Secure Supply-Chain Collaboration

Mikhail J. Atallah • Vinayak Deshpande • Keith B. Frikken • Leroy B. Schwarz

CERIAS and Department of Computer Sciences, Purdue University, 656 Oval Drive, West Lafayette, Indiana 47907-2086, USA

Krannert School of Management, Purdue University, 100 S. Grant St., West Lafayette, Indiana 47907-2076, USA

CERIAS and Department of Computer Sciences, Purdue University, 250 N. University St., West Lafayette, Indiana 47907-2066, USA

Krannert School of Management, Purdue University, 403 West State Street, West Lafayette, Indiana 47907-2056, USA

mja@cs.purdue.edu • deshpandev@mgmt.purdue.edu • kbf@cs.purdue.edu • lschwarz@exchange.purdue.edu

Revised July 2005, September 2004, February 2004, Original July 2003

Supply-chain interactions have huge economic importance, yet these interactions are often managed inefficiently. One of the major sources of inefficiency is information asymmetry; i.e., information that is available to one or more organizations in the chain (e.g., manufacturer, retailer) is not available to others. There are several causes of information asymmetry, among them fear that a supplier-chain partner will take advantage of private information, that information will leak to a competitor, etc. We propose Secure Supply-Chain Collaboration (SSCC) protocols that enable supply-chain partners to cooperatively achieve desired system-wide goals without revealing the private information of any of the parties, even though the jointly-computed decisions require the information of all the parties. Secure supply-chain collaboration has the potential to improve supply-chain management practice, and, by removing one major inefficiency therein, improve productivity. We present specific SSCC protocols for two types of supply-chain interactions: Capacity allocation, and market-clearing transactions under discriminatory and non-discriminatory pricing. Hence this work bridges three distinct research areas: Secure Multi-party Computation (SMC), Mechanism Design (MD) and Supply-Chain Management (SCM). We also provide a framework for further supply-chain collaboration research that incorporates both preservation of the participants' private information and incentives to collaborate.

(*Supply-chain online interactions; privacy; security; secure multi-party computation; capacity allocation in e-commerce; market-clearing transactions*)

---

1 Introduction

Information asymmetry is known to create inefficiencies in managing supply chains, among them under-investment in capacity, which leads to shortages; misallocation of inventory, transportation and management resources; increased prices; and reduced customer service. It can also lead to increased use of premium shipping, penalties resulting from line shut-downs, and lost future business. However, the barriers to information-sharing are significant, among them, fear that information voluntarily shared with a partner will be used against the volunteer, fear that sensitive information will leak to a competitor, government regulations about information-sharing, etc. Further, if one of the parties is, say, the U.S. government, then there are national-security reasons to protect secret information.

The goal of this work is the design of protocols that enable the parties in supply chains to cooperatively achieve desired results without revealing their own private data, even though the jointly-computed answers depend on every participant's private data. The contributions of this paper are (i) to present such protocols for two classes of supply-chain interactions (capacity allocation, and market-making under either discriminatory or nondiscriminatory pricing), and (ii) to provide techniques that are likely to be useful in the design of future SSCC protocols. Although the specific protocols described are highly stylized, they illustrate the nature of SSCC protocols for more complicated scenarios. Further, one of our assumptions – linear demand curves – because of its two-parameter specification, provides a more severe test of privacy preservation than non-linear demand curves.

In what follows, we will first review the related literature in computer science, supply-chain management, and economics. Section 3 introduces our security model, and has a more in depth discussion on results in SMC. Section 4 then introduces a well-known capacity-allocation model, describes the information required for secure collaboration, and then describes the corresponding protocol. Section 5 does the same for two market-clearing scenarios. Section 6 provides a brief summary and suggestions for additional research.

2 Literature Review

This paper draws on work from three broad areas: (1) cryptographic techniques for Secure Multi-Party Computation (SMC), which is a form of cooperative distributed computing that preserves the privacy of the participants' data; (2) mechanism design (MD) from Economics,

which deals with determining incentives to participants to truthfully reveal their private information; and (3) Supply-Chain Management (SCM). Typically, SMC assumes that participants have the right incentives to participate, while mechanism design typically assumes that participants would be willing to truthfully reveal their private information if provided the right incentives. Seldom is either of these assumptions valid in supply-chain transactions. Finally, the traditional Supply Chain Management literature assumes that information and decision-making are centralized; and, hence, neither privacy-preservation nor incentives is an issue.

Hence, a key contribution of this paper is the analysis of specific supply-chain interactions which preserve the privacy of the participants data and provide incentives to the participants to collaborate truthfully. We summarize the relevant literature from all the three fields below.

2.1 Cryptographic Techniques

Cryptographic techniques have revolutionized the way distinct entities interact via a computer network. Indeed, they are already an integral part of the daily computing experience of a typical (often oblivious) user. The intricacies of this complex and powerful technology are hidden from such a user, who uses it without even realizing what is going on "under the hood" (e.g., the use of SSL by Web browsers to protect credit-card and other private information). Cryptographic techniques can also, through the use of digital certificates and signatures, help ascertain that the other party is who they claim to be (authentication). We next review the sub-areas of cryptography relevant to this work.

2.1.1 Secure Multi-party Computation

The sub-area of cryptography closest to this work is secure multiparty computation. Secure multi-party protocols are a form of cooperative distributed computing that preserves the privacy of the participants' data. This general class of computations typically takes the following form between two parties (usually called Alice and Bob): "Alice" and "Bob" each have private data (say, x_A for Alice and x_B for Bob), and they want to conduct a computation based on their joint data; that is, to compute $f(x_A, x_B)$ where the function f is known to both Alice and Bob, and $f(x_A, x_B)$ is efficiently computable by someone who had both x_A and x_B. However, neither Alice nor Bob is willing to disclose his/her private data to the other, or even to a third party. A protocol that involves only Alice and Bob, is said to be secure if, at its end, Alice and Bob have learned only $f(x_A, x_B)$. Of course, Alice might infer

something about x_B from her knowledge of x_A, f, and $f(x_A, x_B)$, but that is unavoidable. Such inferences are related to the problem of inverse optimization (Ahuja and Orlin 2001), which will be discussed below.

The history of the multi-party computation problem is extensive since it was introduced by Yao (Yao 1982) and extended by Goldreich, Micali, Wigderson (Goldreich et al. 1987) and others. Goldreich states in (Goldreich 1998) that although the general secure multi-party computation problem is solvable in theory, using the solutions derived by these general results for special cases can be impractical. In other words, efficiency dictates the development of special solutions for special cases.

In many cases, supply-chain transactions require an extension of the typical SMC scenario. For example, in some supply-chain management scenarios, one can no longer necessarily assume that all parties are computing a function f that is known to all participants. Instead, Alice is computing some $f_A(x_A, x_B)$ and Bob is computing some $f_B(x_A, x_B)$, where Bob should not learn either x_A or f_A and Alice should not learn either x_B or f_B. That is, not only is Alice's data x_A proprietary, but so is the function f_A that she seeks to compute, and similarly for Bob and his x_B and f_B. In fact, in some SSCC problems, what Alice is really computing depends on the private functions $f_B, f_C, f_D, \ldots$etc of the other participants as well as on their private data $x_B, x_C, x_D, \ldots$etc. To see how this can happen, consider the case of a multi-party supply-chain negotiation where any party (say, Bob) can "drop out" of the negotiation depending on the value of $f_B(x_A, x_B, x_C, \ldots)$. The theoretical general secure multiparty computation techniques could be modified to handle this, but the resulting methods would be as impractical as for the case when all sides are cooperatively computing the same function.

2.1.2 Selective Private Function Evaluation

Selective Private Function Evaluation (SPFE) was introduced in (Canetti et al. 2001). In this problem, a client interacts with one or more servers holding copies of a database $x = x_1, \ldots, x_n$ in order to compute $f(x_{i_1}, \ldots, x_{i_m})$, for some function f and indices $i = i_1, \ldots, i_m$ chosen by the client. Ideally, the client must learn nothing more about the database than $f(x_{i_1}, \ldots, x_{i_m})$, and the servers should learn nothing. Various approaches for constructing sublinear-communication SPFE protocols are presented in (Canetti et al. 2001). Although our protocols do not yet do so, we anticipate that some of the techniques from the SPFE literature will be useful in carrying out future SSCC research; e.g., the requirement that the

server not know f in SPFE is similar to the requirement that f_A not be known to any of the other participants.

2.2 Mechanism Design

Mechanism design (MD) studies how private information can be elicited from independent agents by providing incentives to the participants. In other words, mechanism design is the art of designing rules of the game so that the participants are motivated to report their information truthfully and a desirable outcome is chosen. Conitzer and Sandholm (2002) show that the general MD problem is NP-complete.

The use of auctions for eliciting information and allocating resources such as securities is described by Harris and Raviv (1981). They derive the optimal allocation mechanism for a supplier, under the assumption of a unitary demand function, and a uniformly-distributed marginal willingness-to-pay for each retailer. An optimal auction procedure, where retailers submit quantity bids, was derived by Maskin and Riley (1989), under assumptions similar to ours.

One significant new research direction within MD is the blending of economics with the traditional distributed-computing notions of computational complexity and algorithmics (Feigenbaum and Shenker 2002): More specifically, the distributed algorithmic mechanism design (DAMD) model considers the agents participating in a distributed computation to be acting in their own selfish best interest (as in mechanisms), while also considering computational complexity and algorithmics (as in traditional distributed computing). One of the two open problems listed in (Feigenbaum and Shenker 2002) asks whether easy solutions can be shown for natural problems of interest. The present paper can be viewed as a step in that direction for some specific cases of supply-chain interactions.

