_{ij}^t h^t(i) + y_{ij}^t$ (so that it sends i (b_{ji}^t, c_{ji}^t) for all $j \neq i$).

We modify σ_i to the protocol σ_i' where i sends j the value y_{ij}^t in addition to $h^t(i)$. Note that if i modifies the value of $h^t(i)$ to h', i must also modify the value of y_{ij}^t to y' such that $c_{ij}^t = b_{ij}^t h' + y'$; otherwise, i's lie will be caught. The probability of being able to guess an appropriate y' is less than β.

Proposition 2. *If* $\beta < \max_{i \in N} \frac{u_i(N) - u_i(\emptyset)}{2m_i - (u_i(N) + u_i(\emptyset))}$ *and* $k < m \leq n - k$, *then* $(\sigma_1', \ldots, \sigma_n')$ *is a* k*-resilient practical mechanism for* m *out of* n *secret sharing that has expected running time* 2.

Proof. Clearly if everyone follows the strategy, then, with probability 1, everyone will eventually learn the secret, and this will happen in expected time 2. The argument for k-resiliency proceeds much as that for Proposition 1. But note that in the case that $c = 1$, the non-coalition players will get the secret unless some coalition members lie about their shares and the lie is not detected. The probability that a single lie is not detected is at most β; if there are more lies, the probability of detection is even greater. Thus, if $c = 1$ and someone in the coalition lies, with probability at most β, i's payoff will be at most m_i, and with probability at least $1 - \beta$, i's payoff will be $u_i(N)$. If $c = 0$ and someone in the coalition lies, then with probability at least $1 - \beta$, the lie will be detected, and the game will end. If the lie is not detected, the most that i can hope to get is m_i. Thus, i's expected utility from being part of a coalition where someone lies is at most $\frac{1}{2}(1 - \beta)u_i(\emptyset) + \frac{1}{2}(1 - \beta)u_i(N) + \beta m_i$. So i would prefer it's coalition to lie only if

$$\frac{1}{2}(1 - \beta)u_i(\emptyset) + \frac{1}{2}(1 - \beta)u_i(N) + \beta m_i > u_i(N).$$

□

Note that if $k < \lceil n/2 \rceil$, then we can choose m such that $k < m \leq n - k$, so that Proposition 2 applies. Moreover, the proof of Proposition 2 shows that we could have taken $\Pr(c = 1) = 1/(1 + \epsilon)$ for any $\epsilon > 0$ by appropriately modifying β (i.e., by taking the field F sufficiently large), giving an expected running time of $1 + \epsilon$ for the protocol.

If $k < m \leq n - 2k$, then we can do even better. In this case, even if everyone in the coalition lies, the non-coalition members can use Reed-Solomon unique decoding to find the correct polynomial in time almost linear in n [13]; we do not need to send verification information. Thus, we consider the mediator's initial protocol (except that $\Pr(c = 1) = 1/2$) and use the original protocol σ_i for player i.

Proposition 3. *If* $k < m \leq n - 2k$, *then* $(\sigma_1, \ldots, \sigma_n)$ *is a* k*-resilient practical mechanism for any* m *out of* n *secret sharing game that has expected running time* 2.

Note that if $k < \lceil n/3 \rceil$, then we can choose $m = k + 1$ so that Proposition 2 applies. And again, we can take $\Pr(c = 1) = 1/(1 + \epsilon)$ for all $\epsilon > 0$, to get a protocol with expected running time $1 + \epsilon$. In fact, $(\sigma_1, \ldots, \sigma_n)$ is a k-resilient Nash equilibrium even if $\Pr(c = 1) = 1$, that is, if the mediator sends player i $f(i)$ in stage 0. By results of Halpern and Teague, however, this protocol does not survive iterated deletion. To get a practical mechanism, we must take $\Pr(c - 1) < 1$.

There are many games with mediators that can deal with secret sharing. For example, the mediator can just send the secret to each of the players. The advantage of the game that we have just described is that it can be simulated by the players without a mediator using multiparty computation. The simulation proceeds as follows.

Assume again that the players are given signed shares. We actually have a sequence of multiparty computations, one for each round t. For round t, the input of player i to the multiparty computation consists of the following inputs, where a and b are parameters of the protocol chosen so that $a/2^b \leq \min_{i \in N} \frac{u_i(N)-u_i(\emptyset)}{m_i-u_i(\emptyset)}$ (intuitively, $a/2^b$ is playing the role of α):

1. the (signed) secret share that i received initially;
2. a random bitstring c_i^t of length b, uniformly distributed in $\{0, 1, \dots, 2^b\}$;
3. a random degree $m-1$ polynomial g_i^t such that $g_i^t(0) = 0$ and the coefficients of g_i^t are uniformly distributed in the underlying field F.
4. a random bitstring b_i of length $\lceil \log(|F|) \rceil$.

