MPC Joins The Dark Side

John Cartlidge[1], Nigel P. Smart[1,2], and Younes Talibi Alaoui[2]

[1] University of Bristol, Bristol, UK.
[2] KU Leuven, Leuven, Belgium.
john.cartlidge@bristol.ac.uk,
nigel.smart@kuleuven.be,younes.talibialaoui@kuleuven.be

Abstract. We consider the issue of securing dark pools/markets in the financial services sector. These markets currently are executed via trusted third parties, leading to potential fraud being able to be conducted by the market operators. We present a potential solution to this problem by using Multi-Party Computation to enable a trusted third party to be emulated in software. Our experiments show that whilst the standard market clearing mechanism of Continuous Double Auction in lit markets is not currently viable when executed using MPC, a popular mechanism for evaluating dark markets, namely the volume matching methodology, is viable. We present experimental validation of this conclusion by presenting the expected throughputs for such markets in two popular MPC paradigms; namely the two party dishonest majority setting and the honest majority three party setting.

1 Introduction

> "*The art of trading lies in knowing when and how to expose trading interest. Exposure decisions are the most important decisions large traders make. Traders who never expose never trade. Traders who over-expose generate high transaction costs*" [12].

Successful trading in the financial markets requires balancing the conflicting objectives of finding a counterparty with whom to trade (every seller needs a buyer, and *vice versa*), while attempting not to disclose one's trading interest. Finding a counterparty is most easily solved by publicly announcing an intention — an *order* — to trade. However, if the order is large, other traders in the market are likely to react to this information by moving the price in the adverse direction. This market reaction to the discovery of a large order is known as *price impact*, or *market impact*, and the costs to a trader can be severe: far outweighing commission fees and other trading charges. It has been suggested that market impact increases roughly as the square root of order size [9], although estimates are notoriously difficult and no consensus exists (for a review, see [4]; for technical discussion, see [3, Chapters 11 and 12]). Nevertheless, it *is* universally accepted that publicly exposing one's intention to trade — particularly when trading in large volume — is *costly*.

To reduce market impact, one approach is to disguise large orders by salami-slicing them into multiple smaller orders to be submitted at irregular intervals. However, as execution time inevitably increases, this strategy bears the risk of the market moving away; while traders on the other side of this game of *hide and seek* may determine that further order flow is forthcoming. An alternative approach — one traditionally employed — is to pass the order to a *trusted* broker (or other intermediary) who will attempt to find a natural counterparty within their network of customers and connections. If the order information remains secret within the network, there is no market impact. However, there is incentive for the broker to cheat — using this *insider information* for their own gains by *front running* customers, or by selling the information to a third party. Although often difficult to prove, such (illegal) activity is not uncommon. For instance, in 2005, twenty specialists on the New York Stock Exchange (NYSE) were charged with committing thousands of illicit *front running* trades between 1999 and 2003, causing customer losses in the millions of dollars [31].

To counter the reliance on trusting human intermediaries, so called *dark pool* trading venues have emerged — alternative electronic trading systems that automatically match orders *in private* with no human observers. Unlike the publicly visible orders entered onto the public limit order book (PLOB) of a major "lit" exchange — e.g., the London Stock Exchange (LSE) or NASDAQ — orders entering a dark pool are *invisible*. Orders remain in the dark pool until a match occurs, and only afterwards are the details of the trade published. As trading intention remains secret, even large

orders can execute with little or no market impact. The attraction of dark pools is clear, and the demand from traders is strong. In the first quarter of 2018, approximately 11% of stock trading in the US executed on such alternative trading systems (ATS) [23].

However, where there is trust, there is the possibility of abuse. While dark pools offer trading in secrecy away from prying eyes, the *operators* of dark pools — with full access to system data — are trusted not to spy on, or abuse, the information inside. For some, the temptation has proven too great. For example: in 2011, Pipeline paid a $1 million penalty and settled to charges of conflicts of interest and misleading customers [27]; in 2014, Liquidnet paid a $2 million penalty and settled to charges of improperly using subscribers' confidential trading information in marketing its services [28]; in 2015, ITG (the owner of Posit) paid an $18 million penalty for operating a secret trading desk and misusing the confidential trading information of dark pool subscribers [29]; also in 2015, UBS paid a $12 million penalty for failing to properly disclose to all dark pool subscribers an order type that was marketed almost exclusively to market makers and high frequency trading firms, which allowed those participants to place sub-penny-priced orders that then received priority over other orders [30].

The only way to guarantee privacy is to ensure that *nobody* — not even the system operator — can gain access to the information in the system. One mechanism to achieve such privacy is to apply a multi-party computation (MPC) technique to the underlying algorithm. In such a scenario internal algorithm data is held in secret-shared form, and is processed by a set of servers. If a given ratio (depending on the precise MPC system) of the servers remains honest then the internal algorithm variables do not leak, and thus privacy is preserved. All orders can be entered into such a system using a protocol to convert an external user's order into a secret shared form. Then MPC can be used to perform computation (order matching) on the secret shared data, such that no order information is ever *in the clear*. As an additional advantage if a financial regulator — e.g., the SEC in the US or FCA in the UK — is one of the servers in the MPC computation, not only is privacy guaranteed, but the regulator can guarantee that a specific algorithm was used to perform the matching. This aligns well with MiFID II regulatory compliance, introduced for European markets in Jan 2018 to ensure stricter controls on dark pools.

Our Contribution: Previous work has demonstrated the utility of MPC for secure auctions, with particular success and application in one-time clearing auctions [18] in the famous case of Danish Sugar Beet. Other proof of concept work has been performed, which is detailed in the next section, however, all of this prior work has resulted in matching mechanisms which take many seconds to evaluate.

In this work, we introduce a proof-of-concept fully encrypted (MPC) trading venue using the three main matching algorithms used in a dark markets: (i) *continuous double auction* (CDA) using full limit order book (LOB); (ii) periodic interval crossing (or *periodic auction*); and (iii) scheduled volume match (or *scheduled cross*). All our experiments were conducted using the SCALE-MAMBA MPC system [1].

We find (unsurprisingly given the complexity of the matching algorithm) that whilst CDA is the natural auction methodology in lit markets, when performed in a privacy preserving manner the performance is particularly disappointing. Nevertheless, we are able to process orders in the sub-second, resulting in market throughputs of between roughly 10 and 50 orders per second (depending on the size of the hidden order book). In real world markets, hidden LOBs and CDA matching are more commonly used in dark pools where the venue operator (often a bank) is an active participant, trading on principle (i.e., for themselves) and internally matching client order-flows (e.g., Goldman Sachs' Sigma X2 platform).[3] In Europe, dark pools where the operator is trading on its own account are often classified and regulated under MiFID II, as *Systematic Internalisers*.

