Joint Encryption and Message-Efficient Secure Computation

Matthew Franklin*[1] and Stuart Haber[2]

[1] Columbia University, New York, NY 10027
[2] Bellcore, 445 South Street, Morristown, NJ 07960-6438

Abstract. This paper connects two areas of recent cryptographic research: secure distributed computation, and group-oriented cryptography. We construct a probabilistic public-key encryption scheme with the following properties:

- It is easy to encrypt using the public keys of any subset of parties, such that it is hard to decrypt without the cooperation of every party in the subset.
- It is easy for any private key holder to give a "witness" of its contribution to the decryption (e.g., for parallel decryption).
- It is "blindable": From an encrypted bit it is easy for anyone to compute a uniformly random encryption of the same bit.
- It is "xor-homomorphic": From two encrypted bits it is easy for anyone to compute an encryption of their xor.
- It is "compact": The size of an encryption does not depend on the number of participants.

Using this joint encryption scheme as a tool, we show how to reduce the message complexity of secure computation versus a passive adversary (gossiping faults).

1 Introduction

This paper connects two areas of recent cryptographic research: secure distributed computation, and group-oriented cryptography. The problem of securely evaluating an arbitrary boolean circuit under cryptographic assumptions has been much studied, beginning with the work of Yao [14] and Goldreich, Micali, and Wigderson [8]. The notion of group-oriented cryptography, in which the power of a secret key holder is distributed over a number of participants, was introduced by Desmedt [3].

Practical implementations of group-oriented public-key encryption were given by Desmedt and Frankel [4]. (See also the related notion of fair public-key encryption [12].) We extend their implementations to achieve additional useful properties. Our scheme, which we call "additive joint encryption," can then be used to reduce the message complexity of cryptographic multi-party circuit evaluation.

* Paritally supported by an AT&T Bell Laboratories Ph.D. Scholarship. Part of this work done during a summer internship at Bellcore

D.R. Stinson (Ed.): Advances in Cryptology - CRYPTO '93, LNCS 773, pp. 266-277, 1994.

An additive joint encryption scheme enables a group of parties to have individual public keys, such that a message can be encrypted using the keys of any subset of parties. It is easy for any party to "withdraw" from an encryption, and the cooperation of all (current) participants is needed to decrypt. It is easy for each participant to give a "witness" of its contribution to the decryption, so that full decryption can occur in parallel (i.e., anyone can decrypt after seeing all the witnesses). Encrypted messages can be blinded (i.e., replaced by a random encryption of the same message), and they are xor-homomorphic (i.e., from the encryption of two bits it is easy for anyone to compute an encryption of their exclusive-or).

We present an implementation of additive joint encryption with the critical property that the size of an encrypted bit is independent of the number of parties participating in the encryption. This "compact" implementation is a construction that combines El-Gamal public-key encryption and schemes based on quadratic residues and non-residues.

We demonstrate the use of our encryption scheme as a tool for designing efficient multi-party cryptographic protocols (i.e., communication via broadcast channels only). We show that, using additive joint encryption, a factor of n can be gained in the number of bits broadcast by n parties to securely compute a circuit of size C. Specifically, in the privacy setting (against a passive adversary), only $O(nC)$ encrypted bits of communication are needed, as opposed to $O(n^2C)$ encrypted bits using existing methods.

For specific functions, further gains are possible by exploiting particular connections between properties of additive joint encryption and properties of the functions themselves. We illustrate this for the problem of comparing bit-strings.

Definitions and models are given in Sections 5.2 and 5.3. In Section 5.4, we describe the construction of our new encryption scheme. In Sections 5.5 and 5.6, we demonstrate the application of our scheme to reducing the message complexity of secure multi-party computation. Conclusions and some open problems are given in Section 5.7.

2 Model

We assume that there are n parties, each of which is a probabilistic polynomial time Turing Machine (read-only input tape, write-only output tape, random tape, one or more work tapes). The parties communicate by means of a broadcast channel, which can be modeled as an additional tape (communication tape) for each machine that is write-only for its owner and read-only for everyone else. When a party writes a message to this tape, we may say that the message has been "broadcast" or "posted." A protocol begins with all n parties in their start states, and ends when all have reached their final states. The output of the protocol is the (common) value written on the output tapes of the processors.