Let

$$c^t = \begin{cases} 1 & \text{if } \oplus_{i \in N} c_i^t \leq a2^z \\ 0 & \text{otherwise,} \end{cases}$$

where $\oplus$ denotes bitwise exclusive or; let $g^t = (g_1^t + \cdots + g_n^t)$.

The multiparty computation then does essentially what the mediator does in the previous algorithm. More precisely, let F^t be the function that, given the inputs above, does the following computation. It checks that the shares sent are correctly signed; if so, it computes the polynomial f that interpolates them. It also checks that $g_i^t(0) = 0$ for each random polynomial. If both of these properties hold, then let $h^t = c^t \cdot f + g^t$, as before. The output of F^t is then $(h^t(1) \oplus b_1, \dots, h^t(n) \oplus b_n)$.

There is a circuit that computes F^t. As shown by Goldreich, Micali, and Wigderson [8] (GMW from now on), it is possible to arrange it that the agents have a share in each node of the circuit. Then, at the end of the computation, the players each have a share of the output of F^t. Using secret sharing, they can then learn the output. Upon learning the output, since player i knows b_i, player i will be able to compute $h^t(i)$; since b_i^t is random, no other player will be able to compute $h^t(i)$. After computing $h^t(i)$, player i sends an *ack* to the other players. If player i gets an *ack* from all the other players, it shares $h^t(i)$, just as in σ_i.

A coalition member i will follow the protocol during the circuit evaluation of F for the same reasons that i follows it when playing with a mediator: deviation will result either in i being caught (if i does not send the correct signed share, does not send a share to some or to all, or sends an incompatible response) or will not affect the outcome (if i sends a compatible response) or will cause some players to obtain the wrong final shares and may cause all players to learn the wrong secret (if i sends an incorrect value during the circuit evaluation). The argument that players will share their secrets (i.e., that player i will send all the other players $h^t(i)$) gives a k-resilient equilibrium proceeds just as before.

Thus, coalition members can be viewed as what is known as "honest-but-curious" or *passive* adversaries. As is well known, we can do multiparty computation with such adversaries without cryptographic techniques if $k < n/2$ [7, 3]; on the other hand, if $n/2 \leq k < n$, it seems we must use the techniques of GMW [8], which use cryptography and thus our results depend on the existence of oblivious transfer and computationally bounded players. (Of course, the assumption that the issuer signs all the shares with an unforgeable signature also requires cryptography; we show how to get rid of this assumption later.) The exact cryptographic assumptions we need depend in part on the protocol we consider. Since we do not wish to focus here on issues of cryptography, we say *assuming cryptography* to indicate that we assume players are computationally bounded and that *enhanced trapdoor permutations* [7] exist.[3]

If we use cryptography (even given the assumptions above), we typically cannot perfectly implement a desired functionality $\mathcal{F}$, since, for example, there is a small probability that the cryptography will be broken. For our protocols that use cryptography, we can typically make no guarantee as to what happens if the cryptography is broken. Thus, we need to weaken the notion of k-resilient mechanism for $\mathcal{F}$ somewhat to capture this small probability of "error".

Definition 2. $(\Gamma, \vec{\sigma})$ *is a* ϵ-k-resilient mechanism for $\mathcal{F}$ *if $\vec{\sigma}$ satisfies $\mathcal{F}$ but, for each coalition C such that $|C| \leq k$, σ_C is not necessarily a group best response to σ_{-C}, but it is a ϵ-best response: no one in the group can do more than ϵ better than they would using a best response. That is, for all C such that $|C| \leq k$, for all $\tau_C \in \mathcal{S}_C$, and all $i \in C$, we have*

$$u_i(\sigma_C, \sigma_{-C}) \geq u_i(\tau_C, \sigma_{-C}) - \epsilon.$$

Intuitively, if ϵ is small, although players may be able to do better, it may not be worth their while to figure out how. A ϵ-1-resilient equilibrium is an ϵ-*Nash equilibrium* [21].

Theorem 1. *Consider the m out of n secret sharing game. Suppose that players' utilities satisfy U1.*

(a) If $k < m \leq n - 2k$, then assuming U2′, there exists a practical k-resilient mechanism without a mediator for m out of n secret sharing with an expected running time of 2 rounds.

(b) If $k < m \leq n - k$, then, assuming U2, there exists a mechanism without a mediator that takes a parameter β and is a practical k-resilient mechanism for m out of n secret sharing with an expected running time of 2 rounds if $\beta < \max_{i \in N} \frac{u_i(N)-u_i(\emptyset)}{2m_i-(u_i(N)+u_i(\emptyset))}$.

(c) If $k < m \leq n$, then, assuming U2 and cryptography, for all $\epsilon > 0$, there exists a mechanism without a mediator that takes parameter α and is a practical ϵ-k-resilient mechanism for m out of n secret sharing with an expected running time of $O(1/\alpha)$ rounds if $\alpha \leq \min_{i \in N} \frac{u_i(N)-u_i(\emptyset)}{m_i-u_i(\emptyset)}$.