Periodic auction mechanisms are algorithmically simpler for MPC than a CDA and have two phases. First, during the open auction period, orders are collected and sorted, similar to the limit order book of a CDA, but without execution. Then, on auction close, a price discovery phase calculates the clearing price that maximises volume traded. All trades execute at this clearing price. We see that the main cost in evaluating a periodic auction procedure in this context ends up being running the algorithms to produce the initial sorted list, i.e. to process the incoming orders before the final clearing is performed. This limits the periodic auction implemented under MPC to be for auctions with a relatively low throughput.

[3] `https://www.goldmansachs.com/what-we-do/securities/gset/equities/liquidity-access/sigma-x2-us-faq.pdf`

To eliminate front running in periodic auctions one could use a smart contracts/blockchain architecture, in which parties commit to their orders, then reveal and execute the clearing trades. However, the process of revealing *all* orders (even the unmet ones) does not satisfy the need to keep such unmet orders secret as one has in dark markets.

Periodic auctions are classified by regulators as "semi-transparent" and so do not fall under MiFID II dark pool compliance. However, we focus on the non-transparent "semi" and therefore *do* consider these as dark pool venues (as the regulators may also do in due course). Real world periodic auction venues include ITG's Posit Auction and LSE's Turquoise Lit Auction.[4]

Finally, we examine scheduled volume matching, the simplest of the three clearing algorithms in which volume is matched but price is determined by reference to some external lit market. For such markets we are able to obtain throughputs of around 800 orders per second. Thus, while such throughput will not be suitable for markets with high frequency trading, for markets where the priority is large volume execution rather than immediacy, such throughput may be sufficient. Scheduled matching has been used by dark pools since their inception (Instinet and Posit, circa 1987) and is still used today by venues such as LSE's Turquoise and ITG's Posit Match.

Paper Overview: We first describe, in Section 2, the problem we are trying to address, and existing problems with performing computations in a non-privacy preserving manner. We also review prior work of applying MPC to financial auctions. In Section 3 we briefly recap on the notation and issues from the MPC literature we require, and we overview the salient properties of the MPC system we will be using. Then, in Section 4, we examine the three auction mechanisms in detail, and describe how to implement them in a privacy preserving manner, and give performance results. Finally, Section 5 concludes that MPC may be ready for real-world dark-pool implementation.

2 Background

Here we outline the required background in financial markets and prior work in applying MPC to this domain.

2.1 Information leakage, insider trading, and front running

Insider trading is the illegal practice of trading to one's own advantage through having access to *confidential information*. A classic example is a director in a public limited company, upon receiving internal news of unexpected poor quarterly results during a board room meeting—and knowing company share price will fall once these results become public—short-selling shares in their own company ahead of the report's announcement (a legal requirement for public companies). Once share prices have fallen, the director is able to buy back shares (at a lower price) to cover the short Position and make a profit. This is illegal; as is selling (or giving) the confidential information to a third party.

Front running is a specific example of insider trading such that an intermediary (e.g., a market maker, specialist, broker, or trading venue operator) acts on advance confidential trading information for one's own gain. For example, let us assume that broker, B, is instructed by client, C, to purchase 20,000 shares in XYZ. The broker can see the current order book is showing the following offers to sell XYZ: 5,000@$49; 15,000@$50; 10,000@$51. If acting honestly, B will execute C's order by purchasing 5,000 shares at $49 and 15,000 shares at $50, for a total cost to C of $995,000. However, since B knows that C's buy order will move the market (i.e., the large buy order will have a positive price impact), B decides to act on the inside information of C's intention to trade by front running the purchase. To this end, on their own account, B buys 5,000 shares at $49 and simultaneously posts an offer to sell 5,000@$50. The order book for XYZ offers is now: 20,000@$50; 10,000@$51. Broker B then executes C's request to purchase 20,000 shares, for a total cost of $50 \times 20,000 = \$1$ million to the client. Broker B has immediately sold their shares (to the client; and at the direct expense of the client) for a risk-free profit of $(50 - 49) \times 5000 = \$5,000$. This practice is illegal. However, it is not always so direct (B could quite easily have sold the advance information of C's intention to buy to a third party), and therefore not always easy to detect.

To avoid these malicious and predatory practices, traders—particularly large institutional investors that trade in large volumes, such as mutual funds—go to great lengths to avoid *information leakage*. Having a trusted broker, or trusted exchange venue, is of primary concern. To address these concerns, alternative "dark pool" trading venues—designed to match orders automatically, in secret, and with no human observer—emerged, and subsequently flourished.

[4] For a full list of European dark pools, see Redburn Execution's List of Approved Trading Venues [21].

2.2 Trading in the Dark

Dark pools are trading systems that do not publicly display orders. The first dark pool crossing networks appeared in the 1980s (Instinet, in 1986, with one end-of-day cross at the securities' closing price; soon followed by ITG's Posit, in 1987, with multiple intraday crosses using the NBBO midpoint as crossing price reference) [11]. Catering for the relatively niche market of large volume traders prepared to forego immediate execution to avoid significant price impact, the market share of trading in dark pools was initially small. However, over the last decade, largely driven by regulation changes (RegNMS, USA, 2005; MiFID, EU, 2007) and the rise of high-frequency trading (HFT), the number of dark pool venues and the volume they trade has ballooned.[5] In the US, around 40 dark pool venues now operate with approximately 15-18% market share of securities trading (a quadrupling since 2005). Similarly, in the EU, the volume traded on the fifteen major dark pools accounts for over 8% of total value traded in equities (a rise from less than 1% in 2009) [20].

While there is considerable variation in the exact mechanisms used, dark pool matching can be roughly categorised into two types: *scheduled*, where crosses occur at fixed times (e.g., the original Instinet and Posit networks); and *continuous*, where crosses occur immediately. Over time, as technologies have improved and communication latencies have fallen, the majority of dark pools have moved from scheduled to continuous crossing. The European Central Bank (ECB) reports that all dark pools in Europe are now continuous [20]. However, options for non-continuous crossing alongside options for continuous crossing are offered by some venues, such as LSE's Turquoise and ITG's Posit. Turquoise offers the option of randomised period crossing (where matching is at random intervals between 10-45 seconds, depending upon liquidity)[6], while the option Posit Match matches orders during a 30-second interval several times a day at pre-determined times [20, p.39].

