Perfectly Secure Message Transmission

DANNY DOLEV AND CYNTHIA DWORK

IBM Almaden Research Center, San Jose, California

ORLI WAARTS

Stanford University, Stanford, California

AND

MOTI YUNG

IBM Thomas J. Watson Research Center, Yorktown Heights, New York

Abstract. This paper studies the problem of perfectly secure communication in general network in which processors and communication lines may be faulty. Lower bounds are obtained on the connectivity required for successful secure communication. Efficient algorithms are obtained that operate with this connectivity and rely on no complexity-theoretic assumptions. These are the first algorithms for secure communication in a general network to simultaneously achieve the three goals of perfect secrecy, perfect resiliency, and worst-case time linear in the diameter of the network.

Categories and Subject Descriptors: C.2.0 [**Computer-Communication Networks**]: General—*security and protection*; C.2.4 [**Computer-Communication Networks**]: Distributed Systems—*distributed applications*; F.1.2 [**Computation by Abstract Devices**]: Modes of Computation—*probabilistic computation.*

General Terms: Algorithms, Reliability, Security

Additional Key Words and Phrases: Distributed computing, fault-tolerance, perfectly secure communication

1. *Introduction*

Recent advances in fiber optics make the construction of networks with immense bandwidth realizable. As more and more personal and business communication will take place over these systems, issues of correctness and

The research of Cynthia Dwork was conducted while she was on sabbatical at the Laboratory for Computer Science of the Massachusetts Institute of Technology.

The work of Orli Waarts was supported in part by Office of Naval Research N00014-68-K-0166.

Authors' addresses: D. Dolev and C. Dwork, IBM Research Division, Almaden Research Center, 650 Harry Road, San Jose, CA 95120-6099; O. Waarts, Stanford University, Stanford, CA; M. Yung, IBM Thomas J. Watson Research Center, Yorktown Heights, NY.

privacy become increasingly important. In this paper, we solve the problem of perfectly secure message transmission in communication networks, without complexity-theoretic assumptions and with perfect correctness, for processor and edge faults alike. Our approach is to abstract away the network entirely and concentrate on solving the Secret Message Transmission (SMT) problem for a single pair of processors we call *Sender* and *Receiver*. In the SMT problem, two synchronized nonfaulty processors, Sender and Receiver, are connected by some number n of wires. We may think of these wires as a collection of vertex-disjoint paths between Sender and Receiver in the underlying network; each path corresponds to a wire. The Sender has a secret message m, drawn from a finite set Q of values. There are two parameters, σ (for secrecy) and ρ (for resiliency). The problem is for the Sender to convey m to the Receiver while satisfying:

Perfect Secrecy. For all sets L of at most σ wires, no listening adversary $\mathcal{A}_L$, listening to all the wires of L, learns anything about m (secrecy must be information theoretic).

Perfect Resiliency. For all sets D of at most ρ wires (possibly, but not necessarily, disjoint from L), Receiver correctly learns m, regardless of the disrupting adversary $\mathcal{A}_D$ controlling and coordinating the behavior of the wires in D.

Since each wire corresponds to a path in the underlying network, a compromised wire in Secret Message Transmission corresponds to a compromised processor or edge on the corresponding network path. Thus, connectivity bounds for SMT yield connectivity bounds in the network as a function of the number of faulty nodes or edges to be tolerated.

Our protocol for secure transmission in general networks is the first to simultaneously achieve the three goals of perfect secrecy, perfect resiliency, and worst-case time linear in the diameter of the network (the constant is at most 3). This contrasts with the similarly fast protocol of Rabin and Ben-Or, based on the idea of "check vectors," which has unconditional secrecy but has small probability of error [20]. Ben-Or, et al. [3] showed that every function of p inputs can be efficiently computed by a complete network of p processors even in the presence of $t < p/3$ Byzantine faults so that no set of t faulty processors gets any information other than the function value (see also [1], [7], [16], and [22]). Using our protocol for secret message transmission we can immediately extend these results to any p processor network of connectivity $2t + 1$, at no cost in secrecy or correctness (connectivity $2t + 1$ is necessary [8, 16]). The analogous result obtained by Rabin and Ben-Or in [20] for general networks suffers a small probability of error. In this low-probability case, because the entire computation can go awry, the privacy of correct processors is not guaranteed, even though messages sent between correct processors enjoy perfect secrecy. Our solution does not suffer from this weakness, and pays no price in time.

The bounds on connectivity needed for (σ, ρ)-SMT vary according to whether or not the solution must be 1-*way*, in that information flows only from Sender to Receiver, or 2-*way*, where Sender and Receiver "converse." The bounds are also strongly affected by the extent to which $\mathcal{A}_D$ can communicate to $\mathcal{A}_L$ its view of the communication on the wires in D. One possibility is that the two

adversaries share a "back channel" allowing them explicitly to communicate. A more interesting case is when there is no such channel. Here $\mathscr{A}_D$ can communicate information to $\mathscr{A}_L$ if the sets D and L intersect (by placing messages on the shared wires). More subtly, even if the two sets are disjoint, $\mathscr{A}_D$ may be able to transmit information to $\mathscr{A}_L$ by disrupting the protocol so as to elicit certain behavior on the part of Sender or Receiver that $\mathscr{A}_L$ can recognize. One situation in which $\mathscr{A}_D$ clearly cannot communicate useful information to $\mathscr{A}_L$ is, informally, when it disrupts obliviously, independent of the information on the wires in D. As an example, we propose the weaker fault model in which communication is disrupted only by random noise.

The case in which $\mathscr{A}_D$ and $\mathscr{A}_L$ are constrained so that $D \subseteq L$ or $L \subseteq D$ is an important one, and in this case we say the *containment assumption* holds. In this case, there is effectively one adversary. This is the worst-case assumption made in previous papers treating secrecy and resiliency simultaneously [3, 4, 5, 15, 20]. Generally, we assume $\sigma \geq \rho$ and derive our bounds for this case. In some models, we can drop this assumption and replace all terms σ in our bounds by $\max\{\sigma, \rho\}$. In other models, the problem has no solution if $\sigma < \rho$. We return to this point as needed. Not only do our lower and upper bounds match under the containment assumption, but in this case they are independent of the extent of communication between the adversaries. Under the containment assumption, there is a solution to the 1-way Secret Message Transmission problem if and only if n, the number of wires connecting Sender and Receiver, satisfies $n \geq \sigma + 2\rho + 1$. The solution requires computation and message length polynomial in n. However, if communication is 2-way, in that Sender and Receiver converse, then $n \geq \max\{\sigma + \rho + 1, 2\rho + 1\}$ wires are necessary and sufficient (the latter term arises even in the case $\sigma = 0$, that is, we require correctness but no secrecy). A *phase* is a send from Sender to Receiver or *vice versa*. Surprisingly, the 2-way protocol requires only three phases.

For this value of n we have even obtained a 2-*phase* protocol (beginning with a transmission from Receiver to Sender), but, unlike our 1-phase and 3-phase solutions, the computation and communication costs of the two-phase solution are not polynomial in n. Our 3-phase solution uses two new techniques. The first is a simple fault detection technique, a powerful generalization of which has already been applied to another problem [10]. We say more about the generalization and its application in Section 9. The second is a method of parallelizing our first technique, permitting us to collapse loop iterations in a $\Theta(\rho)$-phase algorithm to obtain the 3-phase algorithm. This too, has been applied in [10].

If the two adversaries can communicate explicitly during the execution of the protocol, say, through some auxiliary "back channel," but the containment assumption does not hold, then the (lower and upper) bounds on n increase by exactly ρ for both the 1-way and 2-way cases. This is because when $\mathscr{A}_D$ can communicate with $\mathscr{A}_L$ it is as if there are $\sigma + \rho$ listeners, of which at most ρ are disruptors. This is the containment assumption with secrecy parameter $\sigma + \rho$ and resiliency parameter ρ, and the bounds increase accordingly over the case with only σ listeners.

Even if the two adversaries cannot communicate through a back channel, it may be possible for the disrupting adversary to elicit certain behavior from Sender or Receiver that communicates some extra information to the listening

adversary. In fact, at least an additional $\rho - 1$ wires must be added for both the 1-way and 2-way cases. These last results are tight for the 1-way case and leave a gap of a single wire in the 2-way case. However, we also show that in this case any 3-*phase* algorithm requires exactly ρ additional wires, even if the disruptors and listeners cannot move between phases. This bound is tight, and our algorithm permits these sets to move. All our protocols tolerate adversaries of unlimited computational power.

1-way SMT has an interesting relation to Verifiable Secret Sharing (VSS), a problem first defined by Chor, et al. [5]. VSS plays a central role in implementing a global coin [13], as well as in the more general results of [3, 4, 20].[1] Ben-Or, et al. remark without proof that secure computation is impossible with $2t + 1$ processors, even in the presence of a broadcast channel [3]. We prove that even the more fundamental task of Verifiable Secret Sharing cannot be achieved in this model.[2] Our approach is to reduce a weakened version of 1-way SMT to VSS so that each processor in the VSS protocol corresponds to a wire in the SMT protocol. We then prove a lower bound on connectivity of $3t + 1$ for this weakened version of SMT. Interestingly, our lower bound of $3t + 1$ also applies to a weak version of VSS called *Unverified Secret Sharing* (*t-USS*). This is essentially VSS without Verification. Thus, the processor cost of Verifiable Secret Sharing comes from the conflicting requirements of secrecy and reconstructability, rather than from the ability to verify that the secret was correctly dealt out.

Rabin and Lehmann [19] showed that in a distributed environment there exist problems with randomized solutions but with no deterministic solution (see also [2], [9], and [17]). There exist error-free and small-error solutions to t-USS requiring $3t + 1$ and $2t + 1$ processors, respectively [3] and [20]. The lower bound of $3t + 1$ processors for error-free t-USS yields a new kind of separation result: within the class of problems admitting no deterministic solution, the cost of an error-free solution may necessarily significantly exceed the cost of a solution with even very small probability of error. In a certain model, it is therefore possible to separate error-free randomized computation from randomized computation with error.

We also explore the problem of secure communication in graphs of bounded degree. Techniques of Dwork, et al. [11] for (nonsecret) communication in size n networks of bounded degree, can be extended to show that for substantial (as a function of n) σ and ρ, no matter which σ nodes are chosen by $\mathscr{A}_L$, and no matter which ρ nodes are chosen by $\mathscr{A}_D$, there is a large set of nodes that can communicate secretly and correctly among themselves, even though the network is of bounded degree.

The rest of the paper is organized as follows: Section 2 describes our adversary models. Section 3 contains definitions of the Secret Message Transmission and the two Secret Sharing problems. Section 4 contains our 3-phase solution to SMT. Additional results for the containment and noncontainment

[1] Roughly speaking, t-VSS permits a (possibly faulty) dealer to commit to a secret in such a way that the secret can later be uniquely reconstructed despite the interference of up to t faulty processors, possibly including the dealer. Moreover, if the dealer is correct then the faulty processors cannot learn any information about the secret until the correct processors execute a reconstruction protocol.

[2] The lower bound for t-VSS with a broadcast channel was obtained independently by Rabin and Ben-Or [20]. An informal argument is also given by Chaum, et al. [4].

cases appear in Sections 5 and 7. Our results about Verifiable Secret Sharing and the separation result for problems with no deterministic solution appear in Section 6. Applications to networks of bounded degree are discussed in Section 8. Section 9 contains additional remarks.

2. *Adversaries*

An *adversary* is an algorithm that takes as input transmissions on certain wires, random bits, and the phase number, and produces a choice of additional wires together with either (faulty) traffic on the chosen wires, in the case of the *disrupting* adversary $\mathscr{A}_D$, or a guess of the message being transmitted, in the case of the *listening* adversary, $\mathscr{A}_L$. A wire tapped or under the control of an adversary is said to be *compromised.*

For our algorithms, our adversaries may be *adaptive*, in the sense that information (communication traffic) obtained from a set of compromised wires can affect the choice of the next wire to be compromised, while our lower bounds hold even if the adversaries are not adaptive.

Our algorithms have a special form: The first phase uses a low-quality "secret" channel, while all subsequent phases use a perfect "public" channel. Without going into detail here of how we achieve these different types of channels, we point out that the issue of choosing wires to compromise in subsequent phases as a function of traffic in the first phase is moot for these algorithms. Similarly moot is the issue of whether $\mathscr{A}_D$ even exists after the first phase, since by definition it cannot interfere with the perfect public channel. Of course, $\mathscr{A}_L$ may use all the information it has gleaned over the entire execution of the protocol for making its guess as to which message is transmitted. The lower bounds hold even for static adversaries that choose which wires to compromise before execution begins. We therefore assume the sets D and L of disrupting and listening wires are chosen by the end of the first phase.