Note that Theorem 1(c) applies if $m = n = 2$ and $k = 1$; that is, for all $\epsilon > 0$, there is a practical ϵ-1-resilient mechanism for 2 out of 2 secret sharing (assuming cryptography). Halpern and Teague [10, Cor. 2.2] claim that there is no practical mechanism for 2 out of 2 secret sharing. Their argument also suggests that there is no ϵ-1 resilient mechanism for secret sharing. It seems that this corollary is

[3] Enhanced trapdoor permutations are a set of permutations such that, given a permutation f_α in the set and an element x generated using an algorithm S, it is hard to compute $f_\alpha^{-1}(x)$ even if the random coins used by S to generate x are known. If enhanced trapdoor permutations exist, then it is possible to do oblivious transfer and to sign secrets with signatures that cannot be forged.

false. Halpern and Teague show correctly that a strategy in what they call $\mathcal{A}^2$, where a player sends his share to another player, will be eliminated by the iterated deletion process. The argument for Corollary 2.2 implicitly assumes that if there is an equilibrium in the 2 out of 2 case, where a player learns the secret, then one of the strategies must be in $\mathcal{A}^2$, that is, a player must realize that he is sending his share to the other player. But our protocol has the property that a player does not know when he is sending a "useful" share to the other player (where a share is useful if $\alpha = 1$).[4] We remark that Gordon and Katz [9] also describe a mechanism 1-ϵ mechanism for 2 out of 2 secret sharing (although they do not explicitly mention the need for ϵ).

It is also worth noting that, unlike the protocol given by Halpern and Teague, our protocol has the property that the secret issuer has to issue the shares only once, at the beginning of the protocol. Gordon and Katz [9] and Lysyanskaya and Triandopoulos [17] also describe protocols for rational secret sharing with this property.

Up to now we have assumed that initial shares given to each player are signed by the issuer using unforgeable signatures. This means that a rational player will always send his true shares to the mediator at stage 0. We can remove this assumption using ideas from check vectors and the ICP protocol [3], much as in the mechanism for the case that $m \leq n - k$. With a mediator, the issuer sends the mediator the pair (b_i, c_i) so it can verify i's signature; without a mediator, the issuer sends a different verification pair (b_{ij}, c_{ij}) to each agent $j \neq i$. Using verification in this way, the probability of a lie going undetected is $1/|F|$. We can modify the bounds on α and in Proposition 1 and Theorem 1(c) and on β in Proposition 2 and Theorem 1(b) to take this into account; we leave details to the reader. Note that if $k < m \leq n - 2k$ (that is, in the situation of Proposition 3 and Theorem 1(a)), we do not need to verify the signatures at all. We have a k-resilient equilibrium using the same arguments as before.

As observed by Halpern and Teague [10], the results for secret sharing carry over to multiparty computation. We can give a k-resilient practical mechanism for multiparty computation by doing the circuit evaluation for f and then using the rational secret sharing protocol from Theorem 1, where we choose the optimal m. Thus, for example, if $k < n/3$, then by taking $m = \lceil n/3 \rceil$, we have $k < m \leq n - 2k$, so Theorem 1(a) applies. Similarly, if $k < n/2$, then Theorem 1(b) applies; Theorem 1(c) applies for all k, where now $u_i(A)$ is i's best-case utility if exactly the agents in A learn the function value.

There is only one caveat: we can only do multiparty computation if rational players (with the same utilities) can compute the function f using a trusted mediator. This is a nontrivial requirement. As shown by Shoham and Tennenholtz [24], functions like parity cannot be computed by rational players, even with a trusted mediator. (A player will lie about his value, so that everyone will get the wrong value of parity, but the lying player can reconstruct the right value.) Thus, we get the following result, which improves Theorem 4.2 of [10], where now a "round" is taken to be the number of steps needed to simulate the computation of the circuit.

In the context of multiparty computation, we take $out_i(r)$ to be 1 if player i correctly outputs the function value given the inputs in r, and 0 otherwise; with this change, U1, U2, and U2$'$ are well defined.

THEOREM 2. *Suppose that players' utilities satisfy U1 and the players can compute f with a trusted mediator.*

(a) If $3k < n$, then, assuming U2$'$, there exists a practical k-resilient mechanism without a mediator for the multiparty computation of f with an expected running time of 2 rounds.

(b) If $2k < n$, then assuming U2, there exists a mechanism without a mediator that takes a parameter β and is a practical k-resilient mechanism for the multiparty computation of f with an expected running time of 2 rounds if $\beta < \max_{i \in N} \frac{u_i(N) - u_i(\emptyset)}{2m_i - (u_i(N) + u_i(\emptyset))}$.

(c) If $k < n$, then, assuming U2 and cryptography, for all $\epsilon > 0$, there exists a practical ϵ-k-resilient mechanism for the multiparty computation of f with an expected running time of $O(1/\alpha)$ rounds if $\alpha \leq \min_{i \in N} \frac{u_i(N) - u_i(\emptyset)}{m_i - u_i(\emptyset)}$.

