Trustworthy Auctions for Grid-style Economies

Kris Bubendorfer, Ian Welch and Blayne Chard
School of Mathematics, Statistics and Computer Science
Victoria University of Wellington, New Zealand,
Email: {kris.bubendorfer, Ian.Welch}@mcs.vuw.ac.nz

Abstract

Commercialisation or globalisation of large scale Grids requires the provision of mechanisms to share the wide pool of Grid brokered resources such as computers, software, licences and peripherals amongst many users and organisations. Quickly and efficiently servicing resource requests is critical to the efficiency of such Grid based utility computing and communication providers.

The CORA architecture is a market based resource reservation system that utilises a trustworthy Vickrey auction to make combinatorial allocations of resources. The primary advantage of such a scheme is that a trusted auctioneer is no longer necessary, and any system entity can safely host a trustworthy auction. This approach results in more flexibility in the design of large economic systems, with the potential for wide distribution of load amongst many auctioneers. In addition, only the winners of the auction and the prices they pay are revealed while all other bid values are kept secret. This paper also provides performance results for our implementation, that identify the constraints within which a practical trustworthy auction scheme can be implemented in a Grid-style Economy.

1 Introduction

Efficient negotiation for and allocation of resources plays an increasingly important role in the performance of large scale computing systems. Commercialisation of Grid systems requires the provision of mechanisms to share a wide pool of Grid brokered resources, such as computers, software, and peripherals amongst many users and organisations. The Application Service Provider model [11, 7] is one such Grid commercialisation model.

The Internet Virtual Organisation (iVO) [9], can be extended to utilise, on demand, resources leased dynamically from Utility Computing Providers (UCP) [13]. A further extension of the UCP model to include the leasing of communication services, giving Utility Computing Communications Providers (UC^{2}P). A UC^{2}P infrastructure could be heavily based on developments in Grid computing.

Unlike the traditional Grid model where fixed resources are acquired in advance of execution, a UC^{2}P infrastructure requires smaller, more dynamic negotiations that support mobile devices and the provision of on demand services. Auctions are touted as an efficient solution to the challenge of distributed resource allocation in both economic [2, 4, 5] and noneconomic [16] resource allocation systems. In the CORA [3, 1] project, we are investigating an economic management model based upon auctions for resource reservation. Many of the lessons learnt in developing CORA are applicable to the wider Grid community.

Central to user acceptance of a market based resource reservation system such as CORA is belief in its security. The security requirements for CORA go beyond the usual need for confidentiality, availability, integrity and access control by including a requirement for trustworthy auctions. A trustworthy auction process is central to the proper allocation of resources. For example, what happens if the auctioneer chooses the winner because the auctioneer's integrity has been subverted, or the auctioneer uses bids to collect information for a competing bidder. The simplest but least flexible approach is to simply trust nominated auctioneers, but this restricts who can host an auction thereby limiting the scalability of the system. Instead we have investigated the application of algorithms for trustworthy auctions that preserve privacy and to detect malfeasance.

This paper provides an overview of trust in electronic auctions, an initial implementation of a trustworthy auction scheme for CORA, and subsequent performance measurements of the implementation.

2 Trust in Auctions

Until recently there was little recourse but to design auction based allocation systems with an auctioneer as a trusted service. However, this approach tends to result in centralised designs and lacks openness, transparency and verifiability. Recently there have been significant research efforts to determine if an auctioneer is acting in a trustworthy manner or to even remove the need for the auctioneer to be trusted at all. Essentially trust in an auction can be estab-

Proceedings of the Sixth IEEE International Symposium on Cluster Computing and the Grid (CCGRID'06)
0-7695-2585-7/06 $20.00 © 2006 IEEE

lished in one of five ways:

1. Pre-existing trust (i.e., system components)
2. Reputation services (i.e., earnt trust)
3. Bid-encryption schemes (i.e., procedural trust)
4. Threshold schemes (i.e., distributed trust)
5. Threshold bid-encryption schemes (i.e. enforced trust)

Reputation services rate the performance of an auctioneer based on reports from previous auctions [6]. Trust can be delegated [12] to bootstrap new auctioneers into the system. However, the problem of initial trust remains, as does the problem of verification – how do participants verify the auction process without revealing potentially sensitive valuation information? For example, zero knowledge proofs can be used to allow the auctioneer to prove that it acted correctly without revealing bid values [14].