Recently, the privacy of the agents within the DAMD framework has also been considered in general terms (Feigenbaum et al. 2002). We label this subfield DAMDP. Specific forms and applications of DAMDP, such as the special case of online auctions, have been analyzed. Naor, Pinkas, and Sumner describe an architecture for mechanisms for the cases of the Groves-Clarke mechanisms, elicit opinions from a group of independent experts, and Stable Matching (Naor et. al 1999). Our work can be viewed as a specific form of DAMDP applied to the supply-chain setting.

2.2.1 Private Auctions

There has been a large volume of work on the development of secure auctions, which is similar to SSCC research (Franklin and Reiter 1996,Naor et al. 1999, Elkind and Lipmaa 2003, De Decker et al. 2001, Brandt 2003)). Franklin and Reiter's work outlined many properties of an online auction, including properties that ensured that the auctioneer can extract the winning bid from the bidder and that it cannot extract anything from the losing bidders. The secrecy requirements were that the bids be protected until after the bidding period. Subsequent work enhanced the privacy of this scheme; one such scheme (Naor et al. 1999) which introduced an architecture that is useful for SSCC (we discuss this further in Section 3.3). Brandt introduced an auction mechanisms that allowed for the bidders of the auction to compute the winning price, and did not require an outside party to help with the auction (Brandt 2003). Surveys of online auction can be found in (Elkind and Lipmaa 2003, De Decker et al. 2001). In our work, we focus on privately computing quantities. However, the techniques in (Franklin and Reiter 1996) could be used to extend our schemes to include these properties.

2.3 Supply-Chain Management

Historically, supply-chain management research has focused on "centralized" policies for optimizing a supply-chain; i.e., decision-rules for optimizing a single objective function (e.g., system profit) under the assumption that all the information about the system (e.g., costs, capacity, inventory status) is available to a central planner. In mathematical terms, supply-chain research has historically focused on problems of the form optimize $f(x)$, where the input vector x is known and available to a single decision-maker. See (Muckstadt and Roundy 1993) and (Federgruen 1993), for examples of this research. Although this literature has contributed decision-rules for managing supply chains that employ centralized information and control, in fact, most real-world supply chains are managed, not by a single decision-maker, but by several decision-makers, each with their own, often incompatible, objective functions, and each using her/his own private information.

Today, research in supply-chain management is largely focused on multiple decision-makers with multiple objective functions, each formulating their own decision-rules on the basis of asymmetric information. In mathematical terms, this stream of research splits the traditional objective function $f(x)$ into separate objective functions $f_A(x_A, x_B)$ and

$f_B(x_A, x_B)$ for Alice and Bob, respectively, based on private inputs x_A and x_B. The intellectual roots of this new focus is auctions and other information-asymmetry models in economic game theory, and hence this research blends ideas from Supply Chains and Mechanism Design described earlier. See (Cachon 2004) for a recent survey of this work.

Meanwhile, in practice, information technology is facilitating information-sharing and decision-making in supply chains. Further, a few companies, most notably, Wal-Mart, have demonstrated the extraordinary payoffs from making decisions based on shared information. In particular, Wal-Mart's RetailLink has become the benchmark by which other supply chains are measured (http://www.walmart.com/cservice/awindex.gsp).

There is also a national program underway, sponsored by the Voluntary Intraindustry Commerce Standards (VICS) association to develop standards and procedures under which independent buyers and sellers can share plans, forecasts, and decision-making involving inventory replenishment. This program, called Collaborative Planning, Forecasting, and Replenishment (CPFR) has attracted the interest of literally hundreds of companies (http://www.cpfr.org/Members.html). Unfortunately, CPFR must overcome at least one major obstacle in order to achieve success: the reluctance of buyers and/or sellers to share private, proprietary information.

2.3.1 Supply-Chain Management Literature on Information-Sharing

Several researchers have examined the value of information-sharing in a supply-chain. Lee, Padmanabhan and Whang (Lee et al. 1997) have identified a phenomenon termed the "bullwhip effect" caused by distorted information in a supply chain. Lee, So and Tang (Lee et al. 2000) have shown that information sharing in a supply chain can greatly dampen the "bullwhip effect". Chen, et al. (Chen et al. 2000) quantified the impact of the "bullwhip effect" in a supply chain. They show that the bullwhip effect can be reduced by centralizing information. Iyer and Ye (Iyer and Ye 2000) assess the value of information sharing in a retail environment, where retailers share promotion information with their suppliers. Song and Zipkin (Song and Zipkin 1996) develop an inventory-replenishment policy to take advantage of information about supply conditions. Cachon and Fisher (Cachon and Fisher 2000) study the value of sharing demand and inventory level information in a supply chain. More recently, Aviv (Aviv 2002, Aviv 2001) has examined the effect of collaborative forecasting on supply-chain performance. This work, the literature on centralized decision-making, and the agency loss associated with decentralized decision-making, provide the supply-chain

motivation and foundation for our work.

2.4 Our Research

In summary, computer scientists have begun to integrate privacy-preserving techniques with mechanism design while supply-chain researchers have begun to integrate mechanism design into supply-chain interactions. Papers by Atallah et al. (2003) and Clifton et al. (2003) are the first to apply privacy-preserving computation techniques to supply-chain research, but without considering mechanism design. This paper bridges all three approaches, i.e., secure multi-party computation (SMC), mechanism design (MD) and Supply-Chain Management (SCM) research. In doing so, we provide a framework for further supply chain research that incorporates privacy preservation of participants' information and incentives to participate in supply-chain transactions.

Although, at first glance, our applications appear to be merely auctions, there are important differences: In our capacity-allocation application, the "order quantities" of the retailers and the corresponding "valuation (bids)" are endogenous (i.e., driven by the supply-chain setting being modeled, e.g. a newsvendor setting), rather than being given values exogenous to the "auction". Further, the supplier's K units of capacity are not necessarily all sold. In a similar manner, our market-clearing applications are driven by the supply and demand functions of the participants, whereas in an auction framework, a fixed number of units are being sold.

3 Security Model and System Architectures

In this section, we outline what is meant by "secure" computation and then discuss general results regarding secure simulation of circuits. Finally, we discuss various architectures for Privacy-Preserving Supply-Chain Management.

3.1 Security Model

We begin by discussing a standard security model (Canetti 2000 and Goldreich 2004). At a high level, a protocol "securely" implements a function f if the information that can be learned by engaging in the protocol, could be learned in an ideal implementation of the protocol where the functionality was provided by a trusted oracle. We also consider two types of adversaries: i) semi-honest adversaries that will follow the protocol exactly but

try to compute "extra" information; and ii) malicious adversaries that will deviate from the protocol at any step in order to gain additional information or to control the outcome. The types of things that an adversary could do include: i) substitute inputs (ideal, semi-honest, and malicious), ii) deviate at an intermediate step (malicious), and iii) terminate early (malicious ideal and malicious). Our protocols do not attempt to address item (iii) In a sense, this is the fairness constraint (it may be possible for one party to learn the result and then terminate the protocol, not allowing the other party to learn the result). The reason that such attacks are ignored is that the solutions for such tasks are complex and do not completely solve the problem. See discussion below.

We now formally define the notions above. We do this by defining the notion of an IDEAL-model adversary (one for the situation where there is an oracle) and a REAL-model adversary for the protocol Π, and then assert that a protocol is secure if the two executions are computationally indistinguishable. We focus on defining the case for two-party protocols (and refer the reader to (Canetti 2000) for multiple parties). Assume the Π computes $f : \{0,1\}^\star \times \{0,1\}^\star \rightarrow \{0,1\}^\star$

Definition of IDEAL model:

The ideal model can be viewed as two Probabilistic Polynomial Time (PPT) algorithms (A, B) each of which is composed of two parts A_I and A_O (and B_I and B_O), and the execution of the protocol is as follows:

1. Alice (Bob) sends $A_I(X_A, r_A)(B_I(X_B, r_B))$ to the oracle (where r_A and r_B are Alice and Bob's respective coin flips).

2. The oracle evaluates $f(A_I(X_A, r_A), B_I(X_B, r_B))$ obtaining output (Y_A, Y_B), and sending Y_A (Y_B) to Alice (Bob).

3. Alice (Bob) outputs $A_O(X_A, r_A, Y_A)(B_O(X_B, r_B, Y_B))$.

Alice is said to be *honest* if $A_I(X_A, r_A) = X_A$ and $A_O(X_A, r_A, Y_A) = Y_A$ (a similar definition holds for Bob). We say that an adversary is *admissible* if at least one party is honest (i.e., we do not concern ourselves with adversaries that corrupt both parties). We now define the ideal model's view in the case where Bob is honest (an analogous definition occurs when Alice is honest), there are two cases:

- Alice does not terminate, then $IDEAL_{A,B}(X_A, X_B) = (A_O(X_A, r_A, Y_A), f(A_I(X_A, r_A), X_B))$.

- Alice terminates, then $IDEAL_{A,B}(X_A, X_B) = (A_O(X_A, r_A, Y_A), \perp)$.

We now define the actual execution for a protocol Π that implements the function f.