5. TOLERATING PLAYERS WITH "UNEXPECTED" UTILITIES

In Theorem 1, as well as Propositions 1, 2, and 3, we assumed that all players have utilities that satisfy U1 and U2. However, in large systems, it seems almost invariably the case that there will be some fraction of users who do not respond to incentives the way we expect. (Certainly this seems to be the case with students in large undergraduate classes!) This is an issue that arises in practice. For example, in a peer-to-peer network like Kazaa or Gnutella, it would seem that no rational agent should share files. Whether or not you can get a file depends only on whether other people share files; on the other hand, it seems that there are disincentives for sharing (the possibility of lawsuits, use of bandwidth, etc.). Nevertheless, people do share files. However, studies of the Gnutella network have shown almost 70 percent of users share no files and nearly 50 percent of responses are from the top 1 percent of sharing hosts [1].

One reason that people might not respond as we expect is that they have utilities that are different from those we expect. In the Kazaa example, it may be the case that some users derive pleasure from knowing that they are the ones providing files for everyone else. In a computer network, inappropriate responses may be due to faulty computers or faulty communication links. Or, indeed, users may simply be irrational. Whatever the reason, it seems important to design protocols that tolerate such unanticipated behavior, so that the payoffs of the users with "standard" utilities do not get affected by the nonstandard players using different strategy. This observation motivates the next definition.

DEFINITION 3. *A joint strategy $\vec{\sigma} \in \mathcal{S}$ is a* t-immune *if, for all $T \subseteq N$ with $|T| \leq t$, all $\vec{\tau}_C \in \mathcal{S}_T$, and all $i \notin T$, we have $u_i(\vec{\sigma}_{-T}, \vec{\tau}_T) \geq u_i(\vec{\sigma})$.*

An equivalent reformulation of t-immunity in a game Γ can be obtained by considering a variant of Γ where nature moves first, chooses some arbitrary subset of up to t players, and changes their utilities arbitrarily. This reformulation has the advantage of allowing a more refined analysis

[4] Note that Theorem 1 does not contradict any other claims of Halpern and Teague. In particular, the claim that there is no mechanism with a fixed upper bound on its running time for secret sharing (or multiparty computation) holds.

of deviations. Specifically, the set of possible changes to the utility functions can be restricted to some fixed set of deviations. For example, when analyzing a Kazaa-like network, we could consider a game where nature can only change the utilities of agents so that they are either "standard" or altruistic (and so get pleasure out of sharing). We remark that this idea of modifying utilities reappears in some of our theorems (cf. the notion of utility variant defined below).

The notion of t-immunity and k-resilience address different concerns. For t immunity, we consider the payoffs of the players not in C; for resilience, we consider the payoffs of players in C. It is natural to combine both notions.

Definition 4. *A joint strategy $\vec{\sigma} \in \mathcal{S}$ is* (k,t)-robust *if for all $C, T \subseteq N$ such that $C \cap T = \emptyset$, $|C| \leq k$, and $|T| \leq t$, for all $\vec{\tau}_T \in \mathcal{S}_T$, for all $\vec{\phi}_C \in \mathcal{S}_C$, for all $i \in C$ we have $u_i(\vec{\sigma}_{-T}, \vec{\tau}_T) \geq u_i(\vec{\sigma}_{N-(C\cup T)}, \vec{\phi}_C, \vec{\tau}_T)$.*

Intuitively, this says that, for all C such that $|C| \leq k$, σ_C continues to be a best response no matter what strategy τ_T the players in a subset T of size t use, as long as the remaining players (i.e., those in $N - (C \cup T)$), continue to use their component of $\vec{\sigma}$). Note that a k-resilient strategy is a $(k,0)$-robust strategy and a t-immune strategy is a $(0,t)$-robust strategy.[5] We can define ϵ-(k,t)-robust equilibrium mechanisms in the obvious way; we leave details to the reader.

A modification of our earlier techniques gives a generalization of Theorem 1. Roughly speaking, we replace the condition $k < m$ in Theorem 1 by $t + k < m$, since it is always possible that the t players with unexpected utilities send their shares to the k coalition members. We also replace expressions of the form $m \leq n-p$ (where p is either k or $2k$) by $m \leq n - p - 2t$, to allow Reed-Solomon unique decoding if the t players send inappropriate values of the polynomial. Assuming cryptography, we can replace the $m \leq n$ bound with $m \leq n - t$ by using the GMW compiler [8]. Finally, we replace the $m \leq n - k$ bound with $m \leq n - (t + k)$ by using the GMW verifiable secret sharing compiler [8].