We are concerned with the message complexity of a protocol. This is measured as the total number of bits written on the communication tapes during the execution of the protocol. Since our protocols are cryptographic, we will state

the message complexity in terms of the number of *encrypted bits* written on the communication tapes. For the protocols we consider, this is all or most of the communication that occurs, and it is also a convenient measure independent of advances in either encryption methods or cryptanalytic techniques.

We will say that a protocol is "private" if its execution reveals no useful information to any subset of (polynomial bounded) gossiping processors. More specifically, anything that is efficiently computable from the views of a subset S of participants is also computable from just the output of the protocol together with the inputs and private keys of S.

3 Additive Joint Encryption

In this section we give definitions for additive joint encryption, and then a naive implementation for which the size of an encrypted bit depends on the number of participating parties.

3.1 Definition

A *joint encryption scheme* for $[1 \cdots n]$ is a collection of encryption functions $\{E_S : S \subseteq [1 \cdots n]\}$ and a collection of partial decryption functions $\{D_i : i \in [1 \cdots n]\}$ such that $D_i(E_S(M)) = E_{S-\{i\}}(M)$ for all messages M and all $i \in S$, and such that it is easy to compute M from $E_\emptyset(M)$. We use the notation $D_S(c)$ to stand for the result of applying to c the decryption functions corresponding to each element of S.

A joint encryption scheme is *public-key* if each E_S is easy to compute, while computing each D_i requires a different trapdoor; in fact, our implementation has the stronger "upward conversion" property: $E_{S \cup S'}(M)$ is easy to compute from $E_S(M)$ for any subset S'. A joint public-key encryption scheme is probabilistic if each E_S is probabilistic; we write $E_S(M)$ to denote a uniformly random choice of possible encryptions of M. A probabilistic public-key joint encryption scheme is *secure* if each E_S is GM-secure [10] (computational indistinguishability of ciphertexts, even given the decryption functions $\{D_i : i \in S'\}$ for any $S' \subset S$).

A probabilistic joint encryption scheme is *blindable* if, given any subset $S \subseteq [1 \cdots n]$, and an encryption $C = E_S(M)$, it is possible to sample efficiently from the uniform distribution on the set $\{C' : D_S(C') = M\}$ of all possible encryptions of M (although it suffices for our purposes to be able to sample efficiently from any computationally indistinguishable distribution). We write $blind(C)$ to denote a random choice from this set.

A joint encryption scheme is *xor-homomorphic* if, given any messages M, M' in $\{0,1\}^k$, any subset $S \subseteq [1 \cdots n]$, and any encryptions $E_S(M), E_S(M')$, it is easy to compute $E_S(M \oplus M')$.

A joint encryption scheme is *witnessed* if there are functions $\{W_i : 1 \leq i \leq n\}$ such that each W_i is hard to compute without the trapdoor for D_i, and such that $D_i(E_S(M))$ can be easily computed from $E_S(M)$ and $W_i(E_S(M))$ for any M and any $i \in S$, but no other $D_i(E_S(M'))$ can be easily computed from $E_S(M')$ and

$W_i(E_S(M))$. If every participant in an encryption provides a witness in parallel, the decryption can be computed much faster than by the use of (inherently sequential) partial decryption functions.

Finally, we use the term *additive joint encryption scheme* to denote a secure, blindable, xor-homomorphic, witnessed probabilistic public-key joint encryption scheme.

Desmedt and Frankel [4] consider "threshold" encryption, which is essentially a public-key joint encryption scheme for which any sufficiently large subset of parties can decrypt a message. We do not include this property in our definition, because it is not needed for our main application, secure circuit evaluation. In Section 5.4, we explain how to add threshold decryption capability to our implementation using their techniques.

Notation: We may abuse the xor symbol by extending it to encryptions in the obvious way. When $\hat{x} = E_S(x), \hat{y} = E_S(y)$ are encryptions, we may write $\hat{x} \oplus \hat{y}$ to denote $E_S(x \oplus y)$.