Dark pools commonly derive execution prices using a "primary" lit venue as reference (e.g., in the US, the midpoint of the NBBO). Executing all trades at a single midpoint reference (e.g., Posit Match and the original crossing networks) provides no price discovery (i.e., the dark pool does not form a price from internal order flow). However, some dark pools (e.g., Goldman Sachs' Sigma X2) provide limited price discovery by maintaining a continuous non-displayed limit order book, where execution prices are bounded between the National Best Bid and Offer (or BBO of some other primary exchange such as the LSE if the dark pool is located in London) [33].

2.3 Out of the dark and into the semi-transparent

As the dark pool trading ecology has evolved, it has become increasingly varied and esoteric. A plethora of matching rules, order types, fee structures, operators, and participants now exist [20][21]. Where there was once a handful of agent-broker networks supplying an exchange venue for large-order customers seeking refuge from market impact, there is now a large and diverse community of providers and participants, including HFT firms looking for short-term profits and broker-dealers trading proprietary flow and acting as market makers.

The complex inter-connected network of dark and lit venues, combined with the inherently secretive nature of dark pools and financial trading in general, has provided an opportunity for dark pool operators to cheat. In some cases, this is illegal practice amounting to insider trading and front running (e.g., see SEC litigations against Pipeline [27], Liquidnet [28], ITG [29], and UBS [30]). However, there are also more subtle forms of "structural insider trading" which are (currently) non-illegal but problematic [32]. Structural insider trading practices enable information leakage and include, for instance, HFTs "algo-sniffing" dark pools. "A particular fear…is that an algorithm that can detect the order in a dark pool, at least probabilistically, can position itself to profit when the purchase or sales in lit markets begin." [16, p.45]

In response, regulators have attempted to fight back by requiring greater trading transparency. In Europe, MiFID II came into force on 3rd January 2018. One of the requirements of MiFID II is the "double volume cap" (DVC), such that in a given share, dark pool trading volume is now limited to 4% at any one venue, or 8% across all dark venues [10].

[5] Here, we are considering *dark pool* liquidity only. Special *dark order* types that enable volume to be hidden on a lit exchange, e.g., an *Iceberg Order*, are not included.

[6] Turqoise Plato Uncross: `https://www.lseg.com/sites/default/files/content/documents/TURQUOISE_UNCROSS_FACTSHEET_2016_AW.pdf`

Interestingly, markets have quickly adapted to the DVC by moving volume away from dark pools to newly-popular periodic auction venues [10]. Unlike continuous central limit order books (CLOBs), where trades are executed instantly as soon as there is a match, in a periodic auction, matching only occurs at the end of a call period. The length of periodic auctions varies by operator, but many are very short. For instance, Turqoise Lit Auctions and Posit Auctions have (randomised) auction periods less than 100 milliseconds.

Periodic auctions offer some of the protections traders look for in dark pool venues, with orders during the call period "hidden". However, periodic auctions are not classified by regulators as dark pools. The UK's Financial Conduct Authority (FCA) are unequivocal: "Periodic auctions are *not* a form of dark trading [because p]ublic information is provided about buying and selling interest during the auction call periods according to MiFID II rules. But, while CLOB trading requires detailed disclosure of buying and selling interest at every price level, periodic auction operators are required only to disclose indicative uncrossing price and volume for the auction" [10].

Given the recent rise in popularity of periodic auctions, in this paper we include a secure implementation of this semi-transparent auction type alongside the traditional dark pool mechanisms: scheduled crossing (matching volume at a given point in time using reference price) and continuous double auction (continuous match with hidden LOB).

2.4 Secure auctions: related work

The problem of avoiding information leakage in financial markets has motivated several previous studies investigating the potential of homomorphic encryption and multi-party computation (MPC) to achieve secure auction protocols. Here, we briefly summarise this work.

In 2007, Thrope and Parkes introduced a secure protocol for a continuous double auction (CDA) mechanism with limit order and market order types [25]. Before order $O(p, q, d)$ is entered, the trader encrypts order price, p, quantity, q, and direction, d (bid/buy or ask/sell), using the market operator's public key.[7] The encrypted order, $E(p, q, d)$, is then sent to the exchange, whereby the market operator privately decrypts E to obtain O. The order O is entered into a limit order book (LOB) and matching is performed *in the clear*. Post-execution, trades are published in encrypted form, enabling delayed secrecy-preserving post-trade proof checking of the correctness of the market operation. An empirical evaluation of the proposed protocol running on low-end contemporary commodity hardware suggested an implementation of the system would have operational costs of approximately 5 cents (US) to place and verify an order. Later extensions by the authors and their colleagues included a combinatorial auction mechanism (trading *baskets* of stocks) [24] and the ability to enter more sophisticated *conditional rule-based* order types [26]. However, the underlying encryption protocol remained unchanged: traders are required to post orders encrypted using the operator's public key, and the operator matches orders *in the clear*. Therefore, while the post-trade audit trail is secure, the real-time market information is not. In each case, the proposed protocols require trusting the market operator; thus enabling the possibility of information leakage, or front-running traders' order flow.

Bogetoft and colleagues' seminal work in the area of secure auctions was introduced and developed over several years: from protocol design in 2006 [2], to first real-world implementation in 2008 [18], and finally commercialisation [19]. Avoiding the pitfalls of trusting a single market operator, Bogetoft et al.'s approach is to replace a single auctioneer (market provider) by a set of n Trusted Third Parties (TTPs), where it is assumed that at most some number t—the threshold trust—of TTPs are corrupt. Multi-party computation (MPC) is used to emulate the auctioneer, ensuring that access to order information is not available *in the clear* to any one party. The protocols can tolerate any set of less than $n/2$ TTPs sharing information, where typical values of (n, t) are $(3, 1)$ and $(5, 2)$.

Bogetoft et al. implement a one-shot double auction mechanism (essentially a periodic auction that is run only once) where bid and ask orders are first received during an open auction period [2]. After auction deadline, the market executes all trades at the same market clearing price, p_c, calculated as the price that minimises excess demand and supply. At price p, if aggregate demand, ΣD_p, is greater than aggregate supply, ΣS_p, then excess demand $ED_p = \Sigma D_p - \Sigma S_p > 0$ and excess supply $ES_p = 0$. If $\Sigma D_p - \Sigma S_p < 0$ then we have excess supply $ES_p = \Sigma S_p - \Sigma D_p > 0$ and $ED_p = 0$.

In the first real-world application [18], the authors develop a system used by Danish farmers to trade contracts for sugar beet production on a nation-wide market. The system implements an electronic double auction, where the role of the auctioneer is played by three parties—a $(3, 1)$ TTP model—(i) Danisco, the only sugar beets processor on the

[7] For market order types, value p is not needed.