When doing multiparty computation in a setting where up to t players can send arbitrary values, we must assume that getting a value that in some sense is close to the true function value is good enough. For example, if we are interested in computing a statistical function such as mean or median, while the exact answer will depend on the values sent by everyone, having a small number of incorrect values will not greatly affect the true value. Indeed, to the extent that we use multiparty computation to compute statistical functions of large datasets, we must allow for some corruption of the inputs. The computed function may contain built-in filters to exclude out-of-range values, and thus limit the influence of inappropriate inputs. Given an input vector $\vec{x}$, define a *t-variant* of $\vec{x}$ to be a vector $\vec{y}$ such that $|\{i : x_i \neq y_i\}| \leq t$. Let U0($t$) be the condition that, if the input of r is $\vec{x}$, then $out_i(r) = 1$ iff the output of i in r is $f(\vec{y})$, where $\vec{y}$ is a t-variant of $\vec{x}$; otherwise, $out_i(r) = 0$. Intuitively, this says an output is acceptable iff it is obtained by applying f to an input that is "close" to the true input.[6]

We can then extend our secret sharing results to multiparty computation, in the same way we got Theorem 2 from Theorem 1. (Theorem 2 is the special case where $t = 0$.)

Theorem 3. *Suppose that can compute f with a trusted mediator and their utilities satisfy U0(t) and U1.*

(a) If $3(t + k) < n$, then, assuming U2′, there exists a practical (k,t)-robust mechanism without a mediator for the multiparty computation of f with an expected running time of 2 rounds.

(b) If $3t+2k < n$, then, assuming U2, there exists a mechanism without a mediator that takes a parameter β and is a practical (k,t)-robust mechanism for the multiparty computation of f with an expected running time of 2 rounds if $\beta < \max_{i\in N} \frac{u_i(N)-u_i(\emptyset)}{2m_i-(u_i(N)+u_i(\emptyset))}$.

(c) If $2(t + k) < n$, then, assuming U2′ and cryptography, for all $\epsilon > 0$, there exists a practical ϵ-(k,t)-robust mechanism for multiparty computation with an expected running time of 2 rounds.

(d) If $2t+k < n$, then, assuming U2 and cryptography, for all $\epsilon > 0$, there exists a practical ϵ-(k,t)-robust mechanism for multiparty computation with an expected running time of $O(1/\alpha)$ rounds if $\alpha \leq \min_{i\in N} \frac{u_i(N)-u_i(\emptyset)}{m_i-u_i(\emptyset)}$.

Proof Sketch. For parts (a) and (b), the secret sharing stage of protocol σ'' is similar to the protocol in Theorem 1, except that we now simulate a mediator that continues playing as long as it gets at least $n-t$ signed shares, and players in later rounds continue if they can interpolate a polynomial h through $n - t$ values (which can be checked using Reed-Solomon decoding) such that $h(0) = 0$. For parts (c) and (d), protocol σ'' compiles protocol σ using the GMW compiler [8]. For part (c) verifiable secret sharing compiler is used [8]. This protects against t arbitrary players (even if they are malicious). □

In all our results that assume cryptography and obtain ϵ-(k,t)-robust mechanisms, the security parameter used in the cryptographic tools is a function of ϵ and the players utilities. Lysyanskaya and Triandopoulos [17] independently obtained a result somewhat like Theorem 3(c) for the special case that $k = 1$. Their result seems to hold without requiring knowledge of the utilities (like Theorem 3(a)), but this is because their notion of "ϵ-error" only requires that σ_C be an ϵ-best response with probability $1 - \epsilon$; on a set of runs of probability ϵ, the difference between the utility using σ_C and the optimal response can be arbitrary.

6. SIMULATING COMMUNICATION EQUILIBRIUM VIA CHEAP TALK

There are many situations where having a mediator makes it possible to find a better or simpler solution for a game. We have seen this with multiparty computation. In real life settings, mediators clearly help with negotiation, for example. This leads to an obvious question: when is it the case that we can simulate a mediator, so that if there is a solution with a mediator, there is a solution without one. This question has been studied in both the economics literature

[5] In fact, our results hold for an even stronger form of robustness: the payoffs of the agents not in T can be taken to be independent of the actions of the agents in T.

[6] U0(t) allows us to continue thinking of $out_i(r)$ as binary—either 0 or 1. Suppose that we instead assume that $out_i(r)$ takes arbitrary values in the interval $[0,1]$, where, intuitively, the closer i's output is to the true function value in run r (given the private inputs in r), the closer $out_i(r)$ is to 1. Then we can prove our results as long as if i outputs a t-variant of $f(\vec{x})$ in r, then $u_i(r)$ is greater than i's expected utility from just guessing the function value.

and the computer science literature. In the economics literature, Ben-Porath [4] showed simulation of Nash equilibrium is possible if there is a "punishment" strategy. Intuitively, a punishment strategy provides a threat that players can use to force compliance with the simulation. For example, in our secret sharing game the threat is that players will stop playing if they detect misbehavior. Since we assume that all players prefer to get the secret than not to get it (no matter how many other players also get the secret), this is indeed a punishment. Heller [11] extends Ben-Porath's results to allow for coalitions.