3.2 Naive Implementation Based on Xor Shares

A naive implementation of additive joint encryption can be based on any blindable and xor-homomorphic probabilistic public-key encryption scheme. For example, each encryption function e_i could be an instance of the scheme due to Goldwasser and Micali [10], based on quadratic residues and nonresidues, using a distinct modulus N_i. Let d_i be the decryption function corresponding to e_i.

An additive joint encryption scheme can be constructed as follows. Encryption is given by $E_S(b) = (b', [e_j(b_j) : j \in S])$, where $b_j \in_R \{0,1\}$ for all $j \in S$, and where $b' = b + \sum_{j \in S} b_j \bmod 2$. Decryption is given by $D_i(\alpha, [\beta_j : j \in S]) = (\alpha \oplus d_i(\beta_i), [\beta_j : j \in S - \{i\}])$ whenever $i \in S$ (or just $d_i(\beta_i)$ can be given as a decryption witness).

Note that the size of an encryption in this scheme grows as the number of participants increases. We seek a compact scheme for which the size of an encryption is independent of the number of parties.

4 Compact Additive Joint Encryption

In this section, we show how additive joint encryption can be implemented compactly, i.e., such that the size of an encrypted bit does not depend on the number of parties participating in the encryption.

4.1 Intuition of El-Gamal Based Scheme

As shown by Desmedt and Frankel [4], it is possible to construct a joint encryption scheme from El-Gamal's method of public-key encryption [6]. Although blindable, this scheme is not xor-homomorphic, and thus not an additive joint

encryption scheme. However, we can convert this into an additive joint encryption scheme for which the size of an encryption is independent of the number of parties.

The basic idea is to use El-Gamal with a composite modulus (whose factorization is unknown), encrypting zeros and ones as El-Gamal encryptions of random quadratic residues and non-residues, respectively (all with Jacobi symbol +1). This almost works as is, since blinding and xor can be achieved by component-wise multiplication of the two parts of an El-Gamal encryption. Unfortunately, although these products preserve the correct quadratic character (residue or non-residue) of the encrypted values, the parties—who don't know the factorization of the modulus—will be unable to make use of them. The parties cannot compute the quadratic character of an El-Gamal decryption, unless the entire history of blindings is stored and revealed. It wouldn't help to give the factorization of the modulus to the parties, since that would allow any party to decrypt on its own.

This problem can be solved by accompanying each encrypted bit with an encryption of the witness that allows its decryption. When s^2 is used to encrypt a zero, or $-s^2$ is used to encrypt a one, then the encryption of s^2 or $-s^2$ is accompanied by an El-Gamal encryption of s. This accompanying information makes decrypted values identifiable as residues or non-residues without knowing the factors of N. We give details in the next section.

4.2 Details of El-Gamal based Scheme

The public key is $[N, g^{x_1} \bmod N, \cdots, g^{x_m} \bmod N]$, where $g \in Z_N^*$, $N = pq$, $p \equiv q \equiv 3 \bmod 4$, although the prime factors p, q are unknown. The trapdoor information for D_i is x_i. Encryption of a zero is given by

$$E_S(0) = [g^r \bmod N, g^{r'} \bmod N, s^2(\prod_{j \in S} g^{x_j})^r \bmod N, s(\prod_{j \in S} g^{x_j})^{r'} \bmod N]$$

for $r, r', s \in_R Z_N^*$. Encryption of a one is given by

$$E_S(1) = [g^r \bmod N, g^{r'} \bmod N, -s^2(\prod_{j \in S} g^{x_j})^r \bmod N, s(\prod_{j \in S} g^{x_j})^{r'} \bmod N]$$

for $r, r', s \in_R Z_N^*$. Decryption is given by $D_i([\alpha, \beta, \gamma, \delta]) = [\alpha, \beta, \gamma\alpha^{-x_i} \bmod N, \delta\beta^{-x_i} \bmod N]$. Note that the fourth component of $E_\emptyset(b)$ enables the quadratic character of the third component (and hence the value of b) to be computed easily.