Danish market; (ii) DKS, the sugar beet growers' association; and (iii) SIMAP, the research project team. During the auction period, each farmer was able to send encrypted orders to the three parties, who then compute on the data *in protected form*. Therefore, no single party ever has access to any order in the clear. In total, 1229 farmers submitted orders. The auction computation was performed on 14 Jan 2008, and approximately 25,000 tons of production rights changed ownership. Timings for the live auction computation were not presented, but on contemporary commodity hardware, tests showed computations for 1000 traders took around 30 minutes; for 3000 traders around 75 minutes. The authors have since commercialised their system through a private company named Partisia [19]. Partisia continues to offer a double auction mechanism with single clearing price using MPC. The one-shot double auction mechanism is particularly suited to a one-off high stake auction with sealed bids and long auction durations. According to Partisia's website, their platform has been tailored for the Norwegian Spectrum Auction to trade spectrum rights for a total of NOK 877.983.276 (approximately USD $100 million) over the course of 7 days and 83 bidding rounds in December 2015.[8]

Using MPC for periodic auctions is further developed by Jutla in 2015 [13]. Jutla argues that while MPC technology's high computation costs make it unsuitable to replace the continuous double auction (CDA)—the predominant mechanism used in modern electronic financial markets—MPC technology *is* (as of 2015) capable of running repeated periodic auctions for financial markets, using a 30 minutes opening auction, followed by a succession of 15 minute auctions, with 5 minute gaps in-between for processing and information digestion. These timings follow the open-auction period (30 minutes) of specialists on the New York Stock Exchange (NYSE). In Jutla's architecture, the market protocol is a secure five party-computation (5-PC), run by a small number of brokers, say 4, and one regulating authority such as the securities and exchange commission (SEC). To ensure regulatory oversight, the SEC (or any other party) can audit saved computations and check if they were performed according to the protocol. During the auction period, traders can enter limit and market orders. At the end of each auction round, the market is cleared (at a single price) and price and volume information is revealed. Uncleared orders remain in the market for the following auction period. Economic modelling of the protocol suggests behaviour (price discovery, information risk, and research advantage) is comparable, or favourable, to a CDA and Walrasian equilibrium. However, the protocol is not implemented or empirically tested in this work, and to date the work has not been published.[9]

Recently, Massacci et al. [17] demonstrated a proof-of-concept implementation of a secure futures market using distributed ledger technology, where traders hide behind a Tor network to communicate anonymously. Designed to replicate the functionality of the Chicago Mercantile Exchange (CME), the system uses a CDA mechanism for order matching. The focus of this work is on enabling anonymity of *who* is executing a trade as opposed to securing *what* and *how much* is being traded as in our work. The methodology uses MPC for a small subset of the operations, so as to enable privacy of who is trading. A proof of concept implementation is demonstrated, containing a population of 10 traders and an order book with five levels. Results show that individual operations—e.g., post order, cancel order, etc.—can be performed in around 24s.[10] Whilst addressing part of the security problem the methodology still requires a trusted third party with access to secret inputs of all participating traders, therefore enabling the potential for information leakage and front-running by the "trusted" party.

In summary, previous work has split focus between CDA and periodic auctions. Computationally more burdensome, work in CDA markets is less mature: with matching in the clear [25, 24, 26], slow operation times (e.g., 24s for post-order operations in a market containing only 10 traders), and the use of a trusted third party with access to private inventory information [17]. Fully secure CDA using MPC has not yet been implemented and stress tested for performance. In comparison, secure clearing price one shot double auctions (i.e., a periodic auction run once, over a long time period) have been implemented [2, 18] and commercialised [19].

3 MPC Background

Our MPC experiments are based on the SCALE-MAMBA system [1]. This is a secret sharing based MPC system in which an internal data item x is held in secret shared form, a process which we shall denote by $\langle x \rangle$. All the data items

[8] For details, see: `https://partisia.com/spectrum-auctions/`

[9] Personal communication with the author, Oct 2018.

[10] For comparison, CME Globex report an average median latency for order entry of 200 microseconds during 2017 [7, p.2].

need to be represented in a finite field $\mathbb{F}_p$, thus $x \in \mathbb{F}_p$. Calculations are performed by expressing the computation in terms of adding, multiplication and opening values in $\mathbb{F}_p$.

The SCALE-MAMBA system implements an actively-secure-with-abort MPC protocol in the pre-processing model. This means that the protocol is guaranteed to provide privacy, and if a set of adversarial parties deviate from the protocol then the honest parties will abort the protocol with overwhelming probability. The probability of aborting in the case of malicious behaviour by a party is given by $1 - 1/p$, and hence we select $\log_2 p = 128$ to make this overwhelmingly likely.

In addition the algorithms are executed in two phases. In the first offline phase function independent data is produced (for example Beaver triples), which are then consumed in a function specific online phase. This maps well to the situation we are dealing with, as the offline data could be produced overnight, and then one only needs to execute the online phase during the day when the specific market is open for trading.

The ability to perform pre-processing enables special data types to be created. The most standard of these are secret shared Beaver triples; which are a triple of shared values $(\langle a \rangle, \langle b \rangle, \langle c \rangle)$ where a and b are uniformly random in $\mathbb{F}_p$ and $c = a \cdot b$. Such triples enable a fast online multiplication procedure. However, SCALE-MAMBA makes use of extensive pre-processing of shared random bits; namely sharings $\langle b \rangle$ where $b \in \{0, 1\}$ is outside the control of any subset of parties. Such shared random bits are used to enable efficient integer operations (see below), which are crucial to our application.

SCALE-MAMBA provides various underlying secret sharing schemes to use within this methodology. The main one is that of full threshold, in which if we have n MPC engines then security is gauranteed unless all n are corrupt. In this case the system implements the SPDZ protocol and its derivatives [8], which utilizes Somewhat Homomorphic Encryption to implement the offline phase. In this case we consider the case of having only two servers.

The other secret sharing scheme we consider is that of Shamir sharing with three players. Here the SCALE system implements the methodoloy from [14], this latter methodology results in a faster online phase then for example using the methodology in [6] which provides a shorter overall run time (at the expense of more expensive online computation).