Definition of REAL model:

In a real model the parties are arbitrary PPT algorithms (A', B'), where for semi-honest adversaries the output function is arbitrary and for malicious adversaries the parties can behave arbitrarily. The adversaries are admissible if at least one party uses the algorithm specified by protocol Π. We define the interaction of protocol Π by $REAL_{\Pi,A',B'}(X_A, X_B)$ as the output from the interaction of $A'(X_A)$ and $B'(X_B)$ for protocol Π.

As is usual, we assert a protocol Π is secure if there is an ideal-model adversary that is as powerful as any real-model adversary up to a negligible degree. To define what is meant by this we use the standard definition of computational indistinguishability (Goldreich 2001):

Definition of Computational Indistinguishability:

We say that two random variables X and Y are computationally indistinguishable if for any PPT algorithm D and any polynomial p, it holds that:

$|(Pr(D(X) = 1)) - (Pr(D(Y) = 1))| < \frac{1}{p(n)}$

Definition of security: We say that a protocol Π securely evaluates a function f if for any admissible adversary in the real model (A', B'), there exists an admissible ideal-model adversary (A, B) such that $IDEAL_{A,B}(X_A, X_B)$ and $REAL_{\Pi,A',B'}(X_A, X_B)$ are computationally indistinguishable.

3.2 Known Results for Circuit Simulation

in this section we summarize known results for secure circuit evaluation. The main part of this is Theorem 1. While we focus primarily on the passive and malicious models, there are many extensions of these protocols to stronger forms of adversaries (Canetti 2000, Canetti et al. 2002).

Theorem 1 *When given a boolean circuit with 2-ary gates with m gates, n inputs, and depth d, then there exist techniques for evaluating the circuit in a private manner for non-adaptive computationally-bounded adversaries in the following cases (Goldreich 2004):*

1. **Two-party, passive***: (Yao 1986) showed that this can be done with with n 1-out-of-2 OTs, $O(m)$ communication, $O(m)$ evaluations of a pseudorandom function (such as AES, which can be done at disk transfer speeds, and $O(1)$ rounds. This is secure against one specified party being malicious (Naor et al. 1999).*

2. **Multi-party ($k \geq 3$), strict minority of malicious parties, assuming a broadcast channel**: *(Goldreich et al. 1987) showed that it was possible to evaluate such a circuit with $O(k^2 m)$ communication and $O(d)$ rounds. There has been work to reduce this to constant rounds (Rogaway 1991), but these solutions are believed to be impractical.*

3. **Multi-party ($k \geq 2$) any number of malicious parties, assuming broadcast channel, early termination not considered a failure of security**: *it is possible to evaluation such a circuit with $O(mk^2)$ communication and $O(d)$ rounds.*

We now discuss some consequences of Theorem 1. First, we believe that the solutions with early termination are problematic, since the solutions have the results split among all parties; and, even if the result is computed, there is the problem of distributing the results to the correct parties. The solutions for this problem do not wholly solve the problem and are very complex; but, given certain trust assumptions for two parties, there are efficient solutions to this problem. The primary consequence of Theorem 1 is Given a circuit for evaluating a problem, there are many ways to compute the values securely. In the next section, we discuss various architectures for SSCC and discuss the effects of Theorem 1 on these architectures.

We now outline various circuit complexities:

1. *Adding two m-bit numbers*: Adding or subtracting two m-bit numbers can be done optimally using circuits of size $O(m)$ gates and depth $O(\log m)$. (Ofman 1963).

2. *Adding k m-bit numbers*: Requires $O(m \log k)$ gates and $O(\log k \log m)$ depth.

3. *Multiplying two m-bit numbers*: The practical circuits for multiplication have size $O(m^2)$ and depth $O(\log m)$. Although there are asymptotic improvements to these circuits, they come at the cost of huge constant factors; the asymptotically best of them (and the worst in terms of having impractically large constant factors) is a circuit of size $O(m \log m \log \log m)$ and depth $O(\log m)$ derived from the textbook Schoenhage-Strassen integer multiplication algorithm (Schoenhage and Strassen 1971; Aho et. al 1974) (which is itself of mainly theoretical interest, and not used in practice).

4. *Integer division of two m-bit numbers*: Just as for multiplication, the practical circuits for multiplication have size $O(m^2)$ and depth $O(\log m)$. However, unlike multiplication,

the "impractical" (but asymptotically better) circuits achieve either $O(m \log m \log \log m)$ using the Schoehage-Strassen technique (Alt 1988; Beame et. al 1984) or depth $O(\log m)$ (Beame et. al 1984) but not simultaneously. The division circuits that come closest to achieving "close to simultaneity' of these size and depth bounds have size $O(m \log m \log \log m)$ and depth $O(\log m \log \log m)$ (Reif and Tate 1989).

5. *Comparing m bit numbers:* Comparisons of the form $\geq, \leq, >, <, =, \neq$ requires $O(m)$ gates and $O(\log m)$ depth.

6. *Sorting n numbers with m bits:* A practical sorting network is one that implements Batcher's sort (Batcher 1968) and has size $O(mn \log^2 n)$ and depth $O(\log m \log^2 n)$. An asymptotically better (but impractical due to its large constant factors) sorting network is the AKS one (Ajtai et. al 1983) that has size $O(mn \log n)$ and depth $O(\log m \log n)$. We discuss sorting in more detail in Appendix B.

3.3 Architectures

We describe four architectures for SSCC: multi-party, untrusted third party, multiple untrusted third parties, and two untrusted third parties. There are several issues that need to be considered for each of these functions, including: efficiency, simplicity (since ease of implementation is important), resilience against malicious parties, verifiability, and trust assumptions.

In the multi-party model, all of the parties engage in a secure protocol, using standard techniques to learn the result. The known solutions for such a protocol are very expensive and are complex. The protocols can be made resilient against any passive adversary, or a malicious party that controls fewer than half of the parties, are corrupt, or any number of parties if early termination is not considered a problem. However, as described above early termination is a problem. Thus, the solution requires strictly fewer than half of the parties to be malicious. The parties can verify the outputs after the protocol, by engaging in another protocol. However, this would require a majority of parties to participate. Furthermore, any majority of the parties can cooperate together to obtain all other parties information. Thus, for example a group of retailers could gang up on the supplier or a supplier could create a bunch of fake retailers and then learn the suppliers private data.

In the untrusted third-party model (Naor et. al 1999; Cachin 2001), there would be a party, which we call Ursula, that is trusted only enough not to be malicious and not to

collude with the supplier. We stress that this party does not need to learn the information of the parties. Essentially, each party would send Ursula and the Supplier a split of their data (so that neither knew anything, but combined they know the values). Ursula and the supplier would engage in a secure circuit simulation (as in Naor, et al. 1999) to learn the result. The result would be computed in a split fashion and each party's component would be sent to that party. In this case, the untrusted party would generate the circuit as this allows the supplier to be malicious. After the protocol, when a retailer makes its order it can easily be verified by Ursula and the supplier that his order is accurate. The trust assumption is that such a untrusted party can be found.

A hybrid of the first two solutions would be for the users to trust multiple Ursulas and the supplier. This is likely to be faster than the first solution, because, in most cases, the number of Ursulas would be smaller than the number of parties. Everything is the same as the first case, except that the trust assumptions are much smaller (e.g., trusting outsiders versus trusting those who have something to gain).

Another hybrid is to have two Ursulas instead of the supplier and one Ursula, and to use the mechanisms such as in 2. The only difference between the two is that the trust assumptions are smaller because both parties are outsiders.

We recommend that either an untrusted third party or a two untrusted third-party solution be used. The primary reason is that the techniques for such solutions are much faster than those for the multiple-party solutions. Furthermore, the verification process is much simpler in these models. While we recommend these solutions, our protocols are not tied to any particular architecture. We present boolean circuits for the supply-chain problems, and, by Theorem 1, the results can be computed securely. We leave for future work the development of protocols that are more efficient than these circuit constructions. In presenting such circuits, we will provide the an analysis on the number of gates, the number of inputs, and the depth of the circuit.

4 Secure Protocols for Capacity Allocation

Consider a single supplier who sells to N retailers. The supplier sets the prices the retailers pay, but has a limited capacity, K, available to satisfy retailer orders. The retailers operate in non-competing retail markets and have demand curves based on the price that they charge in their market. Everyone, including the supplier, knows the form of the re-

tailer's demand curve, but each retailer's demand-curve parameters (i.e., θ) is its private information. The goal of this section is to devise secure protocols for allocating the supplier's fixed capacity in a way that maximizes the supplier's revenue and does not reveal any participant's private parameters. We first introduce a mechanism-design framework which enables revenue maximization for the supplier. This requires specification of a pricing mechanism set by the supplier (as a function of retailer order quantities) and a capacity-allocation mechanism (also a function of retailer orders) which is incentive compatible. In other words, the specified pricing and capacity-allocation mechanisms will motivate the retailers to submit orders that maximize their respective profits. We show that the linear and proportional capacity-allocation mechanisms are optimal for the supplier under various business scenarios. Given this background, in section 4.2, we describe secure protocols for linear and proportional capacity-allocation mechanisms. We then extend our analysis, in section 4.3, to secure capacity-allocation mechanisms that maximize supplier revenue when retailer's have a step-function demand curve.