In this paper, we assume in general that $\mathscr{A}_L$ and $\mathscr{A}_D$ work together to defeat the algorithm. If the adversaries can communicate explicitly during execution of the algorithm, say, through some "back channel," then we simply say that $\mathscr{A}_L$ *and* $\mathscr{A}_D$ *communicate*. Here, a *back channel* is some channel other than the wires connecting Sender and Receiver. In this case, for example, the adversaries might converse before choosing which wires to compromise. If there is no "back channel" we say $\mathscr{A}_D$ *and* $\mathscr{A}_L$ *do not communicate*. Even in this case, some communication is possible. For example, if the sets D and L intersect, then $\mathscr{A}_D$ can convey information to $\mathscr{A}_L$ by putting this information onto the shared wire(s) or disrupting communication on a shared wire. Even if D and L are disjoint, the protocol may require Sender and Receiver to send over wires in L information reflecting the choice of D. This too could be meaningful to $\mathscr{A}_L$.

We also consider the special case in which $\mathscr{A}_D$ behaves obliviously, choosing D, communicating with $\mathscr{A}_L$, and disrupting communication along the wires in D, without regard to the information placed along these wires by Sender and Receiver (but possibly dependent on information it receives from $\mathscr{A}_L$, in the case that the adversaries communicate). This adversary models the special case in which disruption is due only to random noise. Clearly, an oblivious $\mathscr{A}_D$ cannot give to $\mathscr{A}_L$ any information about the transmissions of Sender and Receiver not already available to $\mathscr{A}_L$. Not surprisingly, we obtain better upper bounds against this weaker adversary than in the nonoblivious case.

3. *Definitions*

Sender and Receiver are modeled by communicating probabilistic Turing Machines that communicate through the n wires connecting them. Randomization is modeled by coin flipping (bounded branching).

Throughout, our messages m are drawn from a finite field Q of prime cardinality greater than n, where n is always the number of wires between Sender and Receiver in Secret Message Transmission or the number of participants in a secret sharing protocol, whichever is appropriate to the context. We let Π denote the underlying probability distribution on Q.

We use the notation $[k]$ to denote the set of natural numbers less than or equal to k. Note that $0 \notin [k]$. We let $(k) = \{0\} \cup [k]$. For any set S, we let S^i denote the set of j-subsets of S where $0 \leq j \leq i$.

For any alphabet Σ, for any vectors $W, V \in \Sigma^n$, the distance between W and V, denoted $dist(W, V)$, is the number of components on which the two vectors differ.

Fix any secret message transmission protocol, P, and let $\mathscr{A}_L$ be a listening adversary. Intuitively, we require that for all messages m, m' and for all disrupting adversaries $\mathscr{A}_D$, the probability distribution on $\mathscr{A}_L$'s view, given that the message transmitted is m and the disrupting adversary is $\mathscr{A}_D$, is identical to the probability distribution on $\mathscr{A}_L$'s view, given that the message transmitted is m' and the adversary is still $\mathscr{A}_D$. Here, the probability space is the space of all coin tosses of $\mathscr{A}_L, \mathscr{A}_D$, Sender, and Receiver, and the view, intuitively, is everything seen by $\mathscr{A}_L$.

More precisely, the *View* of a listening adversary $\mathscr{A}_L$ at any point in the execution of the protocol consists of

(1) the algorithms $\mathscr{A}_L$ and $\mathscr{A}_D$, and the protocol P;
(2) the random choices that $\mathscr{A}_L$ has made so far;
(3) the "back channel" messages received up to this point, if any (and if there is a back channel);
(4) for each wire l in the subset of L chosen so far, conversations between Sender and Receiver over l from the time the wire was compromised until this point;
(5) for each wire w in the subset of $L \cap D$ chosen so far, the changes by $\mathscr{A}_D$ to conversations over w from the time w was compromised until this point.

Sometimes we combine the last two items in the view, calling the combination the *traffic* over the wires in L.

None of our lower bound proofs use the assumption that $\mathscr{A}_L$ sees both the original data placed on wires in $L \cap D$ by Sender and Receiver, as well as the changes $\mathscr{A}_D$ makes to these wires. Some proofs use the ability of $\mathscr{A}_L$ to detect that $\mathscr{A}_D$ has disrupted communication on a certain wire, while others (specifically, in the case $D \cap L = \emptyset$) rely on the ability of $\mathscr{A}_L$ to see the conversations. However, our algorithms work even if $\mathscr{A}_L$ has access to all the traffic over the wires in L.

For every message $m \in Q$, any pair of adversaries $\mathscr{A}_L, \mathscr{A}_D$, and any protocol P for SMT, let $\hat{\Pi}(\mathscr{A}_L, m, \mathscr{A}_D, P)$ denote the probability distribution, on the views of $\mathscr{A}_L$ at the end of the executions of P when the message sent is m and the disrupting adversary is $\mathscr{A}_D$. The probability distribution is taken over the coin tosses of $\mathscr{A}_D$, $\mathscr{A}_L$, Sender, and Receiver.

Definition. (σ, ρ)*-Secret Message Transmission* ((σ, ρ)*-SMT*). The Sender begins with a message m drawn from an arbitrary probability distribution Π on Q. For every $\mathscr{A}_L, \mathscr{A}_D$, compromising at most σ and ρ wires, respectively, we require:

Secrecy. $\forall m' \in Q\ \hat{\Pi}(\mathscr{A}_L, m, \mathscr{A}_D, P) = \hat{\Pi}(\mathscr{A}_L, m', \mathscr{A}_D, P)$.

Resiliency. Receiver correctly learns m.

In particular, the secrecy requirement implies that at any point in the execution $\mathscr{A}_L$ has absolutely no information about which message is being transmitted. It follows that the choice of L is independent of the message being transmitted, as is the probability distribution on conversations over wires in L. Our definition of secrecy is equivalent to requiring that the probability that $\mathscr{A}_L$ can correctly guess the secret being transmitted is

$$p_{\max} = \max_{m \in Q} \Pi(m),$$

where the probability is taken over all coin flips of Sender, Receiver, $\mathscr{A}_L$ and $\mathscr{A}_D$ and choice of m (we can think of m as being randomly chosen according to Π). We mention this equivalence because we sometimes prove lower bounds by exhibiting a pair of adversaries $\mathscr{A}_L$ and $\mathscr{A}_D$ such that, if the connectivity is insufficient, $\mathscr{A}_L$ can guess the secret message transmitted with probability greater than $p_{\max}$.

A solution to 1-way (σ, ρ)-SMT runs in exactly one synchronous phase. A solution to 2-way (σ, ρ)-SMT is a solution to (σ, ρ)-SMT of two or more phases. We adopt the convention that if $\sigma = 0$, then there is no secrecy requirement, and if $\rho = 0$, then there is no resiliency requirement. If $\mathscr{A}_D$ and $\mathscr{A}_L$ are constrained so that $D \subseteq L$ or $L \subseteq D$, then we say the *containment assumption* holds. Unless otherwise noted, we assume $\sigma \geq \rho$, and in this case the containment assumption says that $D \subseteq L$. All our results for the containment case are independent of the degree of communication between $\mathscr{A}_L$ and $\mathscr{A}_D$. Under the containment assumption the secrecy condition above is equivalent to the following condition:

Secrecy Under Containment. $\forall m, m' \in Q$, $\forall \mathscr{A}_D, \mathscr{A}_L$, $\forall L \in (n-1)^\sigma$ compromised by $\mathscr{A}_L$, $\forall D \in L^\rho$ compromised by $\mathscr{A}_D$, for all possible traffic T_L over wires in L, the probability that T_L occurs, given that the message transmitted is m and given $\mathscr{A}_D, \mathscr{A}_L$, is equal to the probability that T_L occurs, given that the message is m' and given $\mathscr{A}_D$ and $\mathscr{A}_L$. The probability space is the set of coin tosses of $\mathscr{A}_D$, $\mathscr{A}_L$, Sender, and Receiver.

Note that for any fixed $\mathscr{A}_L, \mathscr{A}_D$ pair, the probability that $\mathscr{A}_L$ will compromise wire 0 is independent of the secret message. Thus, the probability that $\mathscr{A}_L$ chooses any particular L not containing 0 is independent of the message.

As described in the Introduction, we will study a weakened form of 1-way SMT (under containment) in which there is no secrecy requirement if $\mathscr{A}_L$ compromises wire 0. We call this *weakened 1-way SMT*. Specifically, we weaken the above definition to read "If $0 \notin L$, then for all possible traffic... ."

We now turn briefly to secret sharing. This is actually a class of problems, all having a similar form. The model is a distributed system in which certain processors may be disruptors and certain others may be listeners. As above, the disruptors are controlled by a disrupting adversary $\mathscr{A}_D$, while the messages and

other inputs received by the listeners are available to a listening adversary $\mathscr{A}_L$. In this model, there is no way to prevent the disruptors from sending messages to the listeners, and hence $\mathscr{A}_D$ can communicate with $\mathscr{A}_L$. Thus, either we work under the containment assumption or we assume $\mathscr{A}_D$ is oblivious.

Definition. (σ, ρ)*-Secret Sharing.* A protocol for a (σ, ρ)-Secret Sharing problem is a pair of n-processor protocols $(\mathscr{P}_1, \mathscr{P}_2)$, run-in sequence, and designed to tolerate up to ρ faults in any execution of the pair. In other words, if some number $k \leq \rho$ processors fail in $\mathscr{P}_1$, then $\mathscr{P}_2$ need only tolerate faulty behavior by those same k processors and up to $\rho - k$ additional processors. One special processor, p_0, is called the *dealer*. The dealer has a private input m. During $\mathscr{P}_1$ the dealer distributes shares of m in such a way that no set of σ processors not including the dealer, learns any information about the secret during execution of $\mathscr{P}_1$. $\mathscr{P}_2$ is a protocol for reconstructing the secret m from the shares distributed during $\mathscr{P}_1$. Finally, if the dealer p_0 remains nonfaulty throughout $\mathscr{P}_1$, then the value obtained by applying $\mathscr{P}_2$ is in fact the initial value (input) of p_0, provided at most ρ processors fail in total.

In analogy to the definition of SMT, we assume $\mathscr{A}_L$ compromises a set L of listening processors, and $\mathscr{A}_D$ compromises a set D of disrupting processors. In this case, the view of $\mathscr{A}_L$ is the complete history of every processor in L, from the moment it is compromised by $\mathscr{A}_L$ until the beginning of the execution of $\mathscr{P}_2$, together with any information received directly from $\mathscr{A}_D$. In the case of Unverified Secret Sharing, we assume $D \subseteq L$. This is also the assumption in Verifiable Secret Sharing in the literature.

Definition. (σ, ρ)*-Unverified Secret Sharing.* For every $m \in Q$, if p_0 has input m and remains nonfaulty throughout execution of $\mathscr{P}_1$, we require that for all $\mathscr{A}_L$ and $\mathscr{A}_D$ compromising sets $L \in (n-1)^\sigma$ and $D \in L^\rho$, respectively.

Secrecy. $\forall m' \in Q$, if $p_0 \notin L$, then the probability distribution on the views of $\mathscr{A}_L$, given $\mathscr{A}_L, \mathscr{A}_D$, and given that p_0 has input m, is identical to the probability distribution on the views of $\mathscr{A}_L$ given $\mathscr{A}_L, \mathscr{A}_D$, and given that p_0 has input m'.

Resiliency. At the end of $\mathscr{P}_2$, every processor not in D outputs m, regardless of the behavior of the members of D.

Note that execution of $\mathscr{P}_2$ need not immediately follow execution of $\mathscr{P}_1$, but may be delayed, so even if the dealer is correct throughout execution of $\mathscr{P}_1$ it may fail before execution of $\mathscr{P}_2$.

As in the case of weakened 1-way SMT, it follows from this definition that the probability that $\mathscr{A}_L$ compromises p_0 given that the secret is m is the same as the probability that $\mathscr{A}_L$ compromises p_0 given that the secret is m'. Again, as in that case, for all $m, m', \mathscr{A}_D, \mathscr{A}_L$, for all $L \in [n-1]^\sigma$, all $D \in L^\rho$, and all possible views V_L that $\mathscr{A}_L$ could have at the end of execution of $\mathscr{P}_1$, given this choice of L, the probability that V_L occurs given $\mathscr{A}_D$ and $\mathscr{A}_L$, and given that the secret is m, is identical to the probability that V_L occurs given $\mathscr{A}_D, \mathscr{A}_L$, and given that the secret is m'.

In the case of (t, t)-Unverified Secret Sharing, the definition says that if p_0 remains nonfaulty throughout $\mathscr{P}_1$, then no set F of at most t faulty players can prevent the nonfaulty players from outputtting m at the end of $\mathscr{P}_2$, and moreover, if F does not contain p_0 then the members of F have no informa-

tion, in an information theoretic sense, about the secret m. When the secrecy and resiliency parameters are the same, as in this case, we simply write *t-Unverified Secret Sharing*.

t-Verifiable Secret Sharing is t-Unverified Secret Sharing with additional correctness constraints for the case in which the dealer is faulty during execution of $\mathscr{P}_1$. Specifically, even if the dealer is faulty during execution of $\mathscr{P}_1$, VSS requires that the outcome of $\mathscr{P}_2$ is uniquely determined by the states of any subset of $n - t$ processors correct at the end of $\mathscr{P}_1$, provided at most t processors fail in total during execution of the two protocols. That is, once $\mathscr{P}_1$ is completed, the dealer is committed to the secret dealt out. Clearly, t-USS reduces to t-VSS.