As a basic set up, we assume the n MPC servers are executing the dark market operation. Entities which wish to place orders are connected to these n MPC servers using secure connections. These secure connections enable the external traders to place orders into the market in such a way that the servers do not learn the input. Such a protocol is easy to produce, as follows. The servers take a shared random value $\langle r \rangle$ and open it towards the inputing trader (so the trader learns r). The trader then inputs their value as $x + r$, which is essentially a one-time pad encryption of x. The MPC servers can then compute a sharing of x from the equation $\langle x \rangle = (x + r) - \langle r \rangle$.

3.1 Arithmetic on Integers

As explained above, the MPC engine provides the ability to perform arithmetic with integers modulo a prime p. However, in our algorithms we want to do arithmetic on integers. To do this we encode an integer in the range $[-2^{k-1}, \ldots, 2^{k-1}]$ as its representative modulo p. In our algorithms we ensure that there is no wrap around of the integers modulo 2^{k-1}, which is easy as we are basically performing a number of conditional summations, and the conditionals themselves.

To perform conditional operations, such as $b \leftarrow x < y$, we follow the method of [5]. At its heart, this requires we take a shared value $\langle x \rangle$ with $x \in [-2^{k-1}, \ldots, 2^{k-1}]$, mask it with a value $\langle r \rangle$ by performing $\langle x + r \rangle \leftarrow \langle x \rangle + \langle r \rangle$, and then opening $\langle x \rangle$ so that everyone obtains $z = x + r$. The problem is that this reveals information about x if r is not large enough. In particular the statistical distance of the value z from the uniform distribution is $2^{-\mathsf{sec}}$ if we select r from the range $[-2^{\mathsf{sec}+k-1}, \ldots, 2^{\mathsf{sec}+k-1}]$. That is we need to select r from a range which is 2^{sec} times larger than the range of x. This parameter sec is called the *statistical security parameter for arithmetic*. In our work we select $\mathsf{sec} = 40$ and $k = 64$. To ensure valid arithmetic we need to select p such that $k + \mathsf{sec} < \log_2 p$. Which, given we select $\log_2 p = 128$, we are well within range.

The methodology used in [5] is relatively efficient in SCALE-MAMBA due to the pre-production of shared random bits in the offline phase. The protocols in [5] are described in the context that such shared bits are produced in the online phase, whereas pre-processing them produces a much more efficient online phase. In Table 1 we outline the costs in terms of pre-processing and online rounds of communication needed by the four main operations which require interaction and pre-processing; namely opening, multiplication, comparison and equality testing.

Table 1. Costs of operations

Operation	Open	$\langle a \rangle \cdot \langle b \rangle$	$\langle a \rangle < \langle b \rangle$ $\langle a \rangle < b$	$\langle a \rangle = \langle b \rangle$ $\langle a \rangle = b$
Triples	0	1	120	63
Bits	0	0	105	104
Rnds of Comm.	1	1	7	7

3.2 Experimental Setup

After presenting each methodology below we give the experimental run times we achieved using the SCALE-MAMBA system. All our experiments were carried out on a set of machines running Ubuntu. Each machine was identical, having i7-7700K CPUs and 32 GB of RAM. The ping time between machines was .47 milliseconds and the machines were connected by a 10 GBit switch.

4 Algorithms

In this section we detail the three algorithms we will be comparing. The first is the standard Continuous Double Auction (CDA) method, using the full limit order book. This method matches orders in real time as the bid/offer comes in. Each order is in the form of a volume/price pair. The second is the Periodic Auction, which allows users to enter bid/offers in a given period, and then at the end of the period it determines the matched orders, and the clearing price for all orders executed. Again, each order is in the form of a volume/price pair. Finally, we implement our Scheduled Cross methodology which simply determines, based on the input volumes, which orders are matched. In this method the price is not entered, with the final transaction price being determined by reference to an external market value.

In all cases, we assume in our algorithms that we are currently dealing with N sell orders (or *offers*), and M buy orders (or *bids*). These are given by volume values s_i for $i = 1, \ldots, N$ and b_i for $i = 1, \ldots, M$, if we require price information we use the values q_i for sell orders and p_i for buy orders. For convenience we assume that $s_i, b_i, p_i, q_i \in \mathbb{Z}$ and are in the range $[0, \ldots, 2^{63} - 1]$.

As described above, we ran our experiments in two configurations of the SCALE-MAMBA system. In the first configuration we used three servers and a dishonest majority protocol based on Shamir secret sharing, over the ring defined by a prime of 128 bits. In the second configuration we used a two server full threshold implementation, with a prime p of 128 bits.

For each algorithm we recorded a number of factors which affect the run-time; such as the number of multiplication triples 'm' and shared bits 'b' needed to be produced by the offline phase, as well as the number of rounds of communication 'r' needed for the online phase. We also timed the respective times for the online and offline phases.

When reading the algorithms the reader should bear in mind that secure additions comes for free, but the main cost will be the secure multiplications and secure comparisons. A secure comparison results in a shared value of one (for true) and a shared value of zero (for false). The actual logical value is kept secret shared, and hence unknown to the parties.

In all algorithms we are interested in two metrics: the *latency* of each operation — how long it takes to process a single operation; and the *throughput* — the number of trades per second that can be accomplished by the given method.

4.1 CDA Method

At the heart of a CDA is the limit order book (LOB). The LOB contains a sorted list of bids (buy orders) and a sorted list of offers (sell orders), where bids are ordered by price *descending* and offers are ordered by price *ascending* (for two orders with equal price, the order with the earlier timestamp comes first). All orders contain a price and a volume to trade. At the "top" of the LOB, the highest priced bid and lowest priced offer are referred to as the best bid and offer

(BBO). The price difference between the best bed and best offer is called the "spread". When a new order enters the market, if it "crosses the spread" (i.e., if an incoming bid price is equal to or greater than best offer; or incoming offer price is less than or equal to best bid) then there is a "match" and an execution (transaction) will occur, at the price of the resting order (i.e., the best bid, or best offer). Incoming orders will continue to transact with matching orders in the LOB until no volume remains (i.e., the incoming order is fulfilled), or no matches are possible. Orders that do not immediately execute (i.e., do not cross the spread) will enter the LOB (in price-ordered position). The CDA is used for all major lit exchanges and also for some dark pool venues, such Goldman Sachs' Sigma X2. Real markets often include more exotic order types than the simple "limit" orders that we use in this analysis, but we do not believe that these will significantly affect the analysis and so are ignored. Prior work on secure CDA implementations is scarce, and to our knowledge a secure CDA implementation using MPC has not previously been presented or tested. In [25], a "secure" CDA protocol is introduced, but the operator can see the information in the CDA, so this defeats the object for our purpose. In [17], a "secure" CDA is implemented using distributed ledger technology. However, this method is designed to keep *who* is trading a secret, not *what* is being traded. Again, this is not sufficient for our purpose.