Bid encryption schemes dispense with the need to initially trust an auctioneer, indeed, the issue of whether an auctioneer is trustworthy is no longer relevant. Here, cryptographic protocols are used that make it impossible (or excessively expensive) for an auctioneer to learn anything useful from, or to manipulate the outcome of, an auction. For example, Noar [17] proposed a general method for computing any auction protocol securely including combinatorial auctions using a method known as garbled circuits. The main criticism of schemes based upon garbled circuits is that the circuits have be very complex and the circuit transfer between the circuit constructor and auctioneer implies a significant communication overhead.

Threshold schemes are based upon secret sharing [19]. These schemes allow trust to be placed in a set of auctioneers rather than a single auctioneer. As long as a certain number (a quorum) of auctioneers are not corrupt and execute the auction protocol correctly, a minority of malicious auctioneers cannot subvert the protocol and manipulate the auction. For example, the bids could be encrypted using public key cryptography and require cooperation of multiple auctioneers to decrypt the bids so the winner can be computed [10]. The requirement for cooperation of a minimum number of correct auctioneers prevents the auction being manipulated, however it does not prevent auctioneers learning bid values that are decrypted during execution of the protocol.

Threshold bid-encryption schemes provide better privacy guarantees than just threshold trust schemes while moderating overhead. The work in this paper is based on the Secure Generalised Vickery Auction protocol (SGVA) [21]. The SGVA protocol uses homomorphic encryption to hide bid values and distributed decryption to prevent a malicious auctioneer revealing bids or manipulating the outcome of the auction. We describe how we have used this protocol in the next section.

3 Building a Trustworthy Auctioneer

We have implemented the Secure Generalised Vickery Auction (SGVA) [21] threshold bid-encrypted protocol. SGVA is a secure version of the Generalised Vickery Auction Protocol (GVA) [15]. SGVA hides each bidder's evaluation values for different combinations of goods offered by sale, while allowing calculation of the optimal allocation of goods and the prices. A homomorphic cryptographic scheme is used to encrypt the bids while allowing their use in winner determination and price computation. Distributed decryption is used to prevent malicious auctioneers learning any bids [18]. This section provides a brief overview of representation of bid values and the protocol itself.

3.1 Representation of Values

The SGVA protocol represents values as vectors composed of elements encrypted using a homomorphic cryptographic scheme, and defines operations that allow addition and comparison without revealing the values themselves. Each element in the vector is either the encryption of the value one or a common public value chosen by the auctioneers. The value encoded in the vector is equal to the number of encrypted common public values. Note that each element may represent a unit larger than 1. For example, an auction with a vector of size 10 could have bids \$1 to \$10 or \$10 to \$100 using a \$1 or \$10 unit value respectively. A bidder's *weight collection* is the set of all its vectors for each possible combination of goods in the auction.

Addition of two vectors depends upon the use of a homomorphic cryptographic scheme that allows addition of encrypted values without needing to decrypt the values. Our implementation uses the Elgamal public key encryption scheme [8] and allows two vectors to be added together by componentwise multiplication of their vector elements. Besides adding two vectors representing bid values we also need to add constants. An efficient approach is to left shift vector components as many times as the value of the constant. Addition of a random constant is used to hide individual bid values while allowing auctioneers to calculate the maximum bid value from the collection of bids.

Winner computation requires comparing bid values, however comparison of two vectors cannot be done directly because, due to randomizability, two vectors representing the same value will contain different component values. Therefore the SGVA protocol performs comparison as a two-step process. First, all the values to be compared are added together. Second, the result is decrypted one element at a time from left to right until we find an element equal to one. The position of this element (or properly the element one to its left) is the greatest price from the collection of vectors. This reveals the maximum value without revealing individual values.

As described above, auctioneers must be able to decrypt elements. However, this capability must be controlled to prevent a malicious auctioneer simply decrypting a bidder's bid values and leaking them. This is achieved through the use of a distributed decryption protocol that ensures that multiple key shares are required to decrypt a vector element. By splitting the key shares amongst a group of auctioneers we ensure that a single malicious auctioneer cannot act independently.

3.2 The Protocol

The SGVA protocol has been designed to ensure that only the result of auction is made public while bids are kept secret even in the presence of a malicious auctioneer. There are three phases to the protocol: preparation where the parameters of the auction are initialised; bidding where encrypted bids are generated and combined together; and, bid opening where the winner is determined.