If $E_S(b) = [\alpha, \beta, \gamma, \delta]$ and $E_S(b') = [\alpha', \beta', \gamma', \delta']$, then

$$E_S(b \oplus b') = [\alpha\alpha' \bmod N, \beta\beta' \bmod N, \gamma\gamma' \bmod N, \delta\delta' \bmod N];$$

thus this scheme is xor-homomorphic. If $[\alpha, \beta, \gamma, \delta]$ is a joint encryption using E_S, and $r, r' \in_R Z_N^*$, then $[\alpha g^r \bmod N, \beta g^{r'} \bmod N, \gamma(\prod_{j \in S} g^{x_j})^r \bmod N, \delta(\prod_{j \in S} g^{x_j})^{r'} \bmod N]$ is a (nearly) uniformly random joint encryption of the

same value (from a computationally indistinguishable distribution when g has large order); thus this scheme is blindable. If $E_S(b) = [\alpha, \beta, \gamma, \delta]$, then $D_i(E_S(b))$ can be easily computed from $[\alpha^{-x_i} \bmod N, \beta^{-x_i} \bmod N]$ for any $i \in S$; thus this scheme is witnessed. The size of each encrypted bit is four elements of Z_N^*, independent of the number of participants; thus this scheme is compact.

We note that our implementation can be modified to include the property of threshold decryption, i.e., encryption such that any sufficiently large subset of parties can decrypt. This can be done, for example, by incorporating the modified shadow generation scheme based on Lagrange interpolation developed by Desmedt and Frankel [4] (which is possible when g and N are chosen as described in the next section).

4.3 Security of El-Gamal Based Scheme

Theorem 1. *If El-Gamal encryption with a composite modulus is GM-secure, then our compact encryption scheme is GM-secure.*

Proof. Suppose, for purposes of establishing a contradiction, that El-Gamal encryption with a composite modulus is GM-secure while our compact additive joint encryption scheme is not GM-ssecure. Then it would be easy to distinguish between composite El-Gamal encryptions of $+1$ and -1, since these can be easily converted into random additive joint encryptions of one and zero, as follows.

Let $(g^r \bmod N, (-1)^b g^{rx} \bmod N)$ be a composite El-Gamal encryption of $(-1)^b$ (using El-Gamal public key $g^x \bmod N$). Then $[g^r \bmod N, g^{r'} \bmod N$, $s^2(-1)^b g^{rx} \bmod N, s g^{r'x} \bmod N]$ is an additive joint encryption $E_S(b)$ for random $r', s \in_R Z_N^*$ (e.g., using joint public key

$$[N, r_1, \cdots, r_{|S|-1}, (g^x \prod_{i<|S|} r_i^{-1} \bmod N)]$$

for random $r_1, \cdots, r_{|S|-1}$).

However, our encryption scheme (and composite El-Gamal) is not GM-secure if composite quadratic character (residue vs. non-residue) is easy to compute. An attacker sees $g^x \bmod N, \alpha = g^r \bmod N, \beta = g^{r'} \bmod N, \gamma = (-1)^b s^2 g^{rx} \bmod N, \delta = s g^{r'x} \bmod N$, where b is the value of the encrypted bit (and where $x = \sum_{i \in S} x_i$). Let $QR_N(v) = 0$ if v is a quadratic residue modulo N and 1 otherwise. If $QR_N(\cdot)$ is easy to compute, then the attacker can determine $b = (QR_N(\alpha) * QR_N(g^x \bmod N)) \oplus QR_N(\gamma)$. We do not know whether the additional information available to the attacker makes the GM-security of our scheme (and composite El-Gamal) strictly weaker than the difficulty of computing quadratic character modulo N.

The security of the original El-Gamal public-key encryption scheme reduces to the difficulty of breaking an instance of the Diffie-Hellman key exchange scheme [5] (i.e., a problem that is no more difficult than but not known to be equivalent to the discrete log problem). McCurley [11] showed how El-Gamal encryption with a composite modulus (and a careful choice of g and N) can be

secure against an adversary who could break the Diffie-Hellman key exchange, or could factor the modulus, but not both. However this was a proof of security in the sense that no polynomial time algorithm can invert a non-negligible fraction of ciphertexts, and not GM-security (computational indistinguishability of ciphertexts).