4. *The Main Algorithm*

In this section, we present our 3-phase protocol for 2-way (σ, ρ)-SMT. The protocol requires connectivity $\max\{\sigma + \rho + 1, 2\rho + 1\}$ under the containment assumption or in the case $\mathscr{A}_D$ is oblivious. We prove in Section 5 that this is optimal. Extensions to the noncontainment case appear in Section 7. Communication and computation costs are polynomial in σ and ρ.

We develop the protocol in three stages. We begin with an algorithm that might require $O(\rho)$ phases, proceed to an algorithm that can be made to run in three phases with any probability less than 1, and finally arrive at the 3-phase solution.

Throughout this section, we take the field Q to be Z_q, where q is a prime greater than the connectivity n. This is for ease of notation, since in this case the nonzero elements of Q are simply the integers $1, 2, \ldots, q - 1$. For arbitrary Q, we let $\alpha_1, \ldots, \alpha_{q-1}$ be the nonzero elements of Q and, in the sequel, replace all $1 \le i \le n$ with α_i.

Let $T = (t_1, t_2, \ldots, t_n)$, where $t_i \in Q$, $1 \le i \le n$. If the points (i, t_i) can be interpolated by a polynomial of degree d, we say simply that T *can be interpolated by a polynomial of degree d*. (In the case of a general field Q, we would interpolate the points (α_i, t_i).)

Let $\tau = \max\{\sigma, \rho\}$. In Theorem 5.2, we prove that 2-way (σ, ρ)-SMT requires connectivity $n \ge \tau + \rho + 1$. Henceforth, assume $n = \tau + \rho + 1$. Then, since Sender and Receiver are at least $2\rho + 1$ connected, they are essentially connected by a fault-free public channel. To send a message x over this public channel, Sender can simply send x on every wire, that is, x is replicated $n \ge 2\rho + 1$ times, and at most ρ of these copies will be destroyed or modified. Thus, Receiver can simply see which message appears at least $\rho + 1$ times, and that is the message that Sender sent. Similarly, Receiver can send things to Sender in a fault-free, but public, fashion.

The slow protocol works as follows: Let $m \in Q$ be the secret message Sender wishes to send to Receiver. First, Sender chooses uniformly at random a pad $p \in Q$; p bears no relation to m. Sender *attempts* secret transmission of p. If secret transmission of p is successful, Sender will send $Z = p \oplus m$ to Receiver over the "public channel." In this case, Receiver computes $m = Z \ominus p$ (all arithmetic is done in the field Q). If the secret transmission of p fails, then Sender and Receiver will use the public channel to detect at least one previously undetected faulty wire, and the entire protocol is repeated but without the detected faulty wires. During the error detection the secrecy of the

pad p is lost; however, since p was chosen independently of m this yields no information about m.

There are two drawbacks to this general approach. First, Sender may have to attempt to transmit up to $\rho + 1$ times. Second, the faulty wires D cannot change between phases. Our three-phase solution will overcome both of these drawbacks.

We now present the slow protocol.[3] The input W is a set of labeled wires. Note that we have labeled the wires from 1 to k (rather than from 0 to $k - 1$ as in the impossibility proofs). Initially, $k = n$, but in the recursive calls $k < n$. This is because faulty wires are eliminated from the set of wires as errors are detected. All missing or syntactically incorrect messages are treated as zeros. Steps labeled "S" (respectively, "R") are taken by the Sender (respectively, Receiver).

SlowSMT($W = \{w_1, \ldots, w_k\}, \tau, Q$, **private to** S:m)
PHASES 1 and 2:
S: Choose random polynomial $f(x) \in Q(x)$ of degree τ.
Send $s_i = f(i)$ on w_i. Let $p = f(0)$.
R: Let $T = (t_1, \ldots, t_k)$ where $t_i \in Q$ is received on w_i.
IF T can be interpolated by a degree τ polynomial g
Then set $p = g(0)$ and publicly send "OK" to Sender
Else publicly send T to Sender
FI

PHASE 3:
S: IF "OK" received over the public channel in Phase 2,
Then send $Z = p \oplus m$ to Receiver over the public channel.
Else DO:
For each j such that $t_j \neq s_j$, remove w_j from W.
Send new W to Receiver over the public channel.
Call SlowSMT(new W, τ, Q, m).
OD
FI
R: IF "OK" sent to Sender in Phase 2
Then receive Z on the public channel and compute $m = Z \ominus p$.
Else DO.
Receive new W from Sender over the public channel.
Call SlowSMT(new $W, \tau, Q, \cdot$).
OD
FI
End of SlowSMT

Protocol SlowSMT can be speeded up a bit by overlapping the last transmission in a given invocation with the first transmission in the recursive call, but this is not of interest. Our point in presenting this protocol is to develop certain techniques that will be used later.

CLAIM 4.1. *SlowSMT satisfies the resiliency and secrecy conditions for 2-way* (σ, ρ)*-SMT.*

PROOF. Let us call each recursive call to SlowSMT an iteration of the protocol. Initially, SlowSMT is called with all $n = \tau + \rho + 1$ wires and secrecy parameter $\tau = \max\{\sigma, \rho\}$. Thus, in the first iteration, there exist good wires $x_1, \ldots, x_{\tau+1}$ such that $t_{x_i} = s_{x_i}$ for all $1 \leq i \leq \tau + 1$. These values completely

[3] Our slow algorithm was originally more complex. The simplified version described here is due to Rabin and Ben-Or [20].

determine the degree τ polynomials g and h, so if Receiver says "OK" in Phase 2, then $g(\cdot) = h(\cdot)$.

Conversely, if $t_j \neq s_j$, then wire j is faulty. Thus, only truly faulty wires are removed from W by Sender. A simple induction shows that in every subsequent iteration there exist good wires $x_1, \ldots, x_{\tau+1}$ such that $t_{x_i} = s_{x_i}$ for all $1 \leq i \leq \tau + 1$.

Since eventually Receiver either says "OK" or there are more than ρ faulty wires, the resiliency condition is satisfied.

For secrecy, we first observe that the initial and all recursive calls have the same secrecy parameter τ, so we are always using polynomials of degree τ, no matter how many faulty wires are eliminated. Thus, for any successfully transmitted pad p, the shares s_i are τ-wise independent and uniformly distributed over Q, independent of the value of m. Moreover, they are completely independent of all previous pads.

Since we are working under the containment assumption (or $\mathscr{A}_D$ is oblivious), $\mathscr{A}_L$ learns absolutely nothing about the secret from the number of phases required, or from the choice of D, since these variables are controlled by $\mathscr{A}_D$ and $\mathscr{A}_L$ has access to all the shares available to $\mathscr{A}_D$. Thus, if p is successfully transmitted, it is secret according to the definition of secrecy for (σ, ρ)-SMT.

A little more formally, fix a secret message $m \in Q$. Let i be any integer $0 \leq i < |Q|$. We claim that for any view V_L of $\mathscr{A}_L$, V_L occurs with the same probability in a transmission of m as in a transmission of $m' = m \oplus i$. Clearly, for every destroyed pad this is true, since the destroyed pads are chosen and destroyed without relation to m or to the values of shares on wires outside L. Consider the first phase in which the pad is not destroyed, and let f be the random degree τ polynomial chosen by sender in this phase (the pad is $f(0)$). The choice of which shares of f are seen is independent of $f(0)$. In fact, $\mathscr{A}_L$'s view is equally likely in the case that chosen polynomial is f', where every share of f' corresponding to wire in L is identical to the corresponding share of f, but $f'(0) = f(0) \ominus i$. Thus,

$$z = m \oplus f(0) = (m \oplus i) \oplus f'(0) = m' \oplus f'(0),$$

appears (on the public channel) with the same probability given $\mathscr{A}_L, \mathscr{A}_D$, and given that the message is m as it does when the message is $m \oplus i$. □

We now describe a variant of SlowSMT requiring optimal connectivity $n = \tau + \rho + 1$, which, when run with error parameter k, achieves (σ, ρ)-SMT in three phases with probability $1 - \rho e^{-k/2}$. Here, Sender chooses $l = 2\rho k$ random pads $p_1, p_2, \ldots, p_l$, each one chosen exactly as was done in SlowSMT, and sends each one as a vector $S_i = (s_{i1}, s_{i2}, \ldots, s_{in})$ exactly as was done in SlowSMT. For each sent vector S_i corresponding to random pad p_i, let T_i be the actual vector of shares received. If any T_i can be interpolated by a polynomial of degree τ, then Receiver sends "OK; i" over the public channel and the algorithm proceeds as in SlowSMT with a successfully transmitted pad.

In no vector T_i can be interpolated by a degree τ polynomial, then Receiver chooses at random $\rho k = l/2$ of the received vectors $T_{r_1}, \ldots, T_{r_{\rho k}}$ and sends all these vectors to Sender on the public channel, together with the indices $r_1, \ldots, r_{\rho k}$. Sender does error detection on the returned vectors and publicly sends to Receiver the set of detected faulty wires, together with $z_i = p_i \oplus m$ for each index $i \notin \{r_1, \ldots, r_{\rho k}\}$.

For each remaining (unreturned) vector T_i, Receiver deletes from T_i all shares t_{ij} for wires j declared faulty by Sender. If the resulting "corrected" vector T_i can be interpolated then Receiver computes $m = z_i \ominus p_i$.

CLAIM 4.2. *With probability at least* $1 - \rho e^{-k/2}$ *at least one unreturned, corrected vector* T_i *can be interpolated.*

PROOF. Let F be the set of faulty wires w such that w corrupted at least k pads. For any $w \in F$, if a pad P is chosen at random from among the failed pads, then the probability that w corrupted P is at least $1/2\rho$. Thus, if a set H of ρk pads are chosen at random, the probability that w did *not* corrupt any pad in H is bounded by $(1 - 1/2\rho)^{\rho k} \approx e^{-k/2}$. Finally, since there are only ρ faulty wires, the probability that there exists a wire $w \in F$, such that w does not corrupt any pad in H, is at most $\rho e^{-k/2}$.

For all wires $u \notin F$, u corrupted strictly fewer than k pads (by definition of F). Thus, together, the set of all wires not in F collectively corrupted fewer than ρk pads. It follows that at least one of the ρk vectors T_i that Receiver does not return to Sender for error detection was corrupted only by wires in F. Let T be such a vector. Since with probability at least $1 - \rho e^{-k/2}$ every wire in F corrupted at least one of the returned vectors, the probability that every wire in F is detected is at least $1 - \rho e^{-k/2}$. In this case, Receiver will be able to remove all the corrupted shares of T and the corrected T will interpolate correctly.

Secrecy for this three phase algorithm is argued as above. The fact that $\mathscr{A}_D$ is oblivious or $D \subseteq L$ prevents $\mathscr{A}_L$ from learning any information about shares of retained pads on wires not in L from the indices of the pads that were sent back or from the text sent and the faults discovered. □

Our error-free 3-phase algorithm is very similar to the algorithm just described. However, we "strengthen" each random pad by sending, in addition to the shares of a random polynomial, some additional "checking" information. This technique has appeared several times in the literature in the context of verifiable secret sharing and other problems in distributed computing (see, e.g., [3, 4, 12, 13, 20]). After describing the stronger pads, we show that if sufficiently many ($\rho n + 1$) pads are sent then either at least one succeeds (essentially defined as before) or it is possible for Receiver to choose a set of ρn pads to return to Sender such that all the faulty shares of the one retained pad belong to wires whose faultiness will be detected by Sender in the ρn returned pads. Thus, the faulty shares of the retained pad will be removed and the remaining shares can be interpolated.

To send a random pad to Receiver, Sender again chooses a random polynomial $f(x) \in Q(x)$ of degree τ and sets $p = f(0)$. For each i, $1 \le i \le n$, we call $f(i)$ a *principal share* of the pad. For each $1 \le i \le n$ (recall $n = \tau + \rho + 1$), Sender chooses an additional random degree τ polynomial $h_i(x) \in Q(x)$ satisfying $h_i(0) = f(i)$. Recall that in the more simple pads, $s_i = f(i)$ is what Sender sends on wire i. In the stronger pad, Sender sends the entire polynomial $h_i(\cdot)$ together with a vector $C_i = (c_{1i}, c_{2i}, \ldots, c_{ni})$ of *checking pieces*, satisfying, for all $1 \le i, j \le n$, $c_{ji} = h_j(i)$. (To send $h_i(\cdot)$, Sender need only send the $\tau + 1$ coefficients of h_i.) Since $h_i(0) = f(i) = s_i$, the wire carries all the information it carried in the simpler pad, plus additional checking information.