In the computer science literature, Izmalkov, Micali, and Lepinski [12], generalizing the previous work of Lepinski et al. [16], show that simulation is possible provided that we can use a very strong primitive called an *envelope* and an entity called a *ballot-box*. A ballot-box is essentially an honest dealer that can do specific limited actions (like randomly permute envelopes). Envelopes have two operations: $Send(R, m)$, which corresponds to the action where the sender puts m in an envelope, writes his name on the envelope and hands it to R; and $Receive(S)$, which corresponds to the action where the receiver publicly opens all envelopes from S and privately reads their contents. As Lepinski, Micali, and Shelat [15] observe, envelopes cannot be implemented even over broadcast channels. Thus, a solution that depends on envelopes is inadequate when, for example, players are playing a game over the Internet.

We show that we can simulate a (k, t)-robust equilibrium using multiparty computation, provided that k and t satisfy the same bounds as in Theorem 3. Moreover, we do not always need to assume the existence of a punishment strategy; provided that k and t satisfy the appropriate bounds, we can replace the use of a punishment strategy by using Reed-Solomon decoding or (assuming cryptography) by using *verifiable secret sharing* and the GMW compiler. To make this precise, we first formalize the notion of a (k, t) punishment strategy.

DEFINITION 5. *A joint strategy $\vec{\rho}$ is a (k, t)-punishment strategy with respect to $\vec{\sigma}$ if for all $C, T, P \subseteq N$ such that C, T, P are disjoint, $|C| \le k$, $|T| \le t$, and $|P| > t$, for all $\vec{\tau}_T \in S_T$, for all $\vec{\phi}_C \in S_C$, for all $i \in C$ we have*

$$u_i(\vec{\sigma}_{-T}, \vec{\tau}_T) > u_i(\vec{\sigma}_{N-(C \cup T \cup P)}, \vec{\phi}_C, \vec{\tau}_T, \vec{\rho}_P).$$

Intuitively, $\vec{\rho}$ is (k, t)-punishment strategy with respect to $\vec{\sigma}$ if, for any coalition C of at most k players and any set T of nonstandard players, as long as more than t players use the punishment strategy and the remaining players play their component of $\vec{\sigma}$, all the players in C are worse off than they would be had everyone not in T played $\vec{\sigma}$. The idea is that the threat of having more than t players use their component of $\vec{\rho}$ is enough to stop players in C from deviating from $\vec{\sigma}$.

Given this definition, we give a complete characterization of equilibria that can be simulated using what economists call *cheap talk*. In the language of distributed computing, we assume that each pair of agents has a secure private channel, and can use communication over this channel to facilitate reaching an equilibrium. Formally, Γ_{CT} is a *private channel Cheap-Talk* (CT) extension of the game Γ if the mediator in Γ_{CT} acts as a private channel between every pair of players. Specifically, we assume that the inputs that each player i sends to the mediator d in phase 1 of a stage always have the form $((m_1, i_1), \ldots, (m_k, i_k))$; such an input should be interpreted as "send message m_j to i_j", for $j = 1, \ldots k$. In phase 2 of that stage, the mediator simply relays these messages to the appropriate recipients. We omit the formal details here. Although we present our protocols as if there are private channels between the players, for the results where we use cryptography (parts (c) and (d) of Theorem 4), it suffices that there are public channels.

We now can get a generalization of Theorem 3. The U2 requirement is replaced by the assumption that there is a punishment strategy; we do not need this assumption for the cases where U2$'$ suffices. (Recall that U1 and U2 imply that not sending messages is a punishment strategy.) Note that perhaps the right way to think about Theorem 3(a) (and part (a) of all our other theorems) is that we have a mechanism that works for a family of games that have the same game tree except that the utilities of the outcomes may differ (although they always satisfy U1 and U2). Formally, we say that a game Γ' is a *utility-variant* of a game Γ if Γ' and Γ have the same game tree, but the utilities of the players may be different in Γ and Γ'.

Given a game Γ, let $u_i^h(\vec{\rho})$ be i's best-case payoff if at least h players use strategy ρ_j (and the remaining players use an arbitrary strategy); let m_i now denote the maximum payoff that i receives in Γ. Like the other simulation results in the literature, our simulation results apply only to *normal-form* games, which can be viewed as games where each player moves only once, and the moves are made simultaneously.

THEOREM 4. *Suppose that Γ is an n-player normal-form game and that $\vec{\sigma}$ is a (k, t)-robust strategy for a game Γ_d with a mediator d based on Γ.*

(a) If $3(t + k) < n$, then there exists a strategy $\vec{\sigma}'$ and a CT extension Γ_{CT} of Γ such that for all utility variants Γ' of Γ, if $\vec{\sigma}$ is a (k, t)-robust strategy for Γ'_d, then $(\vec{\sigma'}, \Gamma'_{CT})$ is a (k, t)-robust mechanism implementing $(\vec{\sigma}, \Gamma'_d)$, where Γ'_{CT} is the utility variant of Γ_{CT} corresponding to Γ'_d. Moreover, $\vec{\sigma}'$ has an expected running time of 2 rounds.