We note that if g and N are chosen as suggested by McCurley, then the technical condition for incorporating threshold decryption into our scheme can be met. Specifically, McCurley's proof is based on the choices $g = 16$, $N = pq$, $p = 8r + 3$, $q = 8s - 1$ (where r, s have special structure), and this meets the condition of Desmedt and Frankel [4] for their modified shadow generation scheme based on Lagrange interpolation (i.e., that g have odd order in Z_N^*).

5 Message-Efficient General Secure Computation

Several solutions have been found to the problem of securely evaluating an arbitrary boolean circuit under cryptographic assumptions, beginning with the work of Yao [14] and Goldreich, Micali, and Wigderson [8]. We focus on the message complexity (i.e., number of encrypted bits of communication) of such protocols in the privacy setting.

5.1 Previous Approaches to Private Computation

Previously, the lowest message complexity known for n parties to privately evaluate a circuit of size C under reasonable cryptographic assumptions was $O(n^2C)$ encrypted bits of communication. This same complexity was achievable using either of the main techniques for secure circuit evaluation in the cryptographic setting: the "gate-by-gate" approach or the "circuit-scrambling" approach.

In the gate-by-gate approach, each gate of the circuit is computed by having each pair of the n parties perform a private two-party protocol. In the protocol of Galil, Haber, and Yung [7], with efficiency improvements by Goldreich and Vainish [9], each two-party protocol is a single instance of "One out of Two Oblivious Transfer" (1-2-OT). It is possible to implement two-party 1-2-OT privately using a constant number of encrypted bits under a cryptographic assumption (e.g., three encrypted bits suffice under the assumption that composite quadratic character is hard). This gives a total message complexity of $O(n^2C)$ encrypted bits.

In the circuit-scrambling approach, each party takes a turn modifying the truth tables of the gates of the circuit. In the protocol of Chaum, Damgård, and van de Graaf [2], each party can randomly permute the rows, and can randomly complement certain of the rows and columns of each truth table. Records of each party's modifications are preserved in the form of bit commitments, which accompany the scrambled circuit as it passes from party to party (to enable circuit evaluation after the nth party has finished scrambling). Each party contributes a constant number of bit commitments for each gate (e.g., one bit commitment for each truth table row), and so the scrambled circuit as it passes from party i

to party $i+1$ includes $O(iC)$ bit commitments. When each bit commitment is a single encryption, this gives a total message complexity of $O(n^2C)$ encrypted bits.

5.2 Reduced "Gate-by-Gate" Message Complexity

A gain of $O(n)$ in the message complexity of secure computation can be achieved via compact additive join encryption using either a gate-by-gate approach or a circuit-scrambling approach. We describe the gate-by-gate approach in detail in this section.

Theorem 2. *Under the assumption that compact additive joint encryption is possible, any boolean circuit with C gates can be privately evaluated by n parties using $O(nC)$ encrypted bits of communication.*

Proof. No communication is required for each NOT gate. Each AND gate requires two rounds of communication, and message complexity $4n$ encrypted bits (actually, three encryptions and two decryption witnesses per party, where each witness is half the length of an encryption for the El-Gamal based scheme).

The protocol begins with encryptions of the input bits on the shared tape. We show how the encrypted output of any gate can be computed in a constant number of rounds from its encrypted inputs. For a NOT gate, the output can be found without any communication by XORing the encrypted input with a default encryption of a one.

For an AND gate, suppose the encrypted gate inputs are $E_{[1\ldots n]}(x) = \hat{x}$ and $E_{[1\ldots n]}(y) = \hat{y}$. Each party i chooses $b_i, c_i \in_R \{0,1\}$, and broadcasts $\hat{b}_i = E_{[1\ldots n]}(b_i)$ and $\hat{c}_i = E_{[1\ldots n]}(c_i)$. With no communication, the parties can then find $\hat{x}' = E_{[1\ldots n]}(x \oplus b_1 \oplus \cdots \oplus b_n)$ and $\hat{y}' = E_{[1\ldots n]}(y \oplus c_1 \oplus \cdots \oplus c_n)$. The parties broadcast decryption witnesses for $\hat{x}', \hat{y}'$ to find $x' = x + \sum_{1\le j\le n} b_j \bmod 2$ and $y' = y + \sum_{1\le j\le n} c_j \bmod 2$. Let $z' = x' \wedge y'$.