Throughout this discussion, we let h_i, C_i denote the information placed by Sender on wire i, and we let g_i, D_i denote the (possibly corrupted) information

received by Receiver on wire i. Consider attempted transmission of a single strong pad. Let T be the received information. If wire i is correct, then $g_i = h_i$ and $D_i = C_i$. (This just says that if wire i is correct then what is received on wire i is the same as what is sent on wire i.) Thus, if i and j are both correct wires, then $d_{ji} = g_j(i)$ (because i correct implies $d_{ji} = c_{ji}$; j correct implies $g_j = h_j$; and by construction $c_{ji} = h_i(i)$).

If $d_{ji} \neq g_j(i)$, we say the unordered pair (i, j) is a *conflict of* T. Clearly in case of a conflict (i, j) at least one of i and j is faulty.

When Sender attempts to send a strong pad to Receiver, Receiver is said to "throw out" (ignore) all wires j carrying syntactically incorrect messages. In particular, if g_j is not a polynomial of degree τ, then Receiver throws out wire j.

A strong pad is said to *fail* if for all wires i not thrown out, the points $(i, g_i(0))$ cannot be interpolated by a degree τ polynomial. Otherwise, it *succeeds*, regardless of conflicts.

We now describe the three-phase protocol FastSMT. The parameters of FastSMT are the same as those for SlowSMT.

FastSMT(W, τ, Q, **private to** S:m)
Phases 1 and 2:
S: Send $n\rho + 1$ strong pads $P_1, P_2, \ldots, P_{n\rho+1}$.
R: For $1 \leq i \leq n\rho + 1$
Let T_i be received in the attempted transmission of P_i.
Ignoring those wires thrown out,
IF any T_a succeeds
Then compute P_a from T_a and publicly send "a, OK" to Sender
Else find an i such that
{conflicts of T_i} $\subseteq \bigcup_{j \neq i}$ {conflicts of T_j}.
Publicly send "i" and all T_j, $j \neq i$, back to Sender.
FI
PHASE 3:
S: IF "a, OK" received over the public channel in Phase 2,
Then send $Z = P_a \oplus m$ to Receiver over the public channel.
Else perform error detection on all T_j received from Receiver and publicly send detected faults and $Z = P_i \oplus m$ to Receiver.
FI
R: IF "a, OK" sent to Sender in Phase 1,
Then compute $m = Z \ominus P_a$.
Else correct retained T_i to obtain P_i; compute $m = Z \ominus P_i$.
FI
End of FastSMT

LEMMA 4.1. *If all* $n\rho + 1$ *strong pads fail, then there exists an* i *such that*

$$\{\text{conflicts of } T_i\} \subseteq \bigcup_{j \neq i} \{\text{conflicts of } T_j\}.$$

PROOF. With ρ faulty wires there can be at most ρn conflicts (distinct pairs (i, j), not counting multiplicities). The lemma follows by a pigeon hole argument. □

For every conflict (x, y) of the retained T_i, (x, y) is a conflict of some returned T_j, so Receiver learns of the faultiness of at least one of x and y. Of course, both may be faulty. Without loss of generality, suppose the Sender detects that x is faulty, and publicly sends this information to the Receiver. The next lemma says that even if y is faulty, if Sender did not also identify y as

faulty, then the principal share of the retained pad reported by y is the correct share of that pad.

LEMMA 4.2. *Let P be the retained pad (P_i in the algorithm, but we eliminate the subscripts for ease of discussion). For every wire y that is neither thrown out nor detected faulty by Sender, $g_y = h_y$.*

PROOF. Let y be undetected and not thrown out. Since y is not thrown out g_y is a polynomial of degree τ. Let $x_1, \ldots, x_{\tau+1}$ be nonfaulty wires. If for all $1 \le i \le \tau + 1$, $g_y(x_i) = d_{yx_i} (= c_{yx_i} = h_y(x_i)$ because x_i is good), then $g_y = h_y$. Let us assume, for the sake of contradiction, that $g_y \neq h_y$. In this case $g_y(x_i) \neq d_{yx_i}$ for some nonfaulty x_i, whence (y, x_i) is a conflict of T. By choice of T (T_i in the protocol), (y, x_i) is a conflict of some other strong pad $T' \neq T$, so Sender detects the faultiness of at least one of y, x_i. However, x_i is nonfaulty, so Sender detects of the faultiness of y, contradicting the assumption that y is not detected. □

That FastSMT satisfies the resiliency condition of (σ, ρ)-SMT follows from the last two lemmas. Secrecy is argued essentially as it was for SlowSMT, with the further observation that for every share $h_j(\cdot)$ of a strong pad and checking pieces c_{ji} are τ-wise independent and uniformly distributed over Q. We therefore have

THEOREM 4.1. *Under the containment assumption, or if $\mathscr{A}_D$ is oblivious, there is a three phase error-free protocol for (σ, ρ)-SMT requiring connectivity $\sigma + \rho + 1$ and communication polynomial in n.*[4]

The communication complexity can be improved somewhat. The improvements involve using bivariate polynomials as done in [12] and [13] to represent the checking pieces more compactly. A second improvement yields a reduction in the number of strong pads to $\rho^2 + 1$ (from $n\rho + 1$). This is because if a wire j is involved in conflicts with more than ρ distinct wires, then Receiver can detect the faultiness of j without sending anything to Sender. Let Receiver throw out all such wires j. The remaining wires can yield at most ρ^2 conflicts (ignoring multiplicities), so the same counting argument as above shows that if no strong pad succeeds Receiver can still find a T_i whose (remaining) conflicts are contained in the union of the (remaining) conflicts of the T_j, $j \neq i$.

For completeness, we briefly describe our 2-phase protocol requiring connectivity $\sigma + \rho + 1$ under containment or when $\mathscr{A}_D$ is oblivious. The protocol begins with the Receiver sending an enormous number z of strong pads to the Sender. The intent is to send enough pads to that, if all are destroyed by $\mathscr{A}_D$, then there will be $k = \binom{n}{\rho}$ pads, renumbered $P_1, \ldots, P_k$, and additional pads $R_1, \ldots, R_{z-k}$ such that for each i

$$\{\textit{conflicts}(P_i)\} \subseteq \bigcup_{j=1}^{z-k} \{\textit{conflicts}(R_j)\}.$$

Something similar was done in Algorithm FastSMT to obtain a single retained pad whose conflicts were all covered by the conflicts of the returned pads. Here, because we need k retained pads, we let $z = n\rho + k$.

[4] If $\mathscr{A}_D$ is not oblivious and $\rho \ge \sigma \ge 0$, then under the containment assumption $2\rho + 1$ wires suffice.

Sender chooses the k pads to retain, renumbers them $P_1, \ldots, P_k$, and computes a set of possible "fault sets" of wires as follows. Let $\mathcal{F} = \{F_1, \ldots, F_{k'}\}$, for some $k' \leq k$, such that the sets $F_i \in \mathcal{F}$ are precisely those of cardinality ρ with the property that by removing all principal shares corresponding to wires in F_i, Sender can interpolate all the z pads received. For each of the first k' retained pads P_i, $1 \leq i \leq k'$ Sender computes the value of the pad P_i obtained by deleting the principal shares received on wires in F_i. Let v_i be the value obtained. Letting m be the secret message, Sender sends to Receiver over the public channel:

(1) a list of the indices of the retained pads (before renumbering);
(2) the ordered set $\mathcal{F}$,
(3) $s_i = v_i \oplus m$ for each $1 \leq i \leq k'$,
(4) everything received in the attempted transmission of the pads $R_1, \ldots, R_{z-k}$ (these are the "returned pads").

Receiver does error detection on the set of returned pads to obtain a set F of faulty wires. It finds an $F_j \in \mathcal{F}$ such that $F \subseteq F_j$ (we must argue one exists), and subtracts from s_j the actual value encoded by the jth pad listed in item (1). We argue that the resulting value is the message m.

CLAIM 4.3. *Let G be the set of wires that actually destroyed at least one primary share in any of the z strong pads transmitted. Then for every extension $G^* \supseteq G$ of size ρ, $G^* \in \mathcal{F}$.*

PROOF. Consider any received pad $\tilde{P}$. Since no wire not in G destroyed a primary share, deleting all the primary shares of wires in G from $\tilde{P}$ allows the remaining shares in this pad to be interpolated by a polynomial of degree $\tau \geq \sigma$. All these shares are correct, so any subset of $n - \rho \geq \tau + 1$ of these shares can be interpolated to yield the same value. Thus, every extension G^* of G to a set of size ρ has the property that deleting all the primary shares of wires in G^* allows all received pads to be interpolated, so every such G^* is in $\mathcal{F}$. □

CLAIM 4.4. *$\exists j$ such that $F \subseteq F_j$.*

PROOF. Let G be as in the previous claim. If $|G| = \rho$, then we are done, since every wire in G is faulty and every wire in F is faulty and there are only ρ faulty wires in total. If $|G| < \rho$, then by the proof of the previous claim every extension G^* of G to a set of size ρ is in $\mathcal{F}$. Now, by Lemma 4.2 all wires that destroyed at least one primary share in any of the retained pads $P_1, \ldots, P_k$ are in F. Moreover, F also contains any wire that destroyed a primary share in any of the returned pads, since all information received by Sender for those pads was returned over the public channel and used in the fault detection. Thus, $F \supseteq G$, so F is an extension of G of size at most ρ. Since every extension of G to a set of size ρ is in $\mathcal{F}$, every extension of F to a set of size ρ is contained in $\mathcal{F}$. □

Let F_j be as in Claim 4.4. By Lemma 4.2, deleting all shares of retained pad P_j received on wires in F_j deletes all corrupted primary shares of P_j and leaves intact at least $n - \rho \geq \tau + 1$ correct shares of P_j. Thus, the value v_j computed by Sender is the correct value of the original pad, and we are done.

Secrecy hinges on three facts: the pads are mutually independent; for any given retained pad P_i there is only one fault set F_i for which Sender reveals $v_i \oplus m$; for every retained pad P_i, $\mathscr{A}_L$ has access only to $\sigma \leq \tau$ shares (both in the containment case and in the case $\mathscr{A}_D$ is oblivious). This completes our discussion of the 2-phase protocol.

5. *Tight Bounds for the Containment Case*

In this section, we restrict attention to the containment case. Generally, we assume $\sigma \geq \rho$. When $\mathscr{A}_D$ is not oblivious, if $\rho > \sigma \geq 1$ and $L \subset D$, then $\mathscr{A}_D$ can always inform $\mathscr{A}_L$ of all the traffic on the wires in D, either by communicating explicitly to $\mathscr{A}_L$ through a back channel or by writing its entire view on one of the jointly compromised wires. The situation is then as if there were ρ listening wires, all of which could be disruptors, and the lower and upper bounds for (ρ, ρ)-SMT apply.

All the results of this section apply without the containment assumption provided $\mathscr{A}_D$ is oblivious. The upper bounds hold because $\mathscr{A}_D$ cannot communicate to $\mathscr{A}_L$ any information about the conversations on wires not in L. The lower bounds hold because even an $\mathscr{A}_D$ that disrupts completely at random could generate the scenarios leading to erroneous outcomes that will be used in those proofs.

Disruptor-free executions are critical to many of our lower bound proofs. The proofs are by contradiction. We assume the existence of a protocol with a certain amount of connectivity. The protocol must work even against an empty disrupting adversary. We study the protocol with this adversary to learn about its structure and the types of messages Sender and Receiver must send. We then define an $\mathscr{A}_D$ that is chosen accordingly and force an erroneous outcome.

LEMMA 5.1. *Let P be any protocol for weakened 1-way (σ, ρ)-SMT. Then the information sent on any $n - 2\rho$ wires completely determines the secret.*

PROOF. Let $n = \alpha + \beta + \gamma$, where $1 \leq \alpha \leq \rho$, $n - 2\rho \leq \beta < n$ and $0 \leq \gamma \leq \rho$. We say an n-vector *encodes* a value m if in some execution of P, when Sender begins with message m it places the ith component of V on wire i, $0 \leq i \leq n - 1$.

Suppose, for the sake of contradiction that the Lemma is false. Without loss of generality, there exist values $m \neq m' \in Q$ and vectors V, V' encoding m and m', respectively, such that $V = XYZ$, where $X \in \Sigma^\alpha$, $Y \in \Sigma^\beta$, and $Z \in \Sigma^\gamma$ and $V' = X'YZ'$, where $X' \in \Sigma^\alpha$ and $Z' \in \Sigma^\gamma$ (Y remains unchanged). The point here is that Y is a subvector of at least $n - 2\rho$ components that does not determine the secret, since Y occurs in an encoding of m and in an encoding of m'.

Let $W = XYZ'$, where X is as in V, Y is as in both V and V', and Z' is as in V'.