(b) If $3t + 2k < n$, and there exists a (k, t)-punishment strategy $\vec{\rho}$ with respect to $\vec{\sigma}$, then there exists a strategy $\vec{\sigma}'$ that takes a parameter β and a CT extension Γ_{CT} of Γ such that if $\beta < \max_{i \in N} \frac{u_i(\vec{\sigma}) - u_i(\vec{\rho})}{2m_i - (u_i(\vec{\sigma}) + u_i(\vec{\rho}))}$, then $(\vec{\sigma'}, \Gamma_{CT})$ is a (k, t)-robust mechanism implementing $(\vec{\sigma}, \Gamma_d)$, and $\vec{\sigma}'$ has an expected running time of 2 rounds.

(c) If $2(t + k) < n$, then, assuming cryptography, for all $\epsilon > 0$, there exists a strategy $\vec{\sigma}'$ and a CT extension Γ_{CT} of Γ such that if $\vec{\sigma}$ is a (k, t)-robust strategy for Γ'_d, then $(\vec{\sigma'}, \Gamma_{CT})$ is an ϵ-(k, t)-robust mechanism that implements $(\vec{\sigma}, \Gamma_d)$, and $\vec{\sigma}'$ has an expected running time of 2 rounds.

(d) If $2t + k < n$ and there exists a (k, t)-punishment strategy $\vec{\rho}$ with respect to $\vec{\sigma}$, then, assuming cryptography, for all $\epsilon > 0$, there exists a strategy $\vec{\sigma}'$ that takes a parameter α such that if $\alpha \le \min_{i \in N} \frac{u_i(N) - u_i(\emptyset)}{m_i - u_i(\emptyset)}$, then $(\vec{\sigma'}, \Gamma_{CT})$ is an ϵ-(k, t)-robust mechanism implementing $(\vec{\sigma}, \Gamma_d)$, and $\vec{\sigma}'$ has an expected running time of $O(1/\alpha)$ rounds.

We remark that if we assume that the strategy $\vec{\sigma}$ survives iterated deletion in Γ_d, then we can assume that $\vec{\sigma}'$ does as well.

Note that in part (a) of Theorem 4 we do not need to know the utility; in parts (b), (c), and (d) we do. In parts (a) and (c), we do not need to assume a punishment strategy; in parts (b) and (d), we do. Ben-Porath [4] essentially proved the special case of part (b) where $t = 0$ and $k = 1$ (i.e, where $\vec{\sigma}$ is a Nash equilibrium). Theorem 4(a) implies that any Nash equilibrium can be simulated if there are at least four players, even without a punishment strategy, and thus significantly strengthens Ben-Porath's result. Heller [11] extends Ben-Porath's result to arbitrary k, proving the special case of Theorem 4(b) with $t = 0$. Heller also claims a matching lower bound. While we believe that a matching lower bound does hold (see Conjecture 1), Heller's lower bound argument seems to have some gaps. Specifically, he seems to assume that all the randomization (if any) in the implementation happens after messages have been exchanges, rather than allowing the message exchange itself to depend on the randomization.

We believe we have tight lower bounds corresponding to Theorem 4. In particular, we believe we can prove the following, although details remain to be checked.

CONJECTURE 1. *(a) If $n = 3(t + k) > 0$, then there exists an n-player game Γ and strategies $\vec{\sigma}$ such that $\vec{\sigma}$ is a (k, t)-robust equilibrium for a game Γ_d with a mediator d based on Γ, but there is no strategy $\vec{\sigma}'$ and CT extension Γ_{CT} such that $(\vec{\sigma}', \Gamma_{CT})$ is a (k, t)-robust mechanism implementing $(\vec{\sigma}, \Gamma_d)$.*

(b) If $n = 3t + 2k > 0$ then there exists an n-player game Γ and strategies $\vec{\sigma}$ and $\vec{\rho}$ such that $\vec{\sigma}$ is a (k, t)-robust strategy with respect to a (k, t)-punishment strategy $\vec{\rho}$ for a game Γ_d with a mediator d based on Γ, but there is no strategy $\vec{\sigma}'$ and CT extension Γ_{CT} such that $(\vec{\sigma}', \Gamma_{CT})$ is a (k, t)-robust mechanism implementing $(\vec{\sigma}, \Gamma_d)$.

(c) If $n = 2t + 2k > 0$ then, assuming cryptography, for any $\epsilon > 0$, there exists an n-player game Γ and strategies $\vec{\sigma}$ and $\vec{\rho}$ such that $\vec{\sigma}$ is a (k, t)-robust strategy with respect to a (k, t)-punishment strategy $\vec{\rho}$ for a game Γ_d with a mediator d based on Γ, but there is no strategy $\vec{\sigma}'$ and CT extension Γ_{CT} such that $(\vec{\sigma}', \Gamma_{CT})$ is an ϵ-(k, t)-robust mechanism implementing $(\vec{\sigma}, \Gamma_d)$.