In a CDA, each seller (resp. buyer) has a name, an amount, and a price, each of which is meant to be kept secret during the auction. Thus, we assume there is a *state* of a set of buy $\mathcal{B} = [(\langle \mathsf{name}_i^s \rangle, \langle b_i \rangle, \langle q_i \rangle)]_{i=1}^M$ and sell $\mathcal{S} = [(\langle \mathsf{name}_i^b \rangle, \langle s_i \rangle, \langle p_i \rangle)]_{i=1}^N$ orders. The buy orders are at any one time ordered such that $q_i \geq q_{i+1}$ (with ties being sorted by the time they were submitted), whereas sell orders are ordered such that $p_i \leq p_{i+1}$. The state is also such that $p_1 > q_1$, otherwise a trade would have happened. In this initial state we have $N = M = 0$, and thus the algorithm should take new orders in and maintain this sorted state list.

We assume the fact an order is buy or sell can leak during the auction, just not the price or the volume. We present an algorithm for coping with a new buy order in Figure 3, which uses two sub-procedures, shown in Figure 1 and Figure 2. The operation for buy orders is symmetric, and thus we do not give the details for buy orders. The price paid is the price of the matching resting order in the order book (the sell list, $\mathcal{S}$), in this case the sell order's price.

Process Sell List for CDA

Given a new buy order $(\langle \mathsf{name}_0^b \rangle, \langle b_0 \rangle, \langle q_0 \rangle)$ we process the sell list, and check if any items sell. If so we remove these, update the list, and also update the current buy order. This clearly leaks how many orders on the sell list are fulfilled.

(1) Repeat
- (I) $\langle z_2 \rangle \leftarrow \langle b_0 \rangle > 0$.
- (II) $\langle f \rangle \leftarrow \langle q_0 \rangle \geq \langle p_1 \rangle$.
- (III) $\langle f \rangle \leftarrow \langle f \rangle \cdot \langle z_2 \rangle$.
- (IV) Open $\langle f \rangle$.
- (V) If $f = 1$ then
  - (A) $\langle z_1 \rangle \leftarrow \langle b_0 \rangle \geq \langle s_1 \rangle$.
  - (B) Open $\langle z_1 \rangle$.
  - (C) $\langle t \rangle \leftarrow z_1 \cdot (\langle s_1 \rangle - \langle b_0 \rangle) + \langle b_0 \rangle$.
  - (D) Print Sell and open $(\langle \mathsf{name}_1^s \rangle, \langle t \rangle, \langle p_1 \rangle)$.
    [i.e. Party name_1^s on the sell list has sold t items at price p_1]
  - (E) $\langle s_1 \rangle \leftarrow \langle s_1 \rangle - t$.
  - (F) If $z_1 = 1$ then
    - (i) Delete item one from the sell list, and relabel two as one, three as two, and so on.
  - (G) $\langle b_0 \rangle \leftarrow \langle b_0 \rangle - t$.
- (VI) Until $f = 0$ or $N \neq 0$.

Figure 1. Process Sell List for CDA

Note that the algorithm runs in two phases. In the first phase, the new buy order is processed by matching it against any offer on the sell list for which the price of the buy order is greater than the corresponding sell order. The first phase leaks the number of sell orders which have been satisfied (indeed, that is inherent in the auction as parties will always

learn after the event which orders have been completed) and the run time of the first phase of the algorithm depends on the number of satisfied sell orders.

Insert into Buy List

Given a new buy order $(\langle \mathsf{name}_0^b \rangle, \langle b_0 \rangle, \langle q_0 \rangle)$ this inserts it into the sorted list of M existing buy orders, producing a new sorted list of $M+1$ buy orders. An identical algorithm can used to do the same for the sell orders, bar changing the inequalities in lines (1) and (I).
To do this we insert a dummy element at the end of the buy list of the form $(\langle 0 \rangle, \langle 0 \rangle, \langle 0 \rangle)$, i.e. a buy order of price zero. Note if $b_0 = 0$ then this procedure just inserts at the end, so we do not even reveal if, when we get here, we have $b_0 = 0$ or not.

(1) $\langle f_0 \rangle \leftarrow \langle q_0 \rangle < \langle q_1 \rangle$.
(2) For $i = 1, \ldots, M+1$ do
  (I) $\langle f_i \rangle \leftarrow \langle q_0 \rangle \leq \langle q_i \rangle$.
  (II) $\langle f_i' \rangle \leftarrow (1 - \langle f_i \rangle) \cdot \langle f_{i-1} \rangle$.
  (III) $\langle f_i'' \rangle \leftarrow (1 - \langle f_i \rangle) \cdot (1 - \langle f_i' \rangle)$.
  (IV) $\langle \mathsf{name}_i^{b\prime} \rangle \leftarrow \langle f_i \rangle \cdot \langle \mathsf{name}_i^b \rangle + \langle f_i' \rangle \cdot \langle \mathsf{name}_0^b \rangle + \langle f_i'' \rangle \cdot \langle \mathsf{name}_{i-1}^b \rangle$.
  (V) $\langle b_i' \rangle \leftarrow \langle f_i \rangle \cdot \langle b_i \rangle + \langle f_i' \rangle \cdot \langle b_0 \rangle + \langle f_i'' \rangle \cdot \langle b_{i-1} \rangle$.
  (VI) $\langle q_i' \rangle \leftarrow \langle f_i \rangle \cdot \langle q_i \rangle + \langle f_i' \rangle \cdot \langle q_0 \rangle + \langle f_i'' \rangle \cdot \langle q_{i-1} \rangle$.
(3) Output $[(\langle \mathsf{name}_i^{b\prime} \rangle, \langle b_i' \rangle, \langle q_i' \rangle)]_{i=1}^{M+1}$.

Figure 2. Insert into Buy List

In the second phase, any remaining quantity is added into the current buy list, using an insertion sort. At this point, if the quantity remaining in the buy order is zero, we set the price to also be zero. Thus, in the latter case we end up inserting at the end of the buy list, $\mathcal{B}$.

Algorithm for CDA

Suppose we have state $\mathcal{S}$ and $\mathcal{B}$ as above, also suppose a new buy order $(\langle \mathsf{name}_0^b \rangle, \langle b_0 \rangle, \langle q_0 \rangle)$ comes in; a similar operation will happen when a new sell order comes in.