Now, $dist(W, V) \leq \rho$, so by the resiliency requirement if Receiver receives W it must output m. However, $dist(W, V') \leq \rho$, so Receiver must output m', a contradiction. □

COROLLARY 5.1. *Weakened 1-way (σ, ρ)-SMT under the containment assumption requires $\sigma < n - 2\rho$, that is, $n \geq \sigma + 2\rho + 1$.*

PROOF. It follows from Lemma 5.1 that if $\sigma \geq n - 2\rho$, then listening to any σ wires not including wire 0 yields the secret. Thus, by σ-secrecy, $\sigma < n - 2\rho$, whence $n \geq \sigma + 2\rho + 1$. □

It is surprising that the issue of containment did not arise in the proof of the lower bound. Intuitively, this is because the Sender does not know in advance which wires $\mathcal{A}_L$ and $\mathcal{A}_D$ will compromise. Thus, it must simultaneously protect against disruption on any ρ wires (in which case we let $L = D$ so $D \subseteq L$) and listening on any σ wires (in which case we let $D = \emptyset$ so again $D \subseteq L$), even if at most one of these adversaries attacks in any single execution.

THEOREM 5.1. *Under the containment assumption, connectivity* $n = \sigma + 2\rho + 1$ *is necessary and sufficient for* 1*-way* (σ, ρ)*-SMT.*[5]

PROOF. The lower bound is immediate from Corollary 5.1. McEliece and Sarwate [18] observed that Shamir's scheme [21] for sharing secrets is closely related to Reed-Solomon coding schemes. As McEliece and Sarwate point out, the errors-and-erasures decoding algorithms for these codes can be used to detect and correct up to ρ errors, provided the codewords are of length $\sigma + 2\rho + 1$ and the polynomials used in the construction of the codewords are of degree σ. A similar observation was made by Ben-Or, et al. [3] who used these codes in constructing their solution to the harder problem of Verifiable Secret Sharing. □

We now turn to lower bounds on connectivity for the 2-way case. We begin with a technical lemma that hinges on our assumption that the random choices of Sender and Receiver are made by coin flipping, which yields only bounded branching. An alternative would be to allow unbounded branching at each choice node in the computation tree. Although all our results hold in this model as well, the proofs are more difficult.

LEMMA 5.2. *Let P be a protocol for 2-way SMT. Then there exists an upper bound B on the number of phases in any disruptor-free execution of P.*

PROOF. Since we require perfect resiliency, P cannot have infinite disruptor-free executions. Consider executions of P with an empty disrupting adversary. Fix an input to the sender and consider the tree of all possible coin-flip sequences. Since the random choices of Sender and Receiver are made by flipping coins, this tree has bounded branching. Suppose there is no bound B on the length of any path in the tree. Then at least one child of the root is itself the root of a tree of unbounded depth. We can continue down the tree in this fashion forever, but the execution corresponding to the path we follow does not terminate, violating correctness. □

THEOREM 5.2. *Let P be any protocol for* 2*-way* (σ, ρ)*-SMT. Then P requires connectivity* $n \geq max\{\sigma + \rho + 1, 2\rho + 1\}$*, even under the containment assumption.*

PROOF. The condition $n \geq \sigma + \rho + 1$ is needed for σ-secrecy, even if $\rho = 0$. Specifically, we show that in a disruptor-free execution any $n - \rho$ wires must contain enough information to completely determine the secret. It follows

[5] If $\rho > \sigma \geq 1$ and $\mathcal{A}_D$ is not oblivious, then, as explained at the beginning of this section, the bound becomes $3\rho + 1$. If $\sigma = 0$, then $2\rho + 1$ wires suffice.

that if $\sigma \geq n - \rho$, then listening to any σ wires yields the secret. Thus, $\sigma < n - \rho$, whence $n \geq \sigma + \rho + 1$.

Suppose for the sake of contradiction that in some disruptor-free execution E of P, there is a set Z of $n - \rho$ wires such that the information sent over Z in E is insufficient to determine m, the message being sent. This is equivalent to saying there exists a message m' and a disruptor-free execution E' transmitting m', such that E and E' have exactly the same conversations on the wires in Z. Let B be the upper bound on the number of phases of any disruptor-free execution of P given by Lemma 5.2. Throughout the proof of this theorem we assume without loss of generality that B is odd and the first phase is from Sender to Receiver.

Let C_S (C_R) denote the random coin tosses of Sender (Receiver) in E, and C'_S (C'_R) denote the random coin tosses of Sender (Receiver) in E'. Let us describe the communication in E as

$$\begin{array}{c} \alpha_1\, \beta_1 \\ \alpha_2\, \beta_2 \\ \vdots \\ \alpha_B\, \beta_B, \end{array}$$

where α_i represents what is sent in phase i over the wires in Z, and β_i represents what is sent in phase i over all the wires not in Z. Then for some $\gamma_1, \ldots, \gamma_B$, the communication in E' can be described as

$$\begin{array}{c} \alpha_1 \gamma_1 \\ \alpha_2 \gamma_2 \\ \vdots \\ \alpha_B \gamma_B, \end{array}$$

where the α_i's are as before and the γ_i's denote the communication during E' in phases i over wires not in Z. We now construct an execution E^* in which the ρ wires not in Z are faulty. Let us call them D. The notation $(x \to y)$ means that x is placed on the wires in D, but these wires (erroneously) transmit y instead. In E^*, Sender wishes to send message m and has random flips C_S, while Receiver has flips C'_R. The conversations are:

$$\begin{array}{cc} \alpha_1 & (\beta_1 \to \gamma_1) \\ \vdots & \vdots \\ \alpha_{2i} & (\gamma_{2i} \to \beta_{2i}) \\ \alpha_{2i+1} & (\beta_{2i+1} \to \gamma_{2i+1}) \end{array}$$

In other words, the wires in D behave towards Sender as if they are in execution E: in even rounds $2i$ they transmit β_{2i}, while behaving towards Receiver as if they are in execution E': in odd rounds $2i + 1$ they transmit γ_{2i+1}. Since Sender cannot distinguish E^* from E, it does not send after phase B. Since Receiver cannot distinguish E^* from E', it outputs m', violating correctness.

Notice that the set L of wires compromised by $\mathscr{A}_L$ is not mentioned in the description of E^*, and can therefore be arbitrary. It follows that the proof holds even under the containment assumption.

The condition $n \geq 2\rho + 1$ is needed for ρ-resiliency, even if $\sigma = 0$. Intuitively, we see that if $n = 2\rho$ then half the wires can "behave as if" the input to Sender is some value m, and the other half can "behave as if" the input is some $m' \neq m$, and Receiver cannot tell which is the true input.

Assume, for the sake of contradiction, that there exists a protocol P for 2-way $(0, \rho)$-SMT requiring connectivity 2ρ. Let $m \neq m' \in Q$. We construct two executions E and E' of P that, for every k, are indistinguishable to Receiver after k phases: it has the same coin flip sequence and sees exactly the same messages in each execution. However, in E the secret is m, while in E' the secret is m'. Thus, these executions cannot terminate, violating resiliency. We define the executions in parallel, phase by phase. In the following, the α's, γ's, and x's are always placed on wires $0, 1, \ldots, \rho - 1$, and the β's, δ's, and y's are placed on wires $\rho, \ldots, 2\rho - 1$. In E, for all $0 \leq i$, let $\alpha_{2i+1}\beta_{2i+1}$ be sent by Sender in Phase $2i + 1$, and let $\gamma_{2i+1}\delta_{2i+1}$ be sent by Sender in Phase $2i + 1$ of E'. The executions begin

E:		E':	
α_1	$(\beta_1 \to \delta_1)$	$(\gamma_1 \to \alpha_1)$	δ_1
x_2	y_2	x_2	y_2
$\vdots$	$\vdots$	$\vdots$	$\vdots$
α_{2i+1}	$(\beta_{2i+1} \to \delta_{2i+1})$	$(\gamma_{2i+1} \to \alpha_{2i+1})$	δ_{2i+1}
$x_{2(i+1)}$	$y_{2(i+1)}$	$x_{2(i+1)}$	$y_{2(i+1)}$

Clearly, since Receiver cannot distinguish the two executions Sender must continue, and the executions run forever, violating the resiliency requirement.

We can actually extend the proof to demonstrate an adversary that can force *every* execution to run forever with probability depending only on Q and Π, the underlying probability distribution on messages. The construction is as above. $\mathscr{A}_D$ only disrupts during odd phases. We will describe, for each $i \geq 0$ what the adversary does in phase $2i + 1$. For every $j \geq 0$, we let E_j (respectively, E'_j) denote the first j phases of execution E (respectively, E'). Let m' be a message of minimal probability according to Π. As above, let C_S and C_R denote the coin flip sequences of Sender and Receiver in E, and let m be the Sender's secret in E.

$\mathscr{A}_D$ will decide how to disrupt during E by simulating a random instance E' of the protocol, in which Sender has input m' and randomly chosen coin flip sequence C'_S, Receiver has coin flip sequence C_R, and there is a different adversary $\mathscr{A}'_D$. It will not be necessary for $\mathscr{A}_D$ to know C_R, since $\mathscr{A}_D$ will arrange for E and E' to be indistinguishable to Receiver, and therefore Receiver's transmissions in E and E' will be identical.

Assume inductively that E_{2i} and E'_{2i} are indistinguishable to Receiver. This clearly holds for $i = 0$. For $i \geq 0$, let $\alpha_{2i+1}\beta_{2i+1}$ be generated by Sender at phase $2i + 1$ of E. By simulating Sender with history E'_{2i}, $\mathscr{A}_D$ can compute $\gamma_{2i+1}\delta_{2i+1}$ generated by Sender in E'. During phase $2i + 1$ of E, $\mathscr{A}_D$ replaces β_{2i+1} with δ_{2i+1}. During phase $2i + 1$ of E', $\mathscr{A}'_D$ replaces γ_{2i+1} with α_{2i+1}. Thus,

$$E_{2i+1} = E_{2i} \cdot [\alpha_{2i+1}(\beta_{2i+1} \to \delta_{2i+1})],$$
$$E'_{2i+1} = E'_{2i} \cdot [(\gamma_{2i+1} \to \alpha_{2i+1})\delta_{2i+1}].$$

Since E'_{2i+1} is indistinguishable to Receiver from E_{2i+1}, Receiver gives the same response in phase $2i + 2$ of E'. Note that although the actual communication is the same in E and E', the executions are not identical: In general, they are distinguishable to Sender. However, since Receiver has the same coin flip sequence and sees the same messages in the two executions, it cannot distinguish them. This completes the induction.

We have therefore shown that with connectivity 2ρ it is not even possible to solve 2-way $(0, \rho)$-SMT with probability one. □

6. *Perfect and Imperfect Secret Sharing: A Separation Result*

In this section, we show that (σ, ρ)-Unverified Secret Sharing requires $\sigma + 2\rho + 1$ processors.[6] This bound can be achieved (see the proof of Theorem 5.1) [3, 18]. Rabin and Ben-Or show that, for any k, t-USS can be achieved with $2t + 1$ processors with probability at most 2^{-k} of error [20]. An immediate generalization of their result shows that with finite but arbitrarily small probability of error, (σ, ρ)-USS can be achieved with $\sigma + \rho + 1$ processors.

As we demonstrate, (σ, ρ)-SMT cannot be solved deterministically. It follows that, within the class of problems that have no deterministic solution, error-free computation comes at a price (in this case, an extra ρ processors). It is therefore possible to separate error-free randomized computation from small-error randomized computation.

Our $\sigma + 2\rho + 1$-processor lower bound for (σ, ρ)-USS holds even in the model with a broadcast channel. It follows that t-Verifiable Secret Sharing requires at least $3t + 1$ processors, even in the presence of a broadcast channel. This result has been claimed elsewhere [4, 20]. Because our lower bound applies to the weaker problem of t-USS, our result is stronger. Moreover, it follows from our result that the processor cost of Verifiable Secret Sharing has nothing to do with verification, but rather comes from the conflicting requirements of secrecy and resiliency.

In keeping with the literature on secret sharing, the results in this section are for the case in which the containment assumption holds. We state the results for the case $\sigma \geq \rho$. The general results can be obtained by replacing every occurence of "σ" by "$\max\{\sigma, \rho\}$." In addition, all our results hold without the containment assumption if $\mathscr{A}_D$ is oblivious.