4.1 Mechanism Design Framework for Capacity Allocation

Our framework presented below is based on the analysis provided by Maskin and Riley (1989), and Deshpande and Schwarz (2005). Let K be the supplier's fixed capacity. The supplier recognizes that the order from each individual retailer depends on the price it charges and on retailer's private information parameter, which is hidden from the supplier. We model this private retailer information by the scalar parameter θ. Although θ is not known to the supplier, we assume that the supplier has a prior, with a density $f(\cdot)$ on the support $[\underline{\theta}, \overline{\theta}]$. Thus, the revenue function for retailer i, denoted $R_i(q_i, \theta_i)$, is a function of its allocated quantity q_i and its private information parameter θ_i. Retailer i observes its θ_i, but not the values of other retailers, labeled θ_{-i}. The retailer's profit equals its revenue $R_i(q_i, \theta_i)$ minus the purchasing cost (for q_i units) from the supplier.

A mechanism-design approach (Fudenberg and Tirole, 2000) is used to formulate the supplier's problem. In this direct-revelation mechanism, the supplier asks the retailers to reveal their type θ_i, and implements the pricing-and-allocation policy $\{P(\theta_i, \theta_{-i}), Q(\theta_i, \theta_{-i})\}$ based on the types revealed by the retailers. Here $Q(\theta_i, \theta_{-i})$ represents the quantity-allocation function while $P(\theta_i, \theta_{-i})$ represents the retailer purchasing cost based on the retailer's announcements of their individual type θ. From the revelation principle (see Fudenberg and Tirole, 2000, p256.), the supplier can, without any loss in profits, restrict its attention to

truth-telling mechanisms, where it is optimal for each retailer to announce his true information parameter θ_i. At the Dominant Strategy equilibrium of the direct revelation game, truth-telling is the optimal strategy for each retailer independent of the types of all other retailers. A pricing-and-allocation policy is implementable if it is incentive compatible, and provides non-negative profits to the retailers. The following theorem states the condition under which a mechanism $\{P(\theta_i, \theta_{-i}), Q(\theta_i, \theta_{-i})\}$ is implementable.

Theorem 2 *The pricing-and-allocation mechanism* $\{P(\theta_i, \theta_{-i}), Q(\theta_i, \theta_{-i})\}$ *is incentive compatible* if and only if *conditions (1) and (2) hold. In addition, condition (2) is a sufficient condition for individual rationality (i.e., guarantee non-negative profits for the retailers).*

$$\int_x^{\theta_i} R_\theta(Q(x, \theta_{-i}), \theta) d\theta \leq \int_x^{\theta_i} R_\theta(Q(\theta, \theta_{-i}), \theta) d\theta \tag{1}$$

$$P(\theta_i, \theta_{-i}) = R(Q(\theta_i, \theta_{-i}), \theta_i) - \int_{\underline{\theta}}^{\theta_i} R_\theta(Q(\theta, \theta_{-i}), \theta) d\theta \tag{2}$$

The computation of prices for these allocation mechanisms is straightforward from equation 2. Note that the price mechanism $P(\theta_i, \theta_{-i})$ is an algebraic manipulation of the quantity allocation mechanism $Q(\theta_i, \theta_{-i})$. Hence, it is straightforward to construct a secure protocol for the price mechanism, if one can construct a secure protocol for the quantity allocation mechanism $Q(\theta_i, \theta_{-i})$. For example, when retailers face downward sloping linear demand, the price mechanism can be written as:

$$P(\theta_i, \theta_{-i}) = Q(\theta_i, \theta_{-i})(\theta_i - Q(\theta_i, \theta_{-i})) - \int_{\underline{\theta}}^{\theta_i} Q(\theta, \theta_{-i}), \theta) d\theta \tag{3}$$

The above theorem provides the optimal quantity allocation and pricing mechanism for the supplier under very general conditions. We now focus on two special cases of the optimal quantity allocation mechanism: The linear-allocation and proportional-allocation mechanisms, and design secure protocols for them.

Theorem 3 *(Maskin and Riley, 1989) If retailers face deterministic downward sloping linear demand, with the intercept of the demand-curve* θ *private to the retailers, then the linear allocation mechanism (defined below) is optimal for the supplier.*

Definition 1 (Linear Allocation) *Index the retailers in decreasing order of their order quantities,i.e.* $q_1 \geq q_1 \geq ... \geq q_N$. *Retailer* i *is allocated* $Q_i(q_i, q_{-i})$, *where*

$$Q_i(q_i, q_{-i}) = q_i - \frac{1}{\bar{n}} \max\{0, \sum_{i=1}^{\bar{n}} q_i - K\} \quad if \quad i \leq \underline{n}$$

$$Q_i(q_i, q_{-i}) = 0 \quad \textit{if} \quad i > \bar{n}$$

where $\bar{n}$ is the largest integer such that $Q_i(q_i, q_{-i}) \geq 0$ for all i.

Intuitively, linear allocation is simply an "equal sharing of the pain" among the buyers, with the understanding that if that pain exceeds the q_i of a buyer then that buyer drops out. This is why $\bar{n}$, the number of buyers who are allocated a positive share of K, can be less than the total number of retailers (i.e., $\bar{n} \leq N$). These $\bar{n}$ retailers each get the *same* amount less than their order; i.e., the "pain" inflicted on each buyer is equal to (total shortage)/$\bar{n}$ where the total shortage equals what the $\bar{n}$ buyers ordered minus K.

Deshpande and Schwarz also prove the structure of the optimal policy for the supplier if the retailers are "newsvendors", i.e., retailers, like real newsvendors, face demand generated from a probability distribution.

Theorem 4 *(Deshpande and Schwarz, 2005) If retailers are newsvendors with a normal demand distribution with mean θ, and an exponential prior on θ, then the linear allocation mechanism is optimal for the supplier.*

Theorem 5 *(Deshpande and Schwarz, 2005) If retailers are newsvendors with a uniform demand distribution on $[0, \theta]$, and a Pareto supplier's prior on θ, then the proportional allocation mechanism (defined below) is optimal for the supplier.*

Definition 2 (Proportional Allocation) *Retailer i is allocated $Q_i(q_i, q_{-i})$ where $Q_i(q_i, q_{-i}) = \min\{q_i, \frac{Kq_i}{\sum_1^N q_i}\}$. Here N is the number of the retailers, and K is the total capacity that the supplier can provide.*

4.2 Secure Protocols for Linear and Proportional Allocation Mechanisms

The allocation mechanisms described above and their corresponding pricing policies in (Deshpande and Schwarz 2002) might be appropriate if allocation decisions are made once and only once. However, if allocation decisions are repeated, say, weekly over a selling season of several months with the same θs, then there would be no incentive for the retailers to participate after the first allocation, since, they would subsequently make no profits, because their θ's would have been revealed as part of the first allocation process. Specifically, note that either allocation mechanism motivates each retailer i to order its profit-maximizing

quantity ($q_i = \theta_i/2$), thereby revealing its $\theta_i = 2q_i$. Note further, that any stationary non-linear retailer demand curve would require more repetitions, but, again, the supplier would eventually be able to infer each retailer's demand function parameters.

In what follows we describe protocols for the above allocation mechanisms that are secure as long as $K \leq \sum_i q_i$ (more on this below). These protocols use the retailer order quantities $q_i, i = 1, \ldots, N$ as inputs and compute the allocations, $Q(q_1, q_2, \ldots, q_N)$ defined above, without revealing any retailer's private information parameter θ_i either to the supplier or to the other retailers. Since these protocols do not reveal the individual retailers' private information parameter, these protocols can be used repeatedly.

4.2.1 Information Required for Secure Capacity-Allocation Protocol

Before the protocol every retailer knows his profit-maximizing quantity q_i $(= \theta_i/2)$, the supplier knows her capacity K. After the protocol is completed, every retailer knows the quantity $Q_i(q_i, q_{-i})$ he will be allocated under the allocation policy (linear or proportional), $\bar{n}$, and nothing else. In particular, the protocol itself does not reveal the individual q_i, $\sum_i q_i$, or K.

The condition that $\sum_i q_i \geq K$ in order for the protocol to be secure is related to inverse optimization, which is discussed in a subsection 4.2.2. Nonetheless, we will discuss this particular condition here because it is fundamental to how all SSCC protocols are designed: That is, to isolate the decision-making process(es) from the implementation process(es). For example, in the capacity-allocation scenario, in order to implement the allocation decision, the supplier must be told how much capacity to allocate to each retailer. If $\sum_i q_i < K$, then, by observing this fact, and knowing how the allocation mechanism works, the supplier can trivially infer each retailer's θ_i after learning what quantity q_i to provide to each retailer i, as in the *symmetric* information scenario above. Although conditions of this type limit the privacy-preserving potential of SSCC in general, such limitations are obviously unavoidable. More important, in general, SSCC protocols will preserve privacy when it is most important to do so; for example, when capacity is tight in the capacity-allocation scenario. Finally, privacy-preserving SSCC protocols can be used to determine circumstances when use of the protocol would be insecure. We describe such a pre-processing protocol for the capacity-allocation scenario in the inverse optimization discussion in section 4.2.2.

4.2.2 Secure Linear-Allocation Protocol

In this section we present a circuit for determining Linear-Allocation (see Figure 1).

Input: Each retailer R_i has a quantity q_i and the supplier has capacity K.
Output: Each retailer has $Q_i(q_i, q_{-i})$ and the supplier learns $\sum_{i=1}^{N} Q_i(q_i, q_{-i})$.
Description:

(1) For each retailer R_i define $active_i \leftarrow 1$
(2) for $i \leftarrow 1$ to N do
(3) $\quad \bar{n} \leftarrow \sum_{i=1}^{N}(active_i)$
(4) $\quad D \leftarrow (\sum_{i=1}^{N}(q_i) - K)$
(5) $\quad P \leftarrow (\lceil \frac{D}{\bar{n}} \rceil)$
(6) $\quad$ for $j \leftarrow 1$ to N do
(7) $\quad\quad active_j \leftarrow (P < q_j)$
(8) $\quad\quad q_j = q_j \times active_j$

Note: that in step (7) and (8) we treat a boolean value as an integer that is $\{0, 1\}$.