(1) Call the algorithm from Figure 1.
(2) $\langle e \rangle \leftarrow \langle b_0 \rangle = 0$.
(3) $\langle q_0 \rangle \leftarrow \langle q_0 \rangle \cdot (1 - \langle e \rangle)$.
  [Set the buy price to zero if no items are left in this order]
(4) We now need to insert $(\langle b_0 \rangle, \langle q_0 \rangle)$ into the buy list without revealing where it goes, which is done by calling the algorithm from Figure 2.

Figure 3. Algorithm for CDA

To understand the costs of these three algorithms in terms of pre-processed data required and rounds of communication in the online phase, we present Table 2. In this table we let s denote the number of opened sell orders, which is the number of executions of the loop in Figure 1.

Experimental Results Since the run time of the first phase depends on the number of executed sell orders (and to some extent on the total number of sell orders N), given a single incoming buy order, and the run time of the second phase depends on the number of buy orders M on the book, we set up our experiments as follows: For various values of N and M we generated an existing buy and sell order book, consisting of two secret shared sorted lists of $(\mathsf{name}_i^b, b_i, q_i)$ (resp. $(\mathsf{name}_i^s, s_i, p_i)$) values such that $b_i, s_i \in \{1, \ldots, 100\}$ and the difference between successive prices in each list

Table 2. Costs of the Components of the CDA Algorithm

	Figure 1	Figure 2	Figure 3	Total
Multiplications	$11 \cdot (M+1)$	1	N	$11 \cdot M + N + 12$
Comparisons	$M+2$	0	$3 \cdot N$	$M + 3 \cdot N + 2$
Equality tests	0	1	0	1
Rounds of communication	10	8	$9 \cdot (s < N) + 18 \cdot s$	$18 + 9 \cdot (s < N) + 18 \cdot s$

differ by at most one. Then, for each (N, M) pair, we ran four tests. In each test, we selected a new buy order which we knew would result in opening exactly s sell orders, for $s \in \{0, 1, 2, 3\}$.

Our results are given in Tables 3 and 4, respectively, for the case of honest majority three parties (using Shamir sharing) and the case of two parties using full threshold sharing. We measure online throughput, in terms of number of transactions per second, which can be evaluated using this method, and the latency (in seconds) for each order to be processed. As explained above, the online execution time for the CDA algorithm depends only on the number of opened sells s (assuming an incoming buy order is being processed), as well as the size of the order book (N, M). The offline time depends soley on the size of the order book (N, M). In both cases the throughput for the CDA algorithm is one over the latency, as we only process one transaction per execution. Whilst the offline throughputs are less interesting, as one can run the offline processing "overnight", the throughputs for the online processing are not very promising. Thus, while the CDA method is the algorithm of choice in lit markets, it is not well suited to the case of evaluation in an MPC environment for a Dark Market.

Table 3. Online Latency and Throughput for CDA Algorithm. We assume an order books of size $10 < N, M < 50$.

Number of Opened Sell Orders (s)	Rounds of Communication	Two Party Case		Three Party Case	
		Latency	Throughput	Latency	Throughput
0	27	0.004 - 0.012	83 - 250	0.007 - 0.023	43 - 142
1	45	0.005 - 0.013	76 - 200	0.008 - 0.025	40 - 125
2	63	0.006 - 0.014	71 - 166	0.010 - 0.027	37 - 100
3	81	0.007 - 0.015	66 - 142	0.012 - 0.029	34 - 83

Table 4. Offline Latency and Throughput for CDA Algorithm.

		Two Party Case		Three Party Case	
N	M	Latency	Throughput	Latency	Throughput
10	10	4.879	.20	0.095	10.5
10	20	6.099	.16	0.129	7.7
20	10	8.243	.12	0.138	7.2
20	20	9.464	.10	0.164	6.1
20	40	11.906	.08	0.212	4.7
40	20	16.193	.06	0.282	3.5
50	50	23.220	.04	0.376	2.6

4.2 Periodic Auction Method

Periodic auctions have two distinct phases: (i) during the open auction period, limit orders are submitted and stored in price ordered lists (bids and offers); (ii) at auction close, "clearing" is performed to find a single price that will maximise the volume traded. We call these the bid input phase, and the bid completion phase. Orders able to execute at this clearing price are cleared (i.e., transact). Unexecuted orders remain. A new open auction period then begins and the cycle repeats. Prior work for clearing price double auctions (essentially a periodic auction that is run only once) using MPC has successfully demonstrated implementation [2] and real-world application for sugar beet contracts [18]. However, in this work, time is a weak constraint, with auction periods of the order of a day. Testing showed computations for 1000 orders took around 30 minutes (i.e., a throughput of 0.55 orders per second). In contrast, real-world auction periods for financial markets are very short (e.g., Turquoise Lit Auctions and Posit Auctions have auction periods lasting 100 ms), so practical MPC implementation of a periodic auction requires high sub-second throughput. As far as we are aware, no prior work has evaluated MPC for periodic auctions at the speeds (throughput) we consider here.

The bid input phase is executed as a bid is entered during the period under consideration, and is essentially the insertion sort performed at the end of the earlier CDA algorithm, i.e. the algorithm in Figure 2. The second phase of the period auction is to complete the actual orders. This is done by (essentially) calling the algorithm in Figure 1, there are however some modifications, which we outline in Figure 4.

Periodic Auction Bid Completion Phase

On input of sorted lists of buy and sell orders, $[(\langle \mathsf{name}_i^b \rangle, \langle b_i \rangle, \langle q_i \rangle)]_{i=1}^M$ and $[(\langle \mathsf{name}_i^s \rangle, \langle s_i \rangle, \langle p_i \rangle)]_{i=1}^N$, this algorithm outputs the identities of the matched orders, but not the prices. The output is a list of completed orders on the buy and sell side, $\mathcal{S}^C$ and $\mathcal{B}^C$.

(1) $\mathcal{S}^C = \mathcal{B}^C = \emptyset$.
(2) $\langle b \rangle \leftarrow 1$.
(3) While $N \neq 0$ and $M \neq 0$ do
  (I) $\langle f \rangle \leftarrow \langle b \rangle \cdot (\langle q_1 \rangle \geq \langle p_1 \rangle)$.
  (II) Open $\langle f \rangle$
  (III) If $f = 0$ then exit.
  (IV) Execute the algorithm in Figure 1, except do not reveal the sell price in Step (D). Instead add $(\mathsf{name}_1^s, t, \langle p_1 \rangle)$ into $\mathcal{S}^C$. Open $\langle \mathsf{name}_1 \rangle^b$ and add $(\mathsf{name}_1^b, t, \langle q_1 \rangle)$ into $\mathcal{S}^B$.
  (V) $\langle b \rangle \leftarrow (\langle b_1 \rangle = 0)$.
  (VI) Delete item one from the buy list, and relabel two as one, three as two, and so on.
(4) Output $\mathcal{S}^C$ and $\mathcal{B}^C$.