LEMMA 6.1. *(σ, ρ)-USS requires at least $2\rho + 1$ processors even in the presence of a broadcast channel, and even under the containment assumption.*

PROOF. We assume for the sake of contradiction that there exists a pair of protocols $(\mathscr{P}_1, \mathscr{P}_2)$ solving (σ, ρ)-USS and requiring only 2ρ processors total. Let $V = XY$ be a vector of histories of processors $p_0, \ldots, p_{2\rho+1}$ at the end of a disruptor-free execution E of $\mathscr{P}_1$ in which p_0 has input m. Here, X is the first ρ components of V and Y is the last ρ components of V. Let $m' \neq m$ be arbitrary. By the secrecy constraint, since $\rho \leq \sigma$, there exists a disruptor-free execution E' of $\mathscr{P}_1$ in which p_0 has input m' and the resulting vector of histories is $V' = XZ$. Consider an execution F of $\mathscr{P}_2$, extending E, in which processors $p_\rho, \ldots, p_{2\rho-1}$ are faulty, begin $\mathscr{P}_2$ with the states determined by Z, and make no further errors. Processors $p_0, \ldots, p_{\rho-1}$ begin F with the states

[6] We assume $\sigma > 0$, since secret sharing makes no sense if there is no secrecy requirement.

determined by X and make no errors. At the end of F every correct processor must output m, since this was the input to p_0 in E, and p_0 did not fail during E. On the other hand, F must yield m', since F is also a valid extension of E' in which all processors are correct. □

We now describe the relationship between (σ, ρ)-USS and 1-way (σ, ρ)-SMT under containment.

THEOREM 6.1. *Any n processor solution to (σ, ρ)-USS with or without a broadcast channel, yields a connectivity n solution to weakened 1-way (σ, ρ)-SMT under containment.*

PROOF. By Lemma 6.1, $n \geq 2\rho + 1$. Let the wires be labelled $0, 1, \ldots, n - 1$. Let the processors be $p_0, \ldots, p_{n-1}$. To send a message m, the Sender first simulates a disruptor-free execution E of $\mathscr{P}_1$ in which p_0's input is m. Letting v_i denote the complete history of p_i in E, for $1 \leq i \leq n - 1$, Sender places v_i on wire i. Let $V = (v_0, \ldots, v_{n-1})$. Let $W = (w_0, \ldots, w_{n-1})$ denote the vector of histories received by the Receiver. By assumption, $dist(V, W) \leq \rho$. To compute the message encoded by the vector W, Receiver simulates that execution of $\mathscr{P}_2$ in which each processor p_i begins in the state given by w_i and no further disruption occurs. This results in a set of outputs, one for each p_i. Receiver outputs that value which is output by a majority of processors in the simulation.

To see that the secrecy condition is met, we have by the secrecy of (σ, ρ)-USS that no set of σ processors not containing p_0 has any information about the message m before execution of $\mathscr{P}_2$. Let L be any set of at most σ wires compromised by $\mathscr{A}_L$. Since $D \subseteq L$, the only information about m available to $\mathscr{A}_L$ is the subvector of V containing the views of the processors corresponding to the wires in L. By definition of secrecy for (σ, ρ)-USS, if $0 \notin L$ then for every $m' \in Q$ this view appears with the same probability with secret m as with secret m'. On the other hand, if $0 \in L$, weakened 1-way SMT has no secrecy requirement at all.

To see that the resiliency condition is met, we first note that since at most ρ wires are compromised by $\mathscr{A}_D$, a majority of the wires are not compromised. Thus, in the simulation, the Receiver is simulating at least $n - \rho \geq \rho + 1$ correct processors, which, by the resiliency condition for $\mathscr{P}_2$ must all output the input value m. Note that the simulated faulty processors may output no value. However, Receiver need only simulate $\mathscr{P}_2$ until $\rho + 1$ processors output the same value. Although not all of these processors need be correct, this is the correct value, since of the first $\rho + 1$ processors to output at least one is a correct processor, and all correct processors (eventually) output the same value. Moreover, since there are at least $\rho + 1$ nonfaulty processors, all of which will output the same value, the simulation terminates. □

Since VSS is stronger than USS, we have:

COROLLARY 6.1. *Any n processor solution to (σ, ρ)-VSS yields a connectivity n solution to weakened 1-way (σ, ρ)-SMT under containment.*

COROLLARY 6.2. *(σ, ρ)-Unverified Secret Sharing requires at least $\sigma + 2\rho + 1$ processors, with or without a broadcast channel.*

PROOF. The proof is immediate from Theorem 6.1 and the lower bound for weakened 1-way (σ, ρ)-SMT under containment obtained in Corollary 5.1. □

COROLLARY 6.3. *t-Verifiable Secret Sharing requires* $3t + 1$ *processors, even in the presence of a broadcast channel.*

The following observation states that the need for secrecy rules out the possibility of a deterministic solution to Secret Message Transmission, and this holds even if there is no resiliency requirement.

LEMMA 6.2. *For* $\sigma \geq 1$, *neither* 1*-way nor* 2*-way* $(\sigma, 0)$*-SMT can be solved with a deterministic protocol.*

PROOF. For the sake of contradiction let us suppose there exists a deterministic solution P to the (1- or 2-way) (σ, ρ)-SMT problem. For each $m \in Q$ and for each $L \in (n-1)^\sigma$, let $C(m, L)$ denote the conversation on the wires in L during the (unique) failure-free execution E_m in which the secret message is m.

Let m, m', where $m \neq m'$, be arbitrary elements of Q. By the secrecy condition, $\forall L \in (n-1)^\sigma$: $C(m, L) = C(m', L)$. But since this holds for all sets $L \in (n-1)^\sigma$ the conversations over all wires on inputs m and m' in E_m and $E_{m'}$ are identical. In particular, Receiver cannot distinguish the two executions. □

COROLLARY 6.4. (σ, ρ)*-USS has no deterministic solution.*

PROOF. The proof is immediate from Theorem 6.1 and Lemma 6.2. □

THEOREM 6.2. *Within the class of problems having no deterministic solution, the cost of an error-free solution can provably exceed the cost of a solution with arbitrarily small probability of error.*

PROOF. By Corollary 6.4 (σ, ρ)-USS has no deterministic solution. For any k, there exists a $\sigma + \rho + 1$ processor solution to (σ, ρ)-USS with probability at most 2^{-k} of error [20]. By Corollary 6.2, any error-free solution to (σ, ρ)-USS requires $\sigma + 2\rho + 1$ processors. This bound is tight [3, 18]. □

7. *Beyond Containment*

In this section, we study how the bounds obtained in Section 5 change when the containment assumption is removed, provided $\mathscr{A}_D$ is not oblivious. To obtain upper bounds in this case is simple: any algorithm for $(\sigma + \rho, \rho)$-SMT, under the containment assumption completely solves the general (σ, ρ)-SMT problem, even if the adversaries are actually allowed to communicate during execution of the protocol. This yields an increase of ρ wires in both the 1-way and 2-way case. We, therefore, have the following upper bounds:

THEOREM 7.1. *Connectivity* $\sigma + 3\rho + 1$ *is sufficient for* 1*-way* (σ, ρ)*-SMT, and connectivity* $\sigma + 2\rho + 1$ *is sufficient for* 2*-way* (σ, ρ)*-SMT, even without the containment assumption.*

Since the bounds of Theorem 7.1 are tight when the two adversaries can communicate during the execution, we henceforth restrict our attention to the model in which they cooperate but do not communicate except through the shared wires.

In the 1-way case, we can do slightly better, adding only $\rho - 1$ wires. We also show this bound is tight. The reason only $\rho - 1$ additional wires are needed in the 1-way case is that in this case $\mathscr{A}_D$ can only communicate with $\mathscr{A}_L$ if the sets

D and L intersect. This is because, in the 1-way case, there is no "behavior" of Receiver for $\mathcal{A}_L$ to detect: Receiver is completely passive, while Sender sends just once, before $\mathcal{A}_D$ has a chance to act. Since in order to drive up the lower-bound D and L must intersect, $\mathcal{A}_D$ can convey to $\mathcal{A}_L$ information about the communication on, at most, an additional $|D| - 1 \leq \rho - 1$ wires (those not in L), so we can apply the bounds for 1-way $(\sigma + \rho - 1, \rho)$-SMT under containment.

For the 2-way case, we have only proved that full additional ρ wires are needed for all 3-phase algorithms and for algorithms of any number of phases that use only the public channel after the first phase. For the general case, we have a lower bound of only $\sigma + 2\rho$, off by a single wire. For the next proof, we assume $\mathcal{A}_D$ can "disrupt" communication on a wire w by placing some particular text on w that $\mathcal{A}_L$, listening to w, can recognize as disruption.

THEOREM 7.2. *In the model in which $\mathcal{A}_L$ and $\mathcal{A}_D$ cooperate but do not communicate, if $\sigma \geq 1$, then (σ, ρ)-SMT requires $\sigma + 3\rho$ wires in the 1-way case and $\sigma + 2\rho$ wires in the 2-way case.*[7]

PROOF. The proofs for the two cases are similar.

For the sake of contradiction, let P be a protocol for the 1-way case requiring only $n = \sigma + 3\rho - 1$ wires. Let L contain exactly wires $\{1, 2, \ldots, \sigma\}$, and let $D = \{\sigma, \sigma + 1, \ldots, \sigma + \rho - 1\}$. Let T be all wires not in $L \cup D$. Recall that, by Theorem 5.1, the information on wires in $L \cup D$ completely determines the secret, since $|L \cup D| = \sigma + \rho - 1 = n - 2\rho$ wires.

Let $m' \in Q$ be a message of maximal probability, according to the underlying probability distribution Π on messages. Let m be any message different from m', and consider a particular disruptor-free execution E of P in which the message transmitted is m. Since this is a 1-way protocol, communication consists of a single phase. Let X_D denote the information sent during E over the wires in D.

Let us choose $\mathcal{A}_D$ as follows: If the information sent over D is not X_D, then $\mathcal{A}_D$ disrupts transmission on wire σ, otherwise it does nothing.

The corresponding strategy of $\mathcal{A}_L$ is: If communication on wire σ is *not* disrupted, then output the message determined by X_D together with whatever has been sent over the wires $\{1, 2, \ldots, \sigma - 1\}$ (the contents of wire σ are given by X_D). Otherwise, output m'.

To see that $\mathcal{A}_L$ has an advantage in guessing the secret message, note that it is always correct when the message is actually m' and it is always correct when communication on the wire is not disrupted, which occurs (by choice of E) in at least one transmission of $m \neq m'$. Thus, it does better than $\Pi(m')$, violating secrecy.

The proof for the 2-way case is similar in spirit, but is technically more involved. This time, let us assume P is a protocol for the 2-way case requiring only $\sigma + 2\rho - 1$ wires. Recall that by Lemma 5.2 there exists a bound B on the number of phases in any disruptor-free execution of P.

As above, let m' be a "likely" message and let $m \neq m'$ be arbitrary. Let L and D be chosen so that $|L| = \sigma$, $|D| = \rho$, and $|L \cap D| = 1$. Let E be a

[7] If $\sigma = 0$, then connectivity $2\rho + 1$ is necessary and sufficient for both the 1-way and 2-way cases.

disruptor-free execution in which m is transmitted. Let X_D (respectively, X_L) be the full conversation on the wires in D (respectively, L) during E.

By Lemma 5.2, in any disruptor-free execution the information sent over L and D together completely determines the secret. $\mathscr{A}_D$ disrupts on wire w in the first phase in which the conversation over D differs from X_D.

The corresponding strategy for $\mathscr{A}_L$ is: If communication on wire w has not been disrupted at the end of phase B and the conversation over the wires in L is X_L, then output m, else output m'. □

Consider algorithm SlowSMT and let $\sigma > \rho$. Let L contain exactly wires $1, \ldots, \sigma$, while D contains only wire $\sigma + 1$. Let the strategy of $\mathscr{A}_D$ be to alter the contents of wire $\sigma + 1$ if and only if $s_{\sigma+1} \neq 5$. If Receiver responds "OK," indicating that no disruption occurred, then $\mathscr{A}_L$ learns an additional share of the pad, so if $\tau = \sigma$, then $\mathscr{A}_L$ learns a total of $\sigma + 1$ shares, enough to reconstruct the pad. Thus, even if the sets D and L are disjoint, $\mathscr{A}_D$ can indirectly communicate critical information to $\mathscr{A}_L$. We now prove that the bound of $\sigma + 2\rho + 1$ wires is tight for 3-phase algorithms in which the adversaries do not communicate and $\mathscr{A}_D$ is not oblivious.

THEOREM 7.3. *Let P be any protocol for 2-way (σ, ρ)-SMT in the model in which $\mathscr{A}_L$ and $\mathscr{A}_D$ do not communicate and $\mathscr{A}_D$ is not oblivious. If every execution of P lasts exactly 3 phases, beginning with a transmission from Sender to Receiver, then P requires $n \geq \sigma + 2\rho + 1$ wires.*

PROOF. We assume, for the sake of contradiction, the existence of such a protocol P requiring only $\sigma + 2\rho$ wires. Let $M = \{m, m'\}$ and let $\Pi(m) = \Pi(m') = 1/2$ be the underlying probability distribution on M. The secrecy condition for (σ, ρ)-SMT implies that no listening adversary can have probability better than $1/2$ of guessing the secret message being transmitted. We obtain our contradiction by exhibiting a pair of strategies for $\mathscr{A}_D$ and $\mathscr{A}_L$ that will permit the listening adversary to guess the secret message with probability strictly greater than $1/2$, where the probability is taken over the random choices of Sender and Receiver.