For every $1 \le i, j \le n$, party i can find $E_{[1\ldots n]}(b_i \wedge c_j)$ by either encrypting a zero (if $b_i = 0$) or by blinding $\hat{c}_j$ (if $b_i = 1$). Similarly, each party i can find $E_{[1\ldots n]}(b_i \wedge y)$ and $E_{[1\ldots n]}(x \wedge c_i)$. Each party broadcasts a blinded encryption of the XOR of all of these encrypted values (in parallel with the previous broadcast). Now an encryption of the XOR of all received encrypted XOR's, together with an encryption of z', is equal to the encryption of $z = x \wedge y$ (i.e., by the distributivity of AND over XOR: $u \wedge (v \oplus w) = (u \wedge v) \oplus (u \wedge w)$).

When the last gate in the circuit has been computed, all parties know a joint encryption of the circuit output. At this point, the parties broadcast decryption witnesses to enable all of them to compute the actual circuit output.

For privacy, we need to argue that no subset S of parties learns anything about the other parties' inputs beyond what is implied by a knowledge of the inputs of S and the circuit output. It suffices to show that the distribution of transcripts of protocol executions, as viewed by S, can be simulated by a polynomial time machine that has access to the inputs and decryption functions of S, such that the simulated distribution and the actual distribution are computationally

indistinguishable. Excluding the decryption witnesses for the circuit output, the transcript of an execution of the protocol gives a number of joint encryptions for related values, and a number of decryption witnesses for uniformly random values. The simulator computes joint encryptions of uniformly random values to substitute for all of the joint encryptions in the transcript except the last one, and substitutes a uniformly random joint encryption of the output for the joint encryption of the output of the last gate, and substitutes decryption witnesses for uniformly random values for all of the decryption witnesses. By standard cryptographic arguments, the computational indistinguishability of these distributions follows from the computational indistinguishability of individual joint encryptions.

5.3 Reduced "Circuit-Scrambling" Message Complexity

To get the same gain in message complexity with a circuit-scrambling approach, the bit commitments are additive joint encryptions, but only a single commitment accompanies each truth table row as it passes from party to party. The single commitment at a row represents the xor of modifications performed by all parties. When a compact scheme is used, the size of the scrambled circuit doesn't increase from scramble to scramble.

Although the order of magnitude of the message complexity is the same, note that the multiplicative constant is better for our methods using the gate-by-gate approach. In the gate-by-gate approach, four encrypted bits are needed per AND gate (counting one decryption witness as half an encryption), and no communication is needed per NOT gate. In the circuit-scrambling approach, the same four encrypted bits are needed per AND gate, while two encrypted bits are needed per NOT gate (i.e., one bit commitment for each truth table row).

5.4 Measures of Message Complexity

In this section, we have shown that the message complexity of secure computation can be decreased by a linear factor in the number of participants. This gain has been computed under a "broadcast" measure of message complexity. Specifically, if one party posts a bit to the publicly readable bulletin board, then the protocol is charged one bit. The same charge applies no matter how many of the other parties ever read that posted bit.

It is reasonable to consider an alternative "readership" measure of message complexity, in which the number of readers of a message is relevant. By this measure, the protocol is charged k bits if a single posted bit is read by k of the other parties.

The linear gain in message complexity is maintained with respect to the readership measure for the "circuit-scrambling" protocol described in Section 5.3. This is because most broadcasts are only used to pass the scrambled circuit from one party to the next, i.e., to be read by only one other party. However, the "gate-by-gate" protocol loses the linear gain with respect to the readership

measure. Posting messages that are read by all other parties seems to be an essential feature of this approach.