Figure 4. Periodic Auction Bid Completion Phase

This final phase needs to be augmented by an algorithm to determine the actual price which all trades should execute at. This is done following two rules, where $\mathcal{S}$ and $\mathcal{B}$ are the the sell and the buy lists after executing the completion phase, and $\mathcal{S}^C$ and $\mathcal{B}^C$ are the list of completed orders.

- The clearing price has to maximize the volume traded.
- If a set $\mathcal{P}$ of prices maximizes the volume traded, the clearing price is :
  - The highest in $\mathcal{P}$ if $\mathcal{B}$ contains unexecuted orders (orders with a price p, where $\mathcal{B}^C$ contains as well orders with the same price p).
  - The lowest in $\mathcal{P}$ if $\mathcal{S}$ contains unexecuted orders.
  - The mid price of $\mathcal{P}$ otherwise.

The first rule means that the clearing price completely executes all orders in $\mathcal{S}^C$ and $\mathcal{B}^C$, as these are the maximum amounts of orders for sells and for buys we can execute (and therefore, the maximum volume traded). This implies that the clearing price is in $[p'_{v'}, q'_{u'}]$, where $p'_{v'}$ is the price of the last order added into $\mathcal{S}^C$, and $q'_{u'}$ is the price of the

last order added into $\mathcal{B}^C$. If $p'_{v'} = q'_{u'}$, then the clearing price is identified, otherwise, we need to refer to the second rule. The second rule implies that we need to take either $q'_{u'}$ if we have unexecuted orders in $\mathcal{B}$, or $p'_{v'}$ if we have unexecuted orders in $\mathcal{S}$, or $\frac{q'_{u'}+p'_{v'}}{2}$ otherwise. Thus we only need to check whether $q'_{u'}$ is the same as q_1, the price of the first order in $\mathcal{B}$, and then check whether $p'_{v'}$ is the same as p_1, the price of the first order in $\mathcal{S}$. See Figure 5 for a formal description of this algorithm.

Periodic Auction Price Calculation

Given the sets $\mathcal{B}$, $\mathcal{S}$, $\mathcal{B}^C$ and $\mathcal{S}^C$ after the completion phase: $\mathcal{B} = [(\langle \mathsf{name}_i^b \rangle, \langle b_i \rangle, \langle q_i \rangle)]_{i=1}^{u}$ the buy list, $\mathcal{S} = [(\langle \mathsf{name}_i^s \rangle, \langle s_i \rangle, \langle p_i \rangle)]_{i=1}^{v}$ the sell list, $\mathcal{B}^C = [(\langle \mathsf{name}_i^{b'} \rangle, \langle b'_i \rangle, \langle q'_i \rangle)]_{i=1}^{u'}$ the list of completed buys, $\mathcal{S}^C = [(\langle \mathsf{name}_i^{s'} \rangle, \langle s'_i \rangle, \langle p'_i \rangle)]_{i=1}^{v'}$ the list of completed sells. This algorithm determines the final price, price.

(1) If $\mathcal{B}^C = \emptyset$ (And therefore $\mathcal{S}^C = \emptyset$) then exit
[i.e. No matching happened during this period]
(2) Else
(I) If ($\mathcal{B} = \emptyset$ and $\mathcal{S} = \emptyset$) then $\langle \mathsf{price} \rangle \leftarrow (\langle q'_{u'} \rangle + \langle p'_{v'} \rangle)/2$
[All sells and buys were completely matched during completion phase]
(II) Else
(A) If ($\mathcal{B} \neq \emptyset$ and $\mathcal{S} = \emptyset$)
[Only sells were completely matched during completion phase]
(i) $\langle g \rangle \leftarrow (\langle q_1 \rangle = \langle q'_{u'} \rangle)$.
(ii) $\langle \mathsf{price} \rangle \leftarrow \langle g \rangle \cdot \langle q_1 \rangle + (1 - \langle g \rangle) \cdot (\langle q'_{u'} \rangle + \langle p'_{v'} \rangle)/2$.
(B) Else
(i) If ($\mathcal{B} = \emptyset$ and $\mathcal{S} \neq \emptyset$)
[Only buys were completely matched during completion phase]
(a) $\langle h \rangle \leftarrow (\langle p_1 \rangle = \langle p'_{v'} \rangle)$.
(b) $\langle \mathsf{price} \rangle \leftarrow \langle h \rangle \cdot \langle p_1 \rangle + (1 - \langle h \rangle) \cdot (\langle q'_{u'} \rangle + \langle p'_{v'} \rangle)/2$.
(ii) Else
[Neither sells, nor buys were completely matched during completion phase]
(a) $\langle g \rangle \leftarrow (\langle q_1 \rangle = \langle q'_{u'} \rangle)$.
(b) $\langle h \rangle \leftarrow (\langle p_1 \rangle = \langle p'_{v'} \rangle)$.
(c) $\langle \mathsf{price} \rangle \leftarrow \langle g \rangle \cdot \langle q_1 \rangle + \langle h \rangle \cdot \langle p_1 \rangle$.
(d) $\langle \mathsf{price} \rangle = \langle \mathsf{price} \rangle + (1 - \langle g \rangle) \cdot (1 - \langle h \rangle) \cdot (\langle q'_{u'} \rangle + \langle p'_{v'} \rangle)/2$.
(III) Open $\langle \mathsf{price} \rangle$ and return it as the final price

Figure 5. Periodic Auction Price Calculation

Experimental Results As explained above the periodic auction consists of three distinct phases, which we time separately. The first phase, which is the algorithm in Figure 2 needs to be executed *as each order comes in*. In Table 5 we present the online execution time of this phase of the algorithm in the case of processing a buy order, for various values of M. Obviously as each order comes in the value of M increases by one. We see that, roughly, the online time increases linearly as a function of the size of the order book (which is to be expected). Thus, in a real market, this first phase will need to complete as soon as the order book reaches a size that the current incoming orders cannot be processed as they arrive. Once the throughput of incoming orders is greater than one over the online latency needed to deal with the incoming orders, the algorithm will be unable to cope. Thus, this cross over point will determine the length of the period in the periodic auction.

We now turn to examining the phase which happen at the end of the specific period, namely the execution of the bid completion phase from Figure 4 and the price calculation from Figure 5. The main cost in the evaluation of Figure 4 is the evaluation of line 3