In the *preparation* phase, the auctioneers calculate all possible combinations of goods, generates the public and private keys for the chosen encryption scheme and publishes the range of possible evaluation values. Finally, the auctioneers distribute encrypted vectors representing a bid of zero for each combination of goods to each bidder.

In the *bidding* phase, bidders generate their encrypted bids using the vectors distributed to them by the auctioneers. Each bidder creates a weight collection by assigning an evaluation value to each possible combination of goods. In this context, the evaluation value can be thought of as the maximum price they are willing to pay. At the end of this phase, the weight collections are received by the auctioneers and combined by the auctioneers into a directed graph representing the auction. Each node of the graph represents a possible combination of goods and the arcs between the nodes have weights representing the combined evaluation values assigned by bidders.

The final phase is *bid opening*. In SGVA the winner(s) are determined by finding the longest path through the directed graph. Every possible path needs to be evaluated and this requires adding and comparing the weights on the arcs of the graph. As described in the previous section, the addition can be performed without revealing information about the individual bids but comparison requires partial decryption. The use of distributed decryption prevents a minority of malicious auctioneers being able to perform arbitrary decryptions. Furthermore, to increase bid privacy by obscuring what is decrypted, the auctioneers cooperate in adding a random constant to the weights before computation of the longest path takes place. This masks the weights without interfering with comparison or other operations upon the bids. Note that this addition is performed in a distributed manner to prevent an individual auctioneer discovering the complete random value.

4 Evaluation

In this section, we explore the performance of the SGVA algorithm with respect to the cryptographic costs of bid generation and vector manipulation. In particular, we set out to determine the sensitivity of the algorithm to the four main variables: the number of bidders, the encryption key size, the number of items, and the permitted bid range. The selection of these values has an impact on the tractability of the SGVA algorithm. A secondary goal of these experiments was to determine where we need to eliminate bottlenecks from the algorithm and the implementation.

Each of the performance results (environment: 128MB JVM, Java 1.5, 3GHz Pentium 4 with a 1Gbit Ethernet) in sections 4.1– 4.4 involve changing a single variable. Other than the variable under test, the parameters selected for the performance measurements were: ten bidders, three items, 64bit key size, 10bit vector size and a single auctioneer.

4.1 The Number of Bidders

The number of bidders in an auction is one variable that is outside the control of the auctioneer. However, it is worth noting that as a minimum, a Vickrey auction needs at least four bidders to provide an optimal allocation[20].

Figure 1. Number of bidders has a linear effect on winner determination time.

As shown in figure 1, as the number of bidders increases the winner determination time experiences a linear growth. By adding one bidder, the time taken to combine the bids and find the winner increases by one iteration per bidder added. In practice, this linear growth ends abruptly when the auctioneer runs out of heap memory (at approximately 6000 bidders for a 128MB JVM). This is due to the storage requirements of the weight vectors, a tradeoff of space against granularity. In figure 1 the spikes correspond to garbage collection events.

4.2 The Encryption Key Size

The size of the key determines how secure the auction is, however we cannot simply increase the key without limit. Each increase in the key size also increases the winner determination time, the time required to compute the graph representing the auction, and the time required to combine the bids.

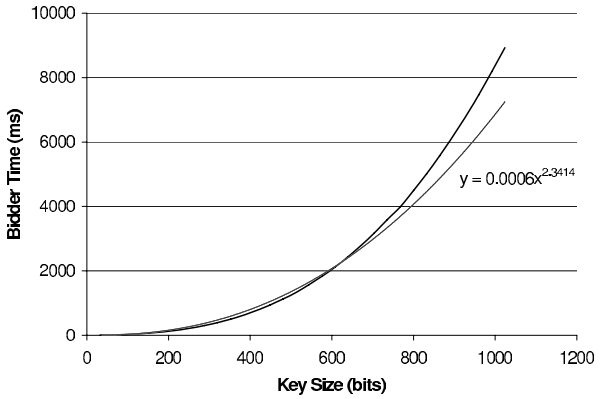

Figure 2. The effect of increasing key size on bid creation time

As figure 2 shows, increasing the key size increases the time taken for a bidder to generate its bid. Encryption keys of 96 bits are currently considered acceptable for general system security, while for our purposes we can use much shorter keys as the bid data is of limited duration. Each bidder makes its own bid collection, so each bidder experiences roughly the same computational cost to make a bid. We also measured the effect of increasing the key size on the time taken for the auctioneer to perform its winner computation. The result is similar to that in figure 2, in particular $numBidders \times y$. This indicates that dominant cost in this part of the protocol is the handling of the encryption keys during the merge, comparison and evaluation of the bids.