6 Customized Secure Protocol for Bit-String Comparison

In this section, we show further application of our encryption method by describing a novel protocol for n parties to privately compare two encrypted bit-strings. The message complexity of this protocol has the same order of magnitude as would be achieved by computing a comparison circuit using our general techniques, although a small constant factor (roughly three) is saved. We believe that this protocol is of interest because it demonstrates that for practical applications (where constant factors count) useful gains in communication complexity can come from customizing cryptographic tools to the specific secure computational task at hand. Note that a constant factor is also gained by this protocol with respect to the readership measure of message complexity.

Before giving our comparison protocol, we need to develop a tool to randomly permute pairs of encrypted inputs with low message complexity.

6.1 Shuffle Gate Computation

A "shuffle gate" has two main inputs x, y, a control input c, and two outputs α, β. When $c = 0$, the inputs pass through the gate unchanged: $\alpha = x$ and $\beta = y$. When $c = 1$, the inputs are flipped as they pass through the gate: $\alpha = y$ and $\beta = x$. A shuffle gate can be represented as a circuit with six AND and OR gates. Using our gate-by-gate approach, $24n$ encrypted bits of communication are needed to privately evaluate a shuffle gate.

By contrast, a uniformly random shuffle gate can be privately computed directly at a cost of only $2n$ encrypted bits of communication using the El Gamal based scheme. Let $\hat{x}, \hat{y}$ be the encryptions of the main inputs. Each party i chooses a uniformly random $c_i \in_R \{0, 1\}$. We want x, y to be flipped only if the xor of all of the c_i values is one. Each party i posts two encrypted values as follows: two encrypted zeros if $c_i = 0$, and two encryptions of $x \oplus y$ if $c_i = 1$. By xoring all of these posted pairs to the input pair $\hat{x}, \hat{y}$, each party gets an encryption of the appropriate output pair.

6.2 Details of the Comparison Protocol

Intuitively, the comparison protocol works on l-bit strings in $l - 1$ rounds, where each round reduces the length of the encrypted bit strings by one. The reduction is done in such a way that the result of comparing the two decrypted bit-strings is preserved after each round. Specifically, the parties remove the leading bits if these bits are equal, and remove the next-to-leading bits if the leading bits are unequal. Of course, it would violate privacy if any proper subset of parties could determine which of these two actions occurred in any round.

Our protocol guarantees that the correct action occurs obliviously (i.e., so that no proper subset of the parties can detect which action occurred) by repeatedly using the shuffle gate construction described in the preceding section. In each round, two shuffle gates are controlled by the same control bit γ. The decision about which pair of bits to discard each round depends critically on the value of the control bit for that round. The details are given in Figure 1.

Initially, $\tilde{x} = E_{[1\cdots n]}(x)$, $\tilde{y} = E_{[1\cdots n]}(y)$.

1. In Round i, $1 \le i \le l-1$, the following messages are sent:
   (a) Each party j $(1 \le j \le n)$ posts $[\alpha_{1j}, \beta_{1j}, \alpha_{2j}, \beta_{2j}; \gamma_j] =$
      i. $[blind(\tilde{x}.i \oplus \tilde{x}.(i+1)), blind(\tilde{y}.i \oplus \tilde{y}.(i+1)), blind(\tilde{x}.i \oplus \tilde{x}.(i+1)),$ $blind(\tilde{y}.i \oplus \tilde{y}.(i+1)); E_{[1\cdots n]}(1)]$ with prob $\frac{1}{2}$.
      ii. $[E_{[1\cdots n]}(0), E_{[1\cdots n]}(0), E_{[1\cdots n]}(0), E_{[1\cdots n]}(0); E_{[1\cdots n]}(0)]$ with prob $\frac{1}{2}$.
   (b) Each party j $(1 \le j \le n)$ posts a witness for $D_j(\tilde{x}.i \oplus \tilde{y}.i \oplus \gamma)$, where $\gamma = \gamma_1 \oplus \cdots \oplus \gamma_n$.
   (c) Before the start of Round $i+1$, each party (without communication) locally replaces $\tilde{x}.(i+1), \tilde{y}.(i+1)$ with
      i. $blind(\alpha_{21} \oplus \cdots \oplus \alpha_{2n} \oplus \tilde{x}.(i+1))$, $blind(\beta_{21} \oplus \cdots \oplus \beta_{2n} \oplus \tilde{y}.(i+1))$ if $D_{[1\cdots n]}(\tilde{x} \oplus \tilde{y} \oplus \gamma) = 0$.
      ii. $blind(\alpha_{11} \oplus \cdots \oplus \alpha_{1n} \oplus \tilde{x}.i)$, $blind(\beta_{11} \oplus \cdots \oplus \beta_{1n} \oplus \tilde{y}.i)$ if $D_{[1\cdots n]}(\tilde{x} \oplus \tilde{y} \oplus \gamma) = 1$.
2. In Round l, the following messages are sent:
   (a) Each party j $(1 \le j \le n)$ posts a witness for $D_j(\tilde{x}.l)$.
   (b) Without communication, each party computes the final answer to be $v = D_{[1\cdots n]}(\tilde{x}.l)$. (If $v = 1$ then $x \ge y$ else $x \le y$.)