(d) If $n = 2t + k > 0$ then, assuming cryptography, for any $\epsilon > 0$, there exists an n-player game Γ and strategies $\vec{\sigma}$ such that $\vec{\sigma}$ is a (k, t)-robust strategy for a game Γ_d with a mediator d based on Γ, but there is no strategy $\vec{\sigma}'$ and CT extension Γ_{CT} such that $(\vec{\sigma}', \Gamma_{CT})$ is an ϵ-(k, t)-robust mechanism implementing $(\vec{\sigma}, \Gamma_d)$.

Our sketch proof of Conjecture 1 uses ideas from the Byzantine agreement literature. For example, the proof of part (a) involves constructing a game that essentially captures Byzantine agreement; we then appeal to the fact that Byzantine agreement cannot be done without cryptography if there are more than $n/3$ Byzantine processes [14]. Similarly, part (c) uses the fact that *uniform* agreement, where the faulty processes have to decide on the same value as the nonfaulty processes (if they decide on anything at all) cannot be done with *generalized omission failures* (where processes may both fail to send message and fail to receive messages) if there are more than $n/2$ faulty processes [20]. We also need to use ideas from the GMW compiler to force processes to behave as if they are only suffering from omission failures, rather than Byzantine failures. We hope to complete the proof and report on the details shortly.

7. REFERENCES

[1] E. Adar and B. Huberman. Free riding on Gnutella. *First Monday*, 5(10), 2000.

[2] R Aumann. Acceptable points in general cooperative n-person games. *Contributions to the Theory of Games, Annals of Mathematical Studies*, IV:287–324, 1959.

[3] 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 1–10, 1988.

[4] E. Ben-Porath. Cheap talk in games with incomplete information. *J. Economic Theory*, 108(1):45–71, 2003.

[5] B. D. Bernheim, B. Peleg, and M. Whinston. Coalition proof Nash equilibrium: Concepts. *J. Economic Theory*, 42(1):1–12, 1989.

[6] F. M. Forges. An approach to communication equilibria. *Econometrica*, 54(6):1375–85, 1986.

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

[8] O. Goldreich, S. Micali, and A. Wigderson. How to play any mental game. In *Proc. 19th ACM Symp. Theory of Computing*, pages 218–229, 1987.

[9] D. Gordon and J. Katz. Rational secret sharing, revisited. Unpublished manuscript, 2006.

[10] J. Y. Halpern and V. Teague. Rational secret sharing and multiparty computation: extended abstract. In *Proc. 36th ACM Symp. Theory of Computing*, pages 623–632, 2004.

[11] Yuval Heller. A minority-proof cheap-talk protocol. Unpublished manuscript, 2005.

[12] S. Izmalkov, S. Micali, and M. Lepinski. Rational secure computation and ideal mechanism design. In *Proc. 46th IEEE Symp. Foundations of Computer Science*, pages 585–595, 2005.

[13] J. Justesen. On the complexity of decoding Reed-Solomon codes (corresp). *IEEE Trans. on Information Theory*, 22(2):237–238, 1976.

[14] L. Lamport, R. Shostak, and M. Pease. The Byzantine Generals problem. *ACM Trans. on Programming Languages and Systems*, 4(3):382–401, 1982.

[15] M. Lepinksi, S. Micali, and A. Shelat. Collusion-free protocols. In *Proc. 37th ACM Symp. Theory of Computing*, pages 543–552, 2005.

[16] 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 1–10, 2004.

[17] A. Lysyanskaya and N. Triandopoulos. Rationality and adversarial behavior in multi-party computation. Unpublished manuscript, 2006.

[18] D. Malkhi, N. Nisan, B. Pinkas, and Y. Sella. Fairplay - secure two-party computation system. In *Proc. 13th USENIX Security Symposium*, pages 287–302, 2004.

[19] D. Moreno and J. Wooders. Coalition-proof equilibrium. *Games and Economic Behavior*, 17(1):80–112, 1996.

[20] G. Neiger and S. Toueg. Automatically increasing the fault-tolerance of distributed algorithms. *Journal of Algorithms*, 11(3):374–419, 1990.

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

[22] T. Rabin and M. Ben-Or. Verifiable secret sharing and multiparty protocols with honest majority. In *Proc. 21st ACM Symp. Theory of Computing*, pages 73–85, 1989.

[23] A. Shamir. How to share a secret. *Commun. ACM*, 22(11):612–613, 1979.

[24] Y. Shoham and M. Tennenholtz. Non-cooperative computing: Boolean functions with correctness and exclusivity. *Theoretical Computer Science*, 343(1–2):97–113, 2005.

[25] A. Yao. Protocols for secure computation (extended abstract). In *Proc. 23rd IEEE Symp. Foundations of Computer Science*, pages 160–164, 1982.