Figure 1: Protocol for Linear-Allocation

Constructing the Circuit

The circuit is constructed by forming N sub-circuits each with $(Nm + N + m)$ inputs and outputs and ordered in a sequence where the output of the ith sub-circuit is the input to the $(i + 1)$st sub-circuit. The inputs are $active_i$ and q_i for each participant and the supplier's capacity K. Clearly, each of these sub-circuits can be constructed with a set of arithmetic gates.

Corollary 1 *The protocol in Figure 1 can be evaluated securely, as in Theorem 1.*

Complexity Analysis

We assume that the variables $\sum_{i=1}^{N}(q_i)$ and K can be represented by m bits. Furthermore, for simplicity we say that the individual q_i's are also represented by m bits. It is not difficult to show that the variables D and P also can be represented with m bits and $\bar{n}$ requires $\log N$ bits. Step (6)-(8) require $O(Nm)$ gates and steps (3)-(5) require $O(m^2 + \log N)$ gates. Thus, steps (1)-(8) require $O(N^2m + Nm^2)$ gates. There are $O(Nm)$ input and output wires in the circuit. The depth of the circuit is $O(N \log M \log N)$.

Inverse-Optimization

Table 1 summarizes the inverse optimization of the protocol. Note that we let $\mathcal{P}$ represent the set of active retailers. As noted above, the supplier can determine the retailers' θ_i's if

Table 1: Who knows what: linear allocation

Who knows what	K	q_i	pain per active retailer $(\max\{0, \sum_{i\in\mathcal{P}} q_i - K)/\bar{n}\}$	$Q_i(q_i, q_{-i})$	$\bar{n}$
Supplier	$\surd$			$\surd$	$\surd$
Retailer $i(i \in \mathcal{P})$		$\surd$	$\surd$	$\surd$	
Retailer $i(i \notin \mathcal{P})$		$\surd$		$\surd$	

$\sum_i q_i < K$. Testing whether $K > \sum_i q_i$ or not can be securely done as a pre-processing step with a simple circuit of size $O(Nm)$. This step would check to see if $K > \sum_{i=1}^{N} q_i$ without revealing the individual quantities. Unless an a group of retailers knows $\sum_{i\in P} q_i$ it cannot learn K. Since $K = \sum_{i\in P} q_i$, the better a group of adversaries can estimate this value, the better they can estimate K. If the retailers would like to compute such a quantity without revealing their private information, then they could do so by engaging in an SMC protocol.

4.2.3 Secure Proportional-Allocation Protocol

In this section we present a circuit for determining Proportional-Allocation (see Figure 2).

Input: Each retailer R_i has a quantity q_i and the supplier has capacity K.
Output: Each retailer has their own q_i' and the supplier has all of these values.
Description:

(1) $D \leftarrow (\sum_{i=1}^{N}(q_i))$
(2) for $i \leftarrow 1$ to N do
(3) $\quad q_i' \leftarrow \lfloor \frac{q_i \times K}{D} \rfloor$

Note: that in step (3) the result could be computed as a floating point at the expense of a larger circuit.

Figure 2: Protocol for Proportional-Allocation

Constructing the Circuit

The circuit is trivial to construct.

Corollary 2 *The protocol in Figure 2 can be evaluated securely, as in Theorem 1.*

Complexity Analysis

We assume that the variables $\sum_{i=1}^{N}(q_i)$ and K can be represented by m bits. Furthermore, for simplicity we say that the individual q_i's are also represented by m bits. Clearly, the size of the circuit is $O(Nm^2)$. The depth of the circuit is $O(\log M \log N)$.

Table 2: Who knows what: proportional allocation

Who knows what	K	q_i	q'_i	$\sum_{i=1}^{N} q_i$	$K/\sum_{i=1}^{N} q_i$
Supplier	$\surd$		$\surd$		
Retailer i		$\surd$	$\surd$	$\surd$	$\surd$

Inverse-Optimization

Table 2 summarizes the inverse optimization of the protocol. The more information the retailers have about $\sum_i q_i$, the better they can infer the value K. Clearly, to know K exactly they would need to know the value $\sum_i q_i$, which would require all of them to collude. Now if the a group of retailers can guess a value in the range $[(1-\epsilon)\sum_i q_i, (1+\epsilon)\sum_i q_i]$, than they can estimate K, and the estimate will be in $[(1-\epsilon)K, (1+\epsilon)K]$.

The supplier cannot "reverse-engineer" any of the q_i's as a result of knowing row 1 of Table 2. This is because the protocol is run only in case $\sum_i q_i \geq K$, and therefore the supplier does not know the "fractional pain" $K/\sum_i q_i$, nor can he compute it from the protocol. That is, he cannot tell how much $Q(q_i, \bar{n})$ falls short of the originally requested q_i. On the other hand, if one retailer j colluded with the supplier then the supplier would know the fractional pain for that retailer (it is q'_j/q_j), and, from this, the supplier could then figure out every q_i. It is worth noting that a malicious supplier could insert a fake retailer into the mix with small capacity and then learn all other retailers quantities.

4.3 Computing Price-Allocation Securely

We present a boolean circuit for computing $P(\theta_i, \theta_{-i})$, for general allocation mechanisms. It is possible to create allocation mechanism specific mechanisms that are more efficient than this general construction, but we leave this as future work. The circuit we use utilizes circuits for computing $Q(\theta_i, \theta_{-i})$, which are described in the following sections. Figure 3 shows a sketch of how equation (3) is approximated by using k intervals of size $\Delta = \frac{\overline{\theta}-\underline{\theta}}{k}$. We show this for a single retailer, but it is easy to construct this so that all retailers' values are computed.

4.4 Capacity Allocation with Price-Quantity Orders

In the previous sub-section we considered secure capacity allocation when retailer orders were driven by continuous demand curves. We now extend our analysis to the case where retailer demand curves can be modeled as step functions. Each retailer's demand curve is given by

Description:

(1) $q \leftarrow Q(\theta_i, \theta_{-i})$
(2) $p \leftarrow (q)(\theta - q)$
(3) $v \leftarrow \underline{\theta}$
(4) for $j \leftarrow 1$ to k
(5) $\quad b_j \leftarrow (\theta_i \geq v)$
(6) $\quad p \leftarrow (Q(v, \theta_{-i}) \times \Delta \times b_j) + p$
(7) $\quad v \leftarrow v + \Delta$
(8) endfor

Note: that in step (5) and (6) we treat a boolean value as an integer that is $\{0, 1\}$.

Figure 3: Protocol for Computing Pricing-and-allocation mechanism

a set of alternative price-quantity pairs (p_i, q_i). The supplier has to decide for each retailer, which (if any) of its alternative (p_i, q_i) pairs, called orders, to supply in order to maximize his revenue, while satisfying his capacity constraint. Hence, we also call this framework the "pick-and-choose" framework.

Information Required for Pick-and-Choose

The relevant information from each buyer i is its set of price-quantity pairs (p_{1_i}, q_{1_i}), (p_{2_i}, q_{2_i}), $\cdots$, (p_{j_i}, q_{j_i}). The goal for the supplier is to set the prices paid by the buyers so as to maximize its revenue, i.e. $\max \sum_i p_i q_i$, subject to the supply constraint $\sum_i q_i \leq K$.

The problem of maximizing supplier revenue under alternative (p_i, q_i) retailer orders and capacity constraints, was investigated by Sandholm and Suri in (Sandholm and Suri 2001). They proved that it is $\mathcal{NP}$-Complete, and devised a pseudo-polynomial algorithm to solve it. It is not difficult to construct a circuit for the protocol in (Sandholm and Suri 2001) that solves their problem in a secure manner with similar complexity. We provide the details in Appendix A.

5 Market-Clearing Transactions with Linear Supply Curves

In the previous section we assumed that supplier capacity is fixed and exogenously specified. In this section, we consider capacity as a choice variable based on a linear supply curve.

For a single profit-maximizing producer, the supply quantity is chosen so that marginal production cost equals its marginal price. Similarly, for a single utility-maximizing consumer, the demand quantity is chosen so that the consumer's marginal utility for that quantity equals the marginal price. Market equilibrium is defined as the situation when the amount

the consumers are willing to buy is the same as the amount the producers are willing to sell.

We consider two general models: One where all buyers pay the same unit price to the supplier (non-discriminatory pricing), and another where different buyers might pay different prices to the supplier (discriminatory pricing).

5.1 Market-Clearing Transactions with Non-Discriminatory Pricing

In this model, a supplier charges a uniform selling price to all the buyers. Each buyer i has a single price-quantity pair (p_i, q_i) expressing his demand for q_i units at a unit price of p_i or less, or zero units if the supplier's price exceeds p_i. The supplier has a supply curve $q = p + \theta$ and wants to choose a single price $\hat{p}$, the price from its supply curve that corresponds to the total demand $\sum_{i=1}^{n} q_i$.

Under the rules of the market, each buyer's (p_i, q_i) pair is not to be revealed to any other buyer. Further, the supplier is to remain ignorant of any buyer's individual price-quantity pairs before setting her price, thereby facilitating a policy of non-discriminatory pricing.