4.3 Weight Collection

Increasing the size of weight collections allows for finer grained bids to be made. With weight collections the value of each unit varies between auctions. An auction with a weight collection of size 10 could have bids \$1 to \$10 or \$10 to \$100 using a \$1 or \$10 unit value respectively. As described in section 4.1 increasing the size of the weight collections increases the storage requirements in the auctioneer, but otherwise has a benign impact on the winner determination time. An increment in the size of the weight collection increments the total amount of data required to be stored by each bidder by $2 \times$ Key size, and then on the auctioneer that total is multiplied by the number of bidders. Increasing the size of the weight collection is a linear cost to the bidder. For the auctioneer, the cost is also linear (shown in Figure 3), as the winner determination algorithm only has a single iteration added to all of the bid comparisons.

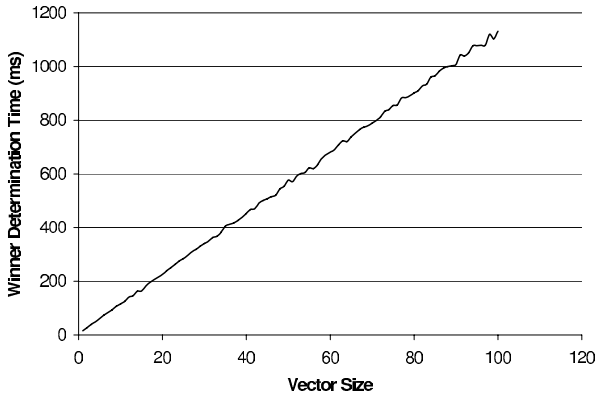

Figure 3. The effect of increasing weight collection size on winner determination time

4.4 Combinations of Goods

The number of available goods for sale in a combinatorial auction is the major performance bottleneck. An auction with two items means there are three possible combinations that need to be computed and evaluated, adding a third item increases the number of combinations to seven. This growth of combinations is $f_n = 2^n - 1$, and solving such a combinatorial problems is always *NP*-hard. We have implemented the pure solution to the GVA, without any of the approximations such as bundling. As Figure 4 illustrates, the algorithm is exponential in computation time. The significance of this result is that we can compute the combinatorial allocation of up to five goods within a reasonable time (1.3 seconds for 4 goods or 7.3 seconds for 5 goods). This gives us a working baseline on which to apply approximations to the CAP, that we will incorporate into future iterations of our system.

Figure 4. The effect of increasing the number of resources on winner determination time

5 Conclusions

We have extended the CORA resource reservation architecture via the inclusion of the Secure Vickrey Auction protocol. The use of SGVA ensures that only the winners of the auction and the prices they pay are revealed while all other bid values are kept secret. The major implication of this change to CORA is that the auctioneer no longer needs to be a single privileged system component, but rather an adhoc group of auctioneers who do not need to be individually trusted. This approach provides the potential for a wide distribution of load amongst many auctioneers.

Furthermore we have evaluated the implementation in a practical setting to determine the sensitivity of its performance with respect to initial parameters such as the number of bidders, goods and cryptographic key length. This performance evaluation has identified that cryptographic performance is a significant bottleneck in bid generation and winner determination. Future work will evaluate this with respect to distributed decryption.

Our results have illustrated the problem of the combinatorial explosion due to the number of goods, and reinforced the need to utilise optimisation techniques in combinatorial auctions. We have identified some of the constraints within which a practical trustworthy auction scheme can be implemented in a Grid-style Economy.

References

[1] K. Bubendorfer. Improving Resource Utilisation in Market Oriented Grid Management and Scheduling. In *in proceedings of the 4th Australasian Symposium on Grid Computing and e-Research (AusGrid 2006)*, volume 54, pages 25–32, Hobart, Tasmania, Australia, January 2006.

[2] K. Bubendorfer and J. H. Hine. Resource Discovery and Negotiation in the NOMAD System. In *in Proceedings of ACSC2005, The Twenty Eigth Australasian Computer Science Conference*, volume 27, pages 297–305, Newcastle, NSW, Australia, January 2005.