As before, we first study P against an empty disrupting adversary. In general, we describe certain parts of an execution, by specifying the message m being transmitted, and the coin flip vectors of Sender and of Receiver: C_S and C_R, respectively. We name such an execution E by writing $E(m, C_S, C_R)$. We break the wires into three groups: $X = \{1, 2, \ldots, \sigma\}$, $Y = \{\sigma + 1, \ldots, \sigma + \rho\}$, and $Z = \{\sigma + \rho + 1, \ldots, \sigma + 2\rho\}$. Throughout the proof, the only wires compromised by $\mathscr{A}_L$ are those in X.

Let the conversations in a particular disruptor-free execution $E_0(m, C_S^0, C_R^0)$ be

$$\alpha_1 \beta_1 \gamma_1,$$
$$\alpha_2 \beta_2 \gamma_2,$$
$$\alpha_3 \beta_3 \gamma_3,$$

where α_i (respectively, β_i, γ_i) denotes what is sent over X (respectively, Y, Z) during phase i of E.

For any given $\mathscr{A}_D$, let $\hat{\Pi}(\alpha_1 \alpha_2 \alpha_3 \mid m, \mathscr{A}_D)$ denote the probability, taken over coin flips of Sender and Receiver, that $\mathscr{A}_L$ sees $\alpha_1 \alpha_2 \alpha_3$ given that the secret message is m and the disruptor is the given $\mathscr{A}_D$. In particular, let

$\hat{\Pi}(\alpha_1\alpha_2\alpha_3 \mid m, \varnothing)$ denote the probability, taken over coin flips of Sender and Receiver, that $\mathscr{A}_L$ sees $\alpha_1\alpha_2\alpha_3$ given that the secret message is m and the disruptor is empty. Analogous definitions can be made for the case that the message is m' and/or $\mathscr{A}_L$ does *not* see $\alpha_1\alpha_2\alpha_3$ (this last condition is denoted $\neg(\alpha_1\alpha_2\alpha_3)$).

LEMMA 7.1. $\forall \mathscr{A}_D\ \hat{\Pi}(\alpha_1\alpha_2\alpha_3 \mid m, \mathscr{A}_D) = \hat{\Pi}(\alpha_1\alpha_2\alpha_3 \mid m', \mathscr{A}_D)$. *That is, regardless of $\mathscr{A}_D$, $\mathscr{A}_L$ is equally likely to see this string when the secret is m as when the secret is m'. The probabilities are taken over the coin flips of Sender and Receiver.*

PROOF. The lemma is immediate from the definitions of secrecy. Indeed, for any $\mathscr{A}_D$ suppose, for the sake of contradiction, that $\hat{\Pi}(\alpha_1\alpha_2\alpha_3 \mid m, \mathscr{A}_D) = p + \epsilon$, while $\hat{\Pi}(\alpha_1\alpha_2\alpha_3 \mid m', \mathscr{A}_D) = p$ for some ϵ, $p > 0$. Then $\hat{\Pi}(\neg(\alpha_1\alpha_2\alpha_3) \mid m', \mathscr{A}_D) = (1 - p)$. Consider the following strategy for $\mathscr{A}_L$: If the conversation over X is $\alpha_1\alpha_2\alpha_3$, then output m, else output m'.

$$\begin{aligned}\Pr[\mathscr{A}_L \text{ correct} \mid \mathscr{A}_D] &= \Pi(m)\hat{\Pi}(\alpha_1\alpha_2\alpha_3 \mid m, \mathscr{A}_D) \\ &\quad + \Pi(m')\hat{\Pi}(\neg(\alpha_1\alpha_2\alpha_3) \mid m', \mathscr{A}_D) \\ &= \frac{1}{2}(p + \epsilon) + \frac{1}{2}(1 - p) \\ &> \frac{1}{2}.\end{aligned}$$

□

LEMMA 7.2. *There exists an execution $E_1(m', C_S^1, C_R^1)$ with conversations*

$$\begin{array}{llll} \alpha_1 & \delta_1 & (\eta_1 \to \gamma_1) \\ \alpha_2 & \delta_2 & (\eta_2 \to \gamma_2) \\ \alpha_3 & \delta_3 & \eta_3 \end{array}$$

for some $\delta_1, \delta_2, \delta_3, \eta_1, \eta_2, \eta_3$. The α_i's and γ_i's are the same here as in E_0.

PROOF. Let $\mathscr{A}_D$ compromise precisely the wires in Z, and let its strategy be to always in phase 1 transmit γ_1 and always in phase 2 transmit γ_2, regardless of what is originally placed on these wires. Suppose no execution such as E_1 exists. Then with this choice of $\mathscr{A}_D$, the only time $\mathscr{A}_L$ sees $\alpha_1\alpha_2\alpha_3$ is when the transmitted message is m. In other words, $\hat{\Pi}(\alpha_1\alpha_2\alpha_3 \mid m', \mathscr{A}_D) = 0$. In contrast, $\hat{\Pi}(\alpha_1\alpha_2\alpha_3 \mid m, \mathscr{A}_D) > 0$ (E_0 is a witness). This violates Lemma 7.1. □

COROLLARY 7.1. $\forall \xi_3$ *there exists an execution $E_2(m', C_S^2, C_R^2)$ with conversations*

$$\begin{array}{llll} \alpha_1 & \delta_1 & (\eta_1 \to \gamma_1), \\ \alpha_2 & \delta_2 & (\eta_2 \to \gamma_2), \\ \alpha_3 & \delta_3 & (\eta_3 \to \gamma_3), \end{array}$$

where the γ's are as in E_0 (and therefore E_1), and the η's and δ's are as in E_1.

PROOF. The proof is immediate from Lemma 7.2 and the fact that every execution always stops after three phases. □

Let $\mathscr{A}_D$ have the following strategy: Compromise the wires in Z. If in the first two phases, the communication observed is $\gamma_1\gamma_2$, then replace γ_2 with η_2; else do nothing.

The following lemma is used to prove first that $\eta_2 \neq \gamma_2$, so that this adversary is not effectively empty, and then to show that, for this specific choice of $\mathscr{A}_D$, the probability that $\mathscr{A}_L$ will see the sequence $\alpha_1\alpha_2\alpha_3$ given that the message is m is smaller than the same conditional probability with the empty disrupting adversary.

LEMMA 7.3. *Fix an arbitrary disruptor-free execution* $E_3(m, C_S^3, C_R^3)$ *of the form*

$$\begin{array}{ccc} \alpha_1 & p_1 & \gamma_1 \\ \alpha_2 & p_2 & \gamma_2 \\ x_3 & y_3 & z_3 \end{array}$$

for some p_1, p_2, x_3, y_3, z_3. *There can be no execution* $E_4(m, C_S^3, C_R^3)$, *of the form*

$$\begin{array}{ccc} \alpha_1 & p_1 & \gamma_1 \\ \alpha_2 & p_2 & (\gamma_2 \to \eta_2) \\ \alpha_3 & r_3 & s_3 \end{array}$$

for any r_3, s_3.

PROOF. By Corollary 7.1 there exists an execution $E_5(m', C_S^2, C_R^2)$ with conversations

$$\begin{array}{ccc} \alpha_1 & \delta_1 & (\eta_1 \to \gamma_1) \\ \alpha_2 & \delta_2 & (\eta_2 \to \gamma_2) \\ \alpha_3 & \delta_3 & (\eta_3 \to s_3). \end{array}$$

Assume for the sake of contradiction that E_4 exists as described. We construct the following erroneous execution $E_6(m, C_S^3, C_R^2)$:

$$\begin{array}{ccc} \alpha_1 & (p_1 \to \delta_1) & \gamma_1 \\ \alpha_2 & (\delta_2 \to p_2) & \eta_2 \\ \alpha_3 & (r_3 \to \delta_3) & s_3 \end{array}$$

Sender cannot distinguish E_6 from E_4, in which it is sending the message m, while Receiver cannot distinguish E_6 from E_5, in which it outputs m'. □

COROLLARY 7.2. $\eta_2 \neq \gamma_2$.

PROOF. Let $E_3 = E_0$. Let the first two phases of $E_4(m, C_S^0, C_R^0)$ be

$$\begin{array}{ccc} \alpha_1 & \beta_1 & \gamma_1 \\ \alpha_2 & \beta_2 & (\gamma_2 \to \eta_2). \end{array}$$

If $\eta_2 = \gamma_2$, then the first two phases of E_4 are not distinguishable to Sender (or Receiver) from the first two phases of E_3 (and hence, E_0), so Sender sends $\alpha_3\beta_3\gamma_3$ in phase 3 of E_4, violating the lemma. Thus, $\eta_2 \neq \gamma_2$. □

The corollary implies that the adversary $\mathscr{A}_D$ that changes γ_2 to η_2 if and only if the first two phases of communication on the wires in Z are $\gamma_1\gamma_2$ and otherwise does nothing is not an empty adversary, in the sense that it sometimes makes real changes to the messages carried on the wires in Z.

Lemma 7.3 says that when the message sent is m, then $\mathscr{A}_L$ is no more likely to see $\alpha_1\alpha_2\alpha_3$ with this particular choice of $\mathscr{A}_D$ than with an empty disrupting adversary. In fact, by taking $p_1 = \beta_1$ and $p_2 = \beta_2$ we obtain the following very important corollary.

COROLLARY 7.3. $\hat{\Pi}(\alpha_1\alpha_2\alpha_3 \mid m, \mathscr{A}_D) < \hat{\Pi}(\alpha_1\alpha_2\alpha_3 \mid m, \varnothing)$.

In other words, when the message is m, $\mathscr{A}_L$ is less likely to see $\alpha_1\alpha_2\alpha_3$ with the active disrupting adversary $\mathscr{A}_D$ than with the empty adversary. To see this, suppose the secret message is m. If $\mathscr{A}_L$ is going to see $\alpha_1\alpha_2\alpha_3$, then the execution must start

$$\begin{array}{l}\alpha_1 \ldots \\ \alpha_2 \ldots\end{array}$$

since $\mathscr{A}_D$ does not act before the second phase. If the last column is not $\gamma_1\gamma_2$, then there will be no difference between the probability that $\mathscr{A}_L$ sees $\alpha_1\alpha_2\alpha_3$ with $\mathscr{A}_D$ and the probability that it sees this string with $\varnothing$, since $\mathscr{A}_D$ does nothing in this case. However, if the last column is $\gamma_1\gamma_2$, then the lemma says there is no extension to this execution in which $\mathscr{A}_L$ sees $\alpha_1\alpha_2\alpha_3$ with $\mathscr{A}_D$, but there is such an extension with $\varnothing$ (witness E_0).

LEMMA 7.4. *For all p_1, p_2, y_3, z_3, there is no disruptor-free execution $E_7(m', C_S^7, C_R^7)$ with conversations*

$$\begin{array}{lll}\alpha_1 & p_1 & \gamma_1 \\ \alpha_2 & p_2 & \gamma_2 \\ \alpha_3 & y_3 & z_3.\end{array}$$

PROOF. Suppose, for the sake of contradiction, that such an execution E_7 exists. Then, since the algorithm must always terminate in 3 phases, there exists an execution $E_8(m', C_S^7, C_R^7)$

$$\begin{array}{lll}\alpha_1 & p_1 & \gamma_1 \\ \alpha_2 & p_2 & \gamma_2 \\ \alpha_3 & y_3 & (z_3 \to \gamma_3).\end{array}$$

We again construct an erroneous execution $E_9(m, C_S^0, C_R^7)$:

$$\begin{array}{lll}\alpha_1 & (\beta_1 \to p_1) & \gamma_1 \\ \alpha_2 & (p_2 \to \beta_2) & \gamma_2 \\ \alpha_3 & (\beta_3 \to y_3) & \gamma_3.\end{array}$$

E_9 is indistinguishable to Sender from E_0, in which Sender transmit m, but it is also indistinguishable to Receiver from E_8, in which Receiver outputs m'. □

Suppose the secret message is m'. If $\mathscr{A}_L$ is going to see $\alpha_1\alpha_2\alpha_3$ then the execution must begin with $\alpha_1\alpha_2$ in the first two phases. If in the first two phases $\mathscr{A}_D$ does not see $\gamma_1\gamma_2$, then the probability that the execution will continue with α_3 on the wires in X is the same with adversary $\mathscr{A}_D$ as with $\varnothing$, since $\mathscr{A}_D$ does nothing in this case. However, if the conversation on the wires in Z in the first two phases is $\gamma_1\gamma_2$ then the Lemma says there is no extension of the execution with $\varnothing$ in which $\mathscr{A}_L$ sees $\alpha_1\alpha_2\alpha_3$, while there may be such an

extension with $\mathscr{A}_D$. We therefore have the following:

COROLLARY 7.4. $\hat{\Pi}(\alpha_1\alpha_2\alpha_3 \mid m', \mathscr{A}_D) \geq \hat{\Pi}(\alpha_1\alpha_2\alpha_3 \mid m', \varnothing)$.