5.1.1 Information Required for Non-Discriminatory Pricing

The relevant information from each buyer i is his price-quantity pair (p_i, q_i). However, the supplier is not to know the total demand of the buyers before setting her price. After the uniform price $\hat{p}$ is announced to everyone, only those buyers i whose price p_i is lower than $\hat{p}$ are allowed not to buy, and those buyers i whose $p_i \geq \hat{p}$ are not allowed to increase their q_i. This is achieved by having the retailer commit to the value before the protocol and then proving that the value matches their commitment. This is discussed in more detail in Section 5.3.

5.1.2 Secure Non-Discriminatory Pricing Protocol

In this section we present a circuit for determining Non-Discriminatory Pricing (see Figure 4) and thus this quantity can be evaluated in a secure manner with the methods outlined in Theorem 1. Before we introduce the protocol, we need to introduce a couple of circuits: $COMPAREGT((p_1, q_1), (p_2, q_2))$ which compares to price-quantity pairs and ii) $SORT((p_1, q_1), \ldots, (p_n, q_n))$ which sorts the list of price quantity pairs in ascending order using $COMPAREGT$ (we discuss this in more detail in Appendix B). We say that a (p_1, q_1) is larger than (p_2, q_2) iff $(p_1 > p_2)$ or $((p_1 = p_2)$ and $(q_1 < q_2))$.

Input: Each retailer R_i has a quantity q_i and a price p_i and the supplier has capacity θ.
Output: The retailers learn the value $\hat{p}$, and the supplier learns q_i for retailer R_i iff $p_i \geq \hat{p}$.
Description:

(1) $\{(\bar{p}_1, \bar{q}_1), \ldots, (\bar{p}_N, \bar{q}_N)\} \leftarrow SORT(\{(p_1, q_1), \ldots, (p_N, q_N)\})$
(2) $(\tilde{p}, \tilde{q}) \leftarrow (\bar{p}_1, \bar{q}_1)$
(3) $\hat{p} \leftarrow (\sum_{i=1}^{N}(q_i) - \theta)$
(4) for $i = 1$ to $(N-1)$ do
(5) if $\hat{p} \leq p_i$ then $(\tilde{p}, \tilde{q}) \leftarrow (\tilde{p}, \tilde{q})$
(6) else $(\tilde{p}, \tilde{q}) \leftarrow (p_{i+1}, q_{i+1})$ endif
(7) $\hat{p} \leftarrow \hat{p} - p_i$
(8) endfor
(9) if $\hat{p} \leq p_N$ then $(\tilde{p}, \tilde{q}) \leftarrow (\tilde{p}, \tilde{q})$
(10) else $(\tilde{p}, \tilde{q}) \leftarrow (p_N + 1, 0)$ endif
(11) for $i = 1$ to N do
(12) $b_i \leftarrow (COMPAREGT((p_i, q_i), (\tilde{p}, \tilde{q})))$
(13) $p_i \leftarrow b_i \times p_i$
(14) $q_i \leftarrow b_i \times q_i$
(15) endfor

Figure 4: Protocol for Non-Discriminatory Pricing

Constructing the Circuit

While the circuit is more complex than the circuits above, it is not difficult to construct. Step (1) can easily be done with a sorting circuit (as described in Appendix B). There are N subcircuits that operate on the output of the sorting circuit's results that find the first point where the output is the pair $(\tilde{p}, \tilde{q})$, which are trivial constructions. The final part of the circuit, steps (11)-(14), can easily be constructed with N subcircuits that perform a comparison and two multiplications; note this last step can be done in parallel.

Corollary 3 *The protocol in Figure 4 can be evaluated securely, as in Theorem 1.*

Complexity Analysis

We assume that the variables $\sum_{i=1}^{N}(q_i)$ and θ can be represented by m bits. Furthermore, for simplicity we say that the individual q_is and p_is are also represented by m bits. Step (1) requires $O(mN\log^2 N)$ gates, Steps (2)-(10) require $O(Nm)$ gates, and Steps (11)-(15) require $O(Nm)$ gates. Thus the circuit has size $O(mN\log^2 N)$ gates. The depth of the circuit is $O(N \log M)$.

Note on Ties

Table 3: Who knows what: non discriminatory pricing

Who knows what	(p_i, q_i)	(p_j, q_j) $i \neq j$	θ	$\sum_{i \in \mathcal{P}} q_i$	$\hat{p}$
Supplier			$\surd$	$\surd$	$\surd$
Buyer i	$\surd$				$\surd$

Here we remark what happens in our protocol, when there are ties among users (i.e., they have the same price quantity pairs). The interesting case is when the values are split by the dividing point found by the circuit. In this situation the dividing point will be the pair where there is a tie, and thus all ties as this point will be filtered out as the comparison is strictly greater than. Thus, the result will still clear the market, and this is the fairest way to break a tie. This is because all participants learn the price for the transaction, and a party that matches the price but does not get their quantity will be unhappy with the process.

Inverse-Optimization

Table 3 summarizes the inverse optimization; we let $\mathcal{P}$ represent the retailers that are active after the evaluation of the circuit.

Note that no buyer i reveals to the supplier both p_i and q_i, and that no buyer knows $\sum_{i=1}^{N} q_i$. An individual buyer cannot compute the supplier's θ unless he colludes with all the other buyers: In the case of such a collusion, a buyer could figure out $\sum_{i \in \mathcal{P}} q_i$ which, together with his knowledge of $\hat{p}$, would enable him to learn θ. Note that if a set of buyers can estimate $\sum_{i \in \mathcal{P}} q_i$, then they can estimate θ, where the accuracy of the result is the accuracy of their original estimate.

The seller clearly cannot reverse-engineer any of the q_i's as a result of knowing row 1 of Table 3.

5.2 Market-Clearing Transactions with Discriminatory Pricing

Market-clearing transactions with discriminatory pricing mechanisms with supply curves and step-function buyer orders were analyzed by Sandholm and Suri (Sandholm and Suri 2002). They show that the problem of finding a supplier profit-maximizing market-clearing solution is $\mathcal{NP}$ complete. Here we propose a secure version of "All-or-none" framework involving a set of price-quantity orders. The supplier has to accept or reject all orders based on the profitability of all orders based on his supply curve. The "all-or-none" framework can be

Table 4: Who knows what: all-or-none

Who knows what	(p_i, q_i)	(p_j, q_j) $i \neq j$	θ	$\sum q_i$	$\sum p_i q_i$
Supplier (if all)			$\surd$	$\surd$	$\surd$
Buyer i (if all)	$\surd$				
Supplier (if none)			$\surd$		
Buyer i (if none)	$\surd$				

easily embedded in an iterative algorithm to improve supplier profits, where unprofitable orders are dropped sequentially; see Section 5.3

5.2.1 All-or-None Framework:

In this framework, each buyer makes his orders as a single price-quantity pair, and the supplier has to either accept or reject all according to his supply curve. The buyers do not want to reveal their orders before the supplier's decision is made.

Let (p_i, q_i) be the price-quantity pair of buyer i. Let the supply curve of the supplier be $q = p + \theta$. Without knowing the buyers' order, the supplier needs to know whether the total revenue, $\sum_i p_i q_i$, will be at least as large as his supply curve requires; if not he will choose a revenue of zero. Hence the problem is to compute this predicate without revealing any additional information about the supply curve or about the price-quantity pairs. If the supplier were to agree to the deal, then his threshold from his supply curve is $(\sum_i q_i - \theta)(\sum_i q_i)$. Because his revenue from the deal would be $\sum_i p_i q_i$, he will accept only if that revenue exceeds the above-mentioned threshold. Hence, the problem is defined as computing the predicate $\sum_i p_i q_i \geq (\sum_i q_i - \theta)(\sum_i q_i)$ without revealing any (p_i, q_i) or θ. The following protocol allows the supplier to make her decision without revealing her supply curve, and without revealing to her any of the price-quantity pairs of the buyers.

5.2.2 Secure All-or-None Protocol:

In this section we present a circuit for determining Non-Discriminatory Pricing (see Figure 5) and thus this quantity can be evaluated in a secure manner with the methods outlined in Theorem 1.

Constructing the Circuit

The circuit can be trivially constructed.

Input: Each retailer R_i has a quantity q_i and a price p_i (note that their actual inputs into the circuit will be q_i and $p_i q_i$ and the supplier has capacity θ.
Output: The supplier learns q_i and $p_i q_i$ for all retailers iff $\sum_{i=1}^{N}(p_i q_i) \geq (\sum_{i=1}^{N}(q_i))(\sum_{i=1}^{N}(q_i) - \theta)$.
Description:

(1) $D \leftarrow \sum_{i=1}^{N}(q_i)$
(2) $RHS \leftarrow (D) \times (D - \theta)$
(3) $LHS \leftarrow \sum_{i=1}^{N}(p_i q_i)$
(4) $p \leftarrow LHS \leq RHS$
(5) for $j \leftarrow 1$ to N do
(6) $\quad q_j = q_j \times p$

Note: that in step (4) and (6) we treat a boolean value as an integer that is $\{0, 1\}$.