Fig. 1. Message-Efficient Multi-party Comparison Scheme

All messages for this protocol, except for the last round, are witnesses for joint encryptions of uniformly random values, or joint encryptions of related values. Privacy follows from an argument similar to that of Theorem 5.2.

Theorem 3. *Under the assumption that compact additive joint encryption is possible, private multi-party comparison of two l-bit strings is possible with communication $4(l-1)n$ encrypted bits and ln decryption witnesses.*

7 Summary and Open Problems

The message complexity of secure distributed computation can be reduced by extending techniques from group-oriented cryptography. We show how gains

in multi-party evaluation of general circuits can be achieved by augmenting a joint encryption scheme to support blinding, witnessing, and adding ciphertexts *without* increasing the length of the ciphertexts.

We would like to find compact implementations of additive joint encryption based on other, possibly weaker, intractability assumptions, and to find other applications for such encryption schemes. In addition, we would like to explore other ways to improve the secure evaluation of specific useful functions by exploiting special properties of customized encryption methods. It would also be interesting to reduce the computational resources required for secure computation in other settings, possibly tolerating stronger adversaries.

References

1. D. Beaver, "Secure multiparty protocols and zero-knowledge proof systems tolerating a faulty minority," J. Cryptology (1991) 4: 75-122.
2. D. Chaum, I. Damgård, and J. van de Graaf, "Multiparty computations ensuring privacy of each party's input and correctness of the result," Crypto 1987, 87-119.
3. Y. Desmedt, "Society and group oriented cryptography: A new concept," Crypto 1987, 120-127.
4. Y. Desmedt and Y. Frankel, "Threshold cryptosystems," Crypto 1989, 307-315.
5. W. Diffie and M. Hellman, "New directions in cryptography," IEEE Transactions on Information Theory, 22(6):644-654, 1976.
6. T. El-Gamal, "A public key cryptosystem and a signature scheme based on discrete logarithms," IEEE Transactions on Information Theory, 31:469-472, 1985.
7. Z. Galil, S. Haber, and M. Yung, "Cryptographic computation: secure fault-tolerant protocols and the public-key model," Crypto 1987, 135-155.
8. O. Goldreich, S. Micali, and A. Wigderson, "How to play any mental game," STOC 1987, 218-229.
9. O. Goldreich and R. Vainish, "How to solve any protocol problem – an efficiency improvement," Crypto 1987, 73-86.
10. S. Goldwasser and S. Micali, "Probabilistic encryption," JCSS, 28(2):270:299, 1984.
11. K. McCurley, "A key distribution system equivalent to factoring," J. Crypt., 1(2):95-105, 1988.
12. S. Micali, "Fair public-key cryptosystems," Crypto 1992, 3.11-3.24 (preproceedings abstracts).
13. S. Micali and P. Rogaway, "Secure Computation," Crypto 1991, 392-404.
14. A. Yao, "How to generate and exchange secrets," FOCS 1986, 162-167.