[3] K. Bubendorfer, P. Komisarczuk, K. Chard, and A. Desai. Fine Grained Resource Reservation and Management in Grid Economies. In *Proceedings of the 2005 International Conference on Grid Computing and Applications*, pages 31–38, Las Vegas, Nevada, USA., June 2005.

[4] R. Buyya, D. Abramson, J. Giddy, and H. Stockinger. Economic models for resource management and scheduling in Grid computing. *Concurrency and Computation: Practice and Experience*, 14:1507–1542, 2002.

[5] C. H. Chien, P. H. M. Chang, and V. W. Soo. Market-Oriented Multiple Resource Scheduling in Grid Computing Environments. In *Proceedings of Advanced Information Networking and Applications (AINA'05)*, volume 1, pages 867–872, Taipei,, March 2005.

[6] E. Damiani, D. C. di Vimercati, S. Paraboschi, P. Samarati, and F. Violante. A reputation-based approach for choosing reliable resources in peer-to-peer networks. In *Proceedings of the 9th ACM conference on Computer and communications security*, pages 207–216, New York, NY, USA, 2002. ACM Press.

[7] T. Dimitrakos, D. M. Randal, F. Yuan, M. Gaeta, G. Laria, P. Ritrovato, B. Serhan, S. Wesner, and K. Wulf. An Emerging Architecture Enabling Grid Based Application Service Provision. In *Seventh International Enterprise Distributed Object Computing Conference (EDOC'03)*, pages 240–251, Brisbane, Queensland, Australia, September 2003.

[8] T. Elgamal. A public key cryptosystem and signature scheme based on discrete logarithms. *IEEE Transactions on Information Theory*, 31(4):69–472, July 1985.

[9] I. Foster and C. Kesselman. *"The Grid: Blueprint for a New Computing Infrastructure"*. Morgan and Kaufmann, 1999.

[10] M. Franklin and M. Reiter. The Design and Implementation of a Secure Auction Service. In *Proceedings of the IEEE Symposium on Security and Privacy*, pages 2–14, Oakland, California, USA., 1995. IEEE Computer Society Press.

[11] S. Graupner, V. Kotov, A. Andrzejak, and H. Trinks. Service-Centric Globally Distributed Computing. *IEEE Internet Computing*, 7(4):36–43, 2003.

[12] L. Kagal, S. Cost, H. Chen, T. Finin, and Y. Peng. An Infrastructure for Distributed Trust Management. In *Workshop on Norms and Institutions in Multiagent Systems, Autonomous Agents*, Montreal, Canada., may 2001.

[13] P. Komisarczuk, K. Bubendorfer, and K. Chard. Enabling virtual organisations in mobile networks. In *IEE 3G2004 Conference*, pages 123–127, London, UK, October 2004.

[14] H. Lipmaa, N. Asokan, and V. Niemi. Secure vickrey auctions without threshold trust. In M. Blaze, editor, *Financial Cryptography*, volume 2357 of *Lecture Notes in Computer Science*, pages 87–101. Springer, 2002.

[15] J. K. MacKie-Mason and H. R. Varian. Generalized Vickrey Auctions. Working paper, University of Michigan, 1994.

[16] T. W. Malone, R. E. Fikes, K. R. Grant, and M. T. Howard. Enterprise: A Market-like Task Scheduler for Distributed Computing Environments. In H. B.A, editor, *The Ecology of Computation*, pages 177–205. Elsevier Science Publishers (North-Holland), 1988.

[17] M. Naor, B. Pinkas, and R. Sumner. Privacy preserving auctions and mechanism design. In *EC '99: Proceedings of the 1st ACM conference on Electronic commerce*, pages 129–139, New York, NY, USA, 1999. ACM Press.

[18] K. Peng, C. Boyd, E. Dawson, and K. Viswanathan. Five sealed-bid auction models. In *CRPITS '03: Proceedings of the Australasian information security workshop conference on ACSW frontiers 2003*, pages 77–86, Darlinghurst, Australia, Australia, 2003. Australian Computer Society, Inc.

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

[20] W. Vickrey. Counterspeculation, Auctions, and Competitive Sealed Tenders. *The Journal of Finance*, 16(1):8–37, March 1961.

[21] M. Yokoo and K. Suzuki. Secure Multi-agent Dynamic Programming based on Homomorphic Encryption and its Application to Combinatorial Auctions. In *Proceedings of the first joint International Conference on Autonomous Agents and Multiagent Systems*, Bologna, Italy, July 2002. ACM.