Figure 5: Secure-All-or-None-Protocol

Corollary 4 *The protocol in Figure 5 can be evaluated securely, as in Theorem 1.*

Complexity Analysis

We assume that the variables $\sum_{i=1}^{N}(p_i q_i)$ and θ can be represented by m bits. Furthermore, for simplicity we say that the individual p_i's and q_i's are also represented by m bits. It is not difficult to show that the variables D and LHS and also can be represented with m bits and RHS needs m^2 bits. Steps (1), (3), and (4) requires $O(m \log N)$ gates, and step (2) requires $O(m^2)$ gates. Steps (5) and (6) require $O(Nm)$ gates, and thus the circuit has size $O(m^2 + Nm)$. There are $O(Nm)$ input and output wires in the circuit. The depth of the circuit is $O(\log N \log M)$.

Inverse-Optimization

Table 4 summarizes who knows what after the protocol completes. From what is in the table it is to state that the suppliers nor the retailers can compute values exactly. If the retailers compute $\sum q_i$ they can get a bound on the value θ. To compute this value exactly, the adversary would have to control all retailers, but an adversary that can guess this value can place a bound as accurate as his guess.

In case the seller accepts the offers, there is the practical issue of how to get the merchandise to the buyers. One way to do this is by using a proxy to whom the buyers reveal their q_i's after the protocol, and who will handle shipping. The proxy sums the q_i's up and sends $\sum q_i$ to the supplier who sends the quantities to the proxy who acts as a "switchboard" sending q_i items to buyer i. We present techniques for preventing a buyer from changing his bid in Section 5.3.

An Extension to the All-or-None Framework

One problem with the all-or-none framework is that it may take several iterations to find an agreement between the suppliers and the retailers. A simple modification to this protocol allows the parties to avoid this iterative process: Instead of making a series of sequential bids, the parties could submit several bids at once in a defined order from highest desirability to lowest desirability (both the supplier and the retailers could submit such bids) and then all of the bids could be processed in parallel with the highest desirability market clearing transaction being chosen. If no, such bid was found then another iteration would be done. This version is a straight-forward modification to the circuit listed in Figure 3.

5.3 Tying the Buyer's hands

In the above protocols, there is a pragmatic issue of preventing a buyer from changing her bid after engaging in the protocol. If it is okay to run another SMC protocol for each of the buyers, then we can simply engage in a protocol to prevent to make sure that a buyer's bid matches her original bid. This is discussed in more detail in Section 3.3. In this section, we discuss techniques for preventing such a change without engaging in such a protocol.

A traditional model for such commitments is that the parties submit a commitment of their bids along with a proof that this matches their bid. For more information about such techniques the reader should look at (Goldreich 2001, Goldreich 2004). The problem with such techniques is that they are very expensive. We argue here that it is easy for the parties involved to check if things are consistent with the results of the circuit, and if there is a discrepancy then in either of the untrusted third parties architectures the results can easily be checked to see who is lying. Thus an adversary that is lying will be detected and subsequently punished, and thus there is little incentive to lie about ones bids. We now mention how to determine if the bids are consistent in both of the previous protocols.

Non-discriminatory Pricing: The retailers that are not active at the end of the protocol can verify that the value $\hat{p}$ is larger than their individual price quote, and those that are active can verify that that it is no larger. The supplier can verify whether or not the sum of quantities from the active retailers fits her supply curve.

All or None-Framework: The supplier can verify if the the total revenue from the proxy matches the value $\sum p_i q_i$ that it learned from the protocol.

6 Concluding Remarks

Secure supply-chain collaboration (SSCC) involves the simultaneous application of the principles of secure multi-party computation (SMC) and mechanism design (MD) to decision-making in supply-chain management (SCM). More specifically, as described above, SSCC can be viewed as preserving the privacy (P) of the agents' information in the framework of distributed algorithmic mechanism design (DAMD). Hence, a new acronym: DAMDP. Although DAMDP has already been examined in the special case of online auctions (Naor et al. 1999, Jakobsson and Juels 2000, Elkind and Lipmaa 2003, De Decker et al. 2001), we believe ours is the first application of DAMDP to supply-chain management.

We believe that the application of DAMDP to SCM provides extremely rich opportunities for computer science, economics, and supply-chain management. What we have described here only scratches the surface. For example, with the proviso that managerially meaningful trust and architectural issues can be formulated for the case of multiple suppliers – a nontrivial issue – the privacy-preserving protocol part of our work extends to multiple sellers. From an application perspective, for example, although most real-world scenarios are considerably more complex than those examined above, the concept of privacy-preservation is the same, and the principles represented in the protocols and mechanisms described are identical. Indeed, the authors of this paper are using these principles in a planned software tool for use by electronics manufacturers. The tool is to be used in a negotiation-type process (led by the manufacturer for a group of component suppliers) to determine the minimum price for a kit of parts without revealing any individual part prices. Some of the techniques described above will be used in this application.

Nonetheless, much additional applications-oriented work remains to be done. For example, although the protocols themselves may be secure from a DAMDP perspective, systems must be designed to facilitate their implementation in a secure manner; i.e., to make it impossible, or at least very difficult, for participants to make accurate inferences about their partners private information (i.e., inverse optimization). Systems must be also be designed to overcome, or at least minimize, the adverse impact of collusion, dishonesty, etc.

Important theoretical/conceptual work in all facets of the problem also remains to be done; for example, additional techniques must be created, and architectures designed, to facilitate the development of the myriads of promising applications.

Acknowledgment of Support

Portions of this work were supported by Grants IIS-0325345, IIS-0219560, IIS-0312357, and IIS-0242421 from the National Science Foundation, Contract N00014-02-1-0364 from the Office of Naval Research, by sponsors of the Center for Education and Research in Information Assurance and Security, and by Purdue Discovery Park's e-enterprise Center. The authors are grateful to Umut Topkara for his useful comments.

References

Atallah, M.J., F. Kerschbaum, W. Du. 2002. Secure and Private Sequence Comparisons. *Proc. 2d. ACM Workshop on Privacy in Electronic Society*, Washington, DC, October 2003.

Atallah, M.J., H.G. Elmongui, V. Deshpande, L.B. Schwarz. 2003. Secure Supply-Chain Protocols. *Proc. 2003 IEEE Conference on Electronic Commerce (CEC)*, Newport Beach, California, June 2003, pp. 293–302.

A.V. Aho, J.E. Hopcrof, and J.D. Ullman. 1974. The Design and Analysis of Computer Algorithms. Addison-Wesley.

Ahuja, R. K. and J. B. Orlin. 2001. Inverse Optimization. *Operations Research*, 49(5), 771–783.

H. Alt. 1988. "Comparing the combinational complexities of arithmetic functions", *Journal of the ACM*. Volume = 35(2), 447–460.

M. Ajtai, J. Komlos, and E. Szemeredi. 1983. "An O((n log n)) sorting network", In *Proceedings of the 15th ACM Symposium on Theory of Computing*, pages 1–9.

Aviv, Y. 2002. Gaining Benefits from Joint Forecasting and Replenishment Processes: The Case of Auto-Correlated Demand. *Manufacturing & Service Operations Management*, 4(1), 55–74.

Aviv, Y. 2001. The Effect of Collaborative Forecasting on Supply Chain Performance. *Management Science* 47(10) 1–18.

K. E. Batcher. 1968. "Sorting Networks and their Applications", *Proc. AFIPS Spring Joint Computer Conference* 32:307-314.

P. Beame and S. Cook and H. Hoover. 1984. "Log depth circuits for division and related problems", *Annual IEEE Symposium on Foundations of Computer Science*, pages 1–6.

Felix Brandt. Fully private auctions in a constant number of rounds. In *Financial Cryptography — Seventh International Conference*, 2003.

Cachin, C. 1999. Efficient Private Bidding and Auctions with an Oblivious Third Party. *Proc. of the 6th ACM Conference on Computer and Communications Security*, 120–127.

Cachon, G. P. (2004). Supply Chain Coordination with Contracts. In Kok, A. G. de, and S. C. Graves, editors, *Handbooks in Operations Research and Management Science: Supply-Chain Management*. North-Holland, Amsterdam, The Netherlands. Forthcoming.

Cachon, G.P., M. Fisher, 2000. Supply Chain Inventory Management and the Value of Shared Information. *Management Science* 46(8) 1032–1050.

R. Canetti. Security and composition of multiparty cryptographic protocols. *Journal of Cryptology*, Vol. 13, Iss. 1, 143–202, 2000.

R. Canetti, Y. Lindell, R. Ostrovsky, and A. Sahai. Universally composable two-party and multi-party secure computation. *Proceedings of the thirty-fourth annual ACM symposium on Theory of computing*, 2002, 494–503.

Canetti, R., Y. Ishai, R. Kumar, M.K. Reiter, R. Rubinfeld, R.N. Wright. 2001. Selective Private Function Evaluation with Applications to Private Statistics (extended abstract). *Proceedings of the Twentieth ACM Symposium on Principles of Distributed Computing (PODC), 1999*, 293–304.

Chang, Y.C., V.J. Lu. 2001. Oblivious Polynomial Evaluation and Oblivious Neural Learning. *Springer Lecture Notes in Computer Science 2248, Proceedings of ASIACRYPT*, Gold Coast, Australia, 2001, 369–384.

Chen, F., Z. Drezner, J.K. Ryan, D. Simchi-Levi, 2000. Quantifying the Bullwhip Effect in a Simple Supply Chain: The Impact of Forecasting, Lead Times, and Information. *Management Science Volume* 46(3) 436–443.

Clifton, C., I.V. Iyer, R. Cho, W. Jiang, M. Kantarcioglu, J. Vaidya. 2003. An Approach to Identifying Beneficial Collaboration Securely in Decentralized Logistics Systems. Working Draft, Purdue University.