In summary, let $\mathscr{A}_D$ have the following strategy. Compromise the wires Z. If in the first two phases the communication observed is $\gamma_1\gamma_2$, then replace γ_2 with η_2; else do nothing. By Corollary 7.3,

$$\hat{\Pi}(\alpha_1\alpha_2\alpha_3 \mid m, \mathscr{A}_D) < \hat{\Pi}(\alpha_1\alpha_2\alpha_3 \mid m, \varnothing).$$

By Lemma 7.1,

$$\hat{\Pi}(\alpha_1\alpha_2\alpha_3 \mid m, \varnothing) = \hat{\Pi}(\alpha_1\alpha_2\alpha_3 \mid m', \varnothing).$$

By Corollary 7.4,

$$\hat{\Pi}(\alpha_1\alpha_2\alpha_3 \mid m', \varnothing) \leq \hat{\Pi}(\alpha_1\alpha_2\alpha_3 \mid m', \mathscr{A}_D).$$

Thus,

$$\hat{\Pi}(\alpha_1\alpha_2\alpha_3 \mid m, \mathscr{A}_D) < \hat{\Pi}(\alpha_1\alpha_2\alpha_3 \mid m', \mathscr{A}_D),$$

violating the secrecy condition for (σ, ρ)-SMT. In fact, let $\mathscr{A}_L$ have the following strategy: If the conversation on X is $\alpha_1\alpha_2\alpha_3$, then output m', else output m. Let $p = \hat{\Pi}(\alpha_1\alpha_2\alpha_3 \mid m, \varnothing)$. Then $\hat{\Pi}(\neg(\alpha_1\alpha_2\alpha_3) \mid m, \varnothing) = 1 - p$. By Corollary 7.3, for some $\epsilon > 0$, $\hat{\Pi}(\alpha_1\alpha_2\alpha_3 \mid m, \mathscr{A}_D) = p - \epsilon$. Thus, $\hat{\Pi}(\neg(\alpha_1\alpha_2\alpha_3 \mid m, \mathscr{A}_D) = 1 - p + \epsilon$. However, by Corollary 7.4, $\hat{\Pi}(\alpha_1\alpha_2\alpha_3 \mid m', \mathscr{A}_D) \geq p$. Thus,

$$\begin{aligned}\Pr[\mathscr{A}_L \text{ correct} \mid \mathscr{A}_D] &= \Pi(m)\hat{\Pi}(\neg(\alpha_1\alpha_2\alpha_3) \mid m, \mathscr{A}_D) \\ &\quad + \Pi(m')\hat{\Pi}(\alpha_1\alpha_2\alpha_3 \mid m', \mathscr{A}_D) \\ &\geq \frac{1}{2}(1 - p + \epsilon) + \frac{1}{2}p \\ &> \frac{1}{2}.\end{aligned}$$

8. *Applications*

We can extend these results to networks of processors, with little change. Given $\sigma \geq \rho$, we consider adversaries $\mathscr{A}_L$ and $\mathscr{A}_D$ that can compromise up to σ and ρ processors or edges, respectively. We require that for all sets L and D of processors compromised by $\mathscr{A}_L$ and $\mathscr{A}_D$, respectively, for any pair of processors $p, q \notin L \cup D$, p can send a secret message m to q so that $\mathscr{A}_L$ learns absolutely nothing about m and q receives m correctly. Clearly, even without containment, connectivity $\sigma + 2\rho + 1$ suffices.

Using our transmission scheme as a building block, we can immediately extend the results of Ben-Or, et al. [3] for secure computation on complete networks to general networks of sufficient connectivity. The increase in time is proportional to the diameter of the network, and there is no loss of correctness or secrecy.

A second application is to secure communication in networks of bounded degree. In [11], Dwork, et al. consider the problem of simulating a completely connected network by bounded-degree networks containing t faulty processors.

The crux of the simulation is a transmission scheme for simulating the point-to-point transmissions of the complete network by sending messages along several paths in the bounded degree network. This is done in such a way that, for all choices T of at most t faulty processors, there exists a large set M of nonfaulty processors capable of communicating among themselves as if they comprise a completely connected subnetwork, independent of the behavior of the faulty processors. For every such set T, we let $POOR(T)$ denote the set of correct processors that are not in M. It is shown in [11] how to simulate the transmission of a message between two processors on the butterfly network such that the set $POOR(T)$ is of size $O(t \log t)$. Additional results are obtained in [11] for random networks of bounded degree, as well as for networks of small but unbounded degree. However, the transmission scheme in [11] yields correct communication among all nonfaulty processors that are not in $POOR(T)$, it does not allow these processors to communicate *secretly*. Consequently, even in a network in which adjacent vertices are connected by private channels, the transmission scheme cannot be used to simulate a complete network with private channels.

Let $\mathscr{A}_D$ and $\mathscr{A}_L$ compromise at most ρ and σ processors, respectively. Letting D and L be defined as usual, we define $POOR(D, L)$ such that all nonfaulty processors that are *not* in $POOR(D, L)$ can communicate among themselves not only correctly but also such that their messages will be completely secret from $\mathscr{A}_L$. If $t = \rho = \theta(\sigma)$, then the maximal size of $POOR(D, L)$ may increase by a constant factor over the bound on $POOR(T)$ obtained in [11]. The intuition is simple. Essentially, in the scheme of [11], to send a message m from p to q (where both are neither faulty nor in $POOR$), p prepares an "encoding" of m consisting of some number k of replicas of m. Each replica is sent to q over a different path (the paths are not vertex disjoint.) The definition in [11] of $POOR(T)$ is that for any two processors not in $POOR(T)$ more than $k/2$ of the paths connecting these processors contain no element of T. Thus, more than half the paths used in the transmission of m contain no processor in T, so q can determine m by taking that value appearing on more than $k/2$ paths. Suppose $\rho = \sigma = t$. Then, for example, using the protocol for 1-way SMT without containment, we can construct an encoding of m of length $k = 4t + 1$ using BCH error correcting codes with secrecy parameter t. If fewer than $k/4$ of the paths from p to q contain a processor in $D \cup L$, then p can transmit a *secret* message m to q such that q receives m and $\mathscr{A}_L$ learns nothing about m. Plugging in this stronger requirement for correct transmission only affects the bounds on the maximal size of $D \cup L$ by a constant.

9. *Additional Remarks*

The concept of two distinct adversaries, $\mathscr{A}_L$ and $\mathscr{A}_D$, is an intriguing one. Generally, we have assumed in this paper that the adversaries cooperate with the goal of defeating the algorithm. However, it may be the case that the adversaries do not cooperate. Essentially, this is the situation when $\mathscr{A}_D$ simply disrupts at random. As we have seen, without the containment assumption the upper bounds are better in this case than when the adversaries cooperate. Are there other models and problems in which it makes sense to consider noncooperating adversaries?

Our study of the roles of the adversaries highlighted a small weakness in the two error-free VSS protocols known to us [3, 13]. Specifically, these protocols have the property that even if the dealer is good, the t faulty processors can force a scenario in which every *nonfaulty* processor knows the dealer's secret *with certainty*. It has recently been shown [10] that this weakness can be removed without increasing the number of processors from the lower bound of $3t + 1$. The construction relies heavily on a generalization of the technique used in the slow protocol sketched in Section 5 for removing faulty wires (or processors) from the system. It also uses the technique for parallelizing error detection used in protocol FastSMT.

In the generalization of the fault-detection technique, instead of identifying and removing the faulty processors (often impossible unless $n \geq (t + 1)^2$), the correct processors agree on a set of $2k$ processors, of which at least k are faulty. Even if we begin with only $3t + 1$ processors, removing all $2k$ processors from the system results in a new system of $3t + 1 - 2k = t + 1 + 2(t - k)$ processors, of which at most $t - k$ are faulty. Then, for example, using an extension of the methods of [3], any computation can be run in the remaining system using secrecy threshold t and resiliency threshold $(t - k)$. We believe the generalized fault detection technique and its parallelization are very powerful, and we expect them to have an impact in the design of fault tolerant algorithms in a Byzantine environment.

ACKNOWLEDGMENTS. Many people have helped us by listening to early arguments and making suggestions. In particular, we thank Hagit Attiya, Shafi Goldwasser, Silvio Micali, Ruediger Reischuk, and Eva Tardos. Most of all, we thank Larry Stockmeyer for many hours of invaluable discussion.

REFERENCES

1. BEAVER, D., AND GOLDWASSER, S. Multiparty computation with faulty majority. In *Proceedings of the 30th Symposium on Foundations on Computer Science*. IEEE, New York, 1989, pp. 468–473.
2. BEN-OR, M. Another advantage of free choice: Completely asynchronous agreement protocols. In *Proceedings of the 2nd Annual ACM Symposium on Principles of Distributed Computing* (Montreal, Que., Canada, Aug. 17–19). ACM, New York, 1983, pp. 27–30.
3. BEN-OR, M., GOLDWASSER, S., AND WIGDERSON, A. Completeness theorems for non-cryptographic fault-tolerant distributed computation. In *Proceedings of the 20th Annual ACM Symposium on Theory of Computing* (Chicago, Ill., May 2–4). ACM, New York, 1988, pp. 1–10.
4. CHAUM, D., CRÉPEAU, C., AND DAMGÅRD, I. Multiparty unconditionally secure protocols. In *Proceedings of the 20th Annual ACM Symposium on Theory of Computing* (Chicago, Ill., May 2–4). ACM, New York, 1988, pp. 11–19.
5. CHOR, B., GOLDWASSER, S., MICALI, S., AND AWERBUCH, B. Verifiable secret sharing and achieving simultaneity in the presence of faults. In *Proceedings of the 26th Symposium on Foundations of Computing*. IEEE, New York, 1985, pp. 383–395.
6. CHOR, B., AND KUSHILEVITZ, E. A zero–one law for Boolean privacy. In *Proceedings of the 21st Annual ACM Sympoisum on Theory of Computing* (Seattle, Wash., May 15–17). ACM, New York, 1989, pp. 62–72.
7. CLEVE, R. Limits on the security of coin flips when half the processors are faulty. In *Proceedings of the 18th Annual ACM Symposium on Theory of Computing*. (Berkeley, Calif., May 28–30). ACM, New York, 1986, pp. 364–369.
8. DOLEV, D. The Byzantine Generals strike again. *J. Algorithms 3* (1982), 14–30.
9. DOLEV, D., AND DWORK, C. On-the-fly generation of names and communication primitives. Extended abstract, November 1989.
10. DWORK, C. Strong verifiable secret sharing. In *Proceedings of the 4th International Workshop on Distributed Algorithms*. Lecture Notes in Computer Science, vol. 480. Springer-Verlag, New York, 1990, pp. 213–227.

11. DWORK, C., PELEG, D., PIPPENGER, N., AND UPFAL, E. Fault tolerance in networks of bounded degree. *SIAM J. Comput. 17*, 5 (1988), 975–988.
12. FELDMAN, P. Optimal algorithms for Byzantine agreement. Ph.D. dissertation, Department of mathematics, MIT, Cambridge, Mass., 1988.
13. FELDMAN, P., AND MICALI, S. Optimal algorithms for Byzantine agreement. *Proceedings of the 20th Annual ACM Symposium on Theory of Computing* (Chicago, Ill., May 2–4). ACM, New York, 1988, pp. 148–161.
14. FISCHER, M., LYNCH, N., AND MERRITT, M. Easy impossibility proofs for distributed consensus problems. *J. Distrib. Comput. 1* (1986), 26–39.
15. GALIL, Z., HABER, S., AND YUNG, M. Primitives for designing multiparty protocols from specifications. Manuscript, 1989.
16. GOLDREICH, O., MICALI, S., AND WIGDERSON, A. How to play ANY mental game. In *Proceedings of 19th Annual ACM Symposium on Theory of Computing* (New York, N.Y., May 25–27). ACM, New York, 1987, pp. 218–229.
17. KARLIN, A., AND YAO, A. Probabilistic lower bounds for Byzantine agreement. Manuscript.
18. MCELIECE, R., AND SARWATE, D. On sharing secrets and Reed–Solomon codes. *Commun. ACM 24*, 9 (Sept. 1981), 583–584.
19. RABIN, M., AND LEHMANN, D. On the advantages of free choice: A symmetric and fully distributed solution to the dining philosophers problem. In *Proceedings of the Symposium on Principles of Programming Languages*. 1981, pp. 133–138.
20. RABIN, T., AND BEN-OR, M. Verifiable secret sharing and multiparty protocols with honest majority. In *Proceedings of the 21st Annual ACM Symposium on Theory of Computing*. (Seattle, Wash., May 15–17). ACM, New York, 1989, pp. 73–85.
21. SHAMIR, A. How to share a secret. *Commun. ACM 22*, 6 (June 1979), 612–613.
22. YAO, A. How to generate and exchange secrets. In *Proceedings of the 29th Symposium on Foundations of Computer Science*. IEEE, New York, 1986, pp. 162–167.

RECEIVED MAY 1990; REVISED MAY 1991; ACCEPTED JULY 1991