Conitzer, V., T. Sandholm. 2002. Complexity of Mechanism Design. *Proceedings of the 18th Conference on Uncertainty in Artificial Intelligence (UAI)*, Edmonton, Canada, Morgan

Kaufmann, 103–110.

De Decker, B., G. Neven, F. Piessens, E. Van Hoeymissen. 2001. "Second Price Auctions, A Case Study of Secure Distributed Computing," *Proceedings Third IFIP WG 6.1 International Working Conference on Distributed Applications and Interoperable Systems,* Krakow, Poland, 217–228.

Deshpande, V., L. Schwarz. 2005. Optimal Capacity Allocation in Decentralized Supply Chains. Working Paper, Krannert School of Management, Purdue University, West Lafayette, Indiana, USA.

Elkind, E., H. Lipmaa. Interleaving Cryptography and Mechanism Design: The Case of Online Auctions. 2003. *Technical Report 2003/021, International Association for Cryptologic Research.*

Federgruen, A. 1993. Centralized Planning Models for Multi-Echelon Inventory Systems Under Uncertainty. Graves, S.C. et al, eds. *Handbooks in OR and MS.* Chapter 3. North Holland.

Feigenbaum, F., N. Nisan, V. Ramachandran, R. Sami, S. Shenker. 2002. "Agents' Privacy in Distributed Algorithmic Mechanisms," *Workshop on Economics and Information Security,* Berkeley, CA.

Feigenbaum, J., S. Shenker. 2002. "Distributed Algorithmic Mechanism Design: Recent Results and Future Directions," *Proceedings of the 6th International Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications,* ACM Press, New York, 1–13.

Fischlin, M. 2001. A Cost-Effective Pay-Per-Multiplication Comparison Method for Millionaires. RSA Security 2001 Cryptographer's Track. *Lecture Notes in Computer Science* 2020, 457–471.

M. Franklin, M. Reiter. 1996. The Design and Implementation of a Secure Auction Service. *IEEE Transactions on Software Engineering*, Vol.22, No. 5.

Fudenberg, D., and J. Tirole, Game Theory, MIT Press, 2000.

Goldreich, O. 1998. Secure Multi-party Computation (working draft). Available from http://www.wisdom.weizmann.ac.il/home/oded/public_html/foc.html

O. Goldreich. *Foundations of Cryptography: Volume I Basic Tools.* Cambridge University Press, 2001.

O. Goldreich. *Foundations of Cryptography: Volume II Basic Application*. Cambridge University Press, 2004.

Goldreich, O., S. Micali, A. Wigderson. 1987. How to Play any Mental Game. *Proceedings of the 19th annual ACM symposium on Theory of computing, 1987*, 218–229.

Harris, M., and Raviv, A. (1981), "Allocation Mechanisms and the Design of Auctions", *Econometrica*, 49(6), p1477-1499.

Iyer, A.V., J. Ye. 2000. Assessing the Value of Information Sharing in a Promotional Retail Environment. *Manufacturing & Service Operations Management* 2) 128–143.

Lee, H.L., K.C. So, C.S. Tang, 2000. The Value of Information Sharing in a Two-Level Supply Chain. *Management Science* 46(5) 626–643.

Lee, H., V. Padmanabhan, S. Whang, 1997. Information Distortion in a Supply Chain. *Management Science* 43(4) 546–558.

Maskin, E. and J. Riley (1989), "Optimal Multi-Unit Auctions", *The Economics of Missing Markets, Information, and Games*, Oxford University Press, New York.

Muckstadt, J., R.O. Roundy. 1993. Analysis of Multistage Production Systems. Graves, S.C. et al, eds. *Handbooks in OR and MS*. Chapter 2. North Holland.

Naccache, D., Stern, 1998. J. New Cryptosystem based on Higher Residues. *Proceedings of the ACM Conference on Computer and Communications Security* 5 59–66.

Naor, M., B. Pinkas. 1999. Oblivious Transfer and Polynomial Evaluation (Extended Abstract). *Proceedings of the 31th ACM Symposium on Theory of Computing, Atanta, GA, USA, May 1999*, 245–254.

Naor, M., B. Pinkas, R. Sumner. 1999. "Privacy preserving auctions and mechanism design." *Proceedings 1st ACM Conf. on Electronic Commerce,* Denver, CO, 129–139.

Ofman, Y. On the algorithmic complexity of discrete functions. Sov. Phys. Dokl. 7 (1963), 589- 591.

Okamoto, T., Uchiyama, S. 1998. A New Public-Key Cryptosystem as Secure as Factoring. Eurocrypt'98 *Lecture Notes in Computer Science* 1403, 308–318.

J.H. Reif and S. R. Tate. 1989. "Optimal size integer division circuits" *Proceedings of the twenty-first annual ACM symposium on Theory of computing table of contents* pages 264-273.

Table 5: Who knows what: pick and choose

Who knows what	(p_i, q_i)	(p_j, q_j) $i \neq j$	K
Supplier			$\surd$
Buyer i	$\surd$		$\surd$

P. Rogaway, 1991. The Round Complexity of Secure Protocols. Ph.D. thesis, MIT.

Sandholm, T., Suri, S. 2001. Market Clearability. *International Joint Conference on Artificial Intelligence (IJCAI), Seattle, WA, 2001.*

Sandholm, T., Suri, S. Optimal clearing of supply/demand curves. 2002. *AAAI-02 workshop on Agent-Based Technologies for B2B Electronic Commerce, Edmonton, Canada, 2002.*

Schneier, B. 1995. *Applied Cryptography.* John Wiley & Sons.

A. Schoehage and V. Strassen. 1971. "Schnelle Multiplikation Grosser Zahlen", *Computing*, volume = 7, pages 281–292.

Song, J.S., P.H. Zipkin. 1996. Inventory Control with Information about Supply Conditions. *Management Science* 42(10) 1409–1419.

Yao, A. 1982. Protocols for secure computations. *Proceedings of the 23rd Annual IEEE Symposium on Foundations of Computer Science.*

A.C Yao, 1986. How to generate and exchange secrets. *Proceedings of the 27th Annual IEEE Symposium on Foundations of Computer Science*, pages 162–167.

A Pick and Choose Details

In this section we outline the details of the Pick and Choose details from Section 4.4 We have n buyers, labeled arbitrarily 0 through $n-1$. All arithmetic of indices representing buyers is done $\bmod n$. In what follows V is a number larger than any price p_{a_b}. This implies that nV is an upper bound on the total revenue. As in (Sandholm and Suri 2001), we let $A(i, v)$ denote the smallest number of units that can be sold to those buyers whose names are in the set $\{0, 2, \ldots, i\}$ with total revenue exactly v. The dynamic program used is as in (Sandholm and Suri 2001).

Inverse-Optimization

Input: Each retailer R_i has a up to j price-quantity pairs $(p_1, q_1), \ldots (p_j, q_j)$ (as not to make the notation confusing we do not have an additional index for i, but it is clear from our protocols which retailers pairs are being used) and the supplier has capacity K.
Output: The supplier learns that quantity of item to supply to each retailer to maximize revenue, given her capacity constraints.
Description:

(1) For retailer 0, values are created such that $A(0, p_t) = q_t$ and $A(0, v) = \infty$ for all other values v including for values in $[-V, 0)$.
(2) for $i \leftarrow 1$ to $N - 1$ do
(3) for $v \leftarrow 1$ to NV
(4) $A(i, v) \leftarrow \infty$
(5) /* Buyer i has bids $(p_1, q_1), \ldots, (p_j, q_j)$ */
(6) for $t \leftarrow 1$ to j do
(7) $\alpha \leftarrow A(i, v)$
(8) $\beta \leftarrow A(i - 1, v)$
(9) $\gamma \leftarrow q_t + A(i - 1, v - p_t)$
(10) $A(i, v) \leftarrow \min\{\alpha, \beta, \gamma\}$

Figure 6: Protocol for Pick and Choose

Table 5 summarizes who knows what after the protocol completes.

In this protocol, the supplier is not involved in the computation steps. He only declares his K at the beginning, then the buyers cooperate to get the result. This protocol preserves the privacy of all the price-quantity pairs. No p_i or q_i of any buyer is revealed to any other buyer or to the supplier.

B Sorting Circuits

Consider the problem of sorting n numbers with m bits each. In this section we describe a Batcher's sorting circuit (Batcher 1968).

Input: Two sorted lists $a_0, \ldots, a_n - 1$ and $b_0, \ldots, b_n - 1$. For simplicity we will assume n is a power of 2.
Output: One sorted list of $2n$ items.

1. If the lists consist of one item the lists are merged in the standard manner using 1 comparison.
2. Otherwise the lists are split into two parts, $A_e = a_0, a_2, \ldots, a_{n-2}$, $B_e = b_0, b_2, \ldots, b_{n-2}$, $A_o = a_1, a_3, \ldots, a_{n-1}$, and $B_o = b_1, b_3, \ldots, b_{n-1}$. The circuit recursively merges A_e and B_o, as well as A_o and B_e obtaining $c_0, \ldots, c_{n-1}$ and $d_0, \ldots, d_{n-1}$ respectively.
3. The circuit forms the list $c_0, d_0, c_1, d_1, \ldots, c_{N-1}, d_{N-1}$ and then compares each pair c_i and d_i swapping them if necessary.

Figure 7: Protocol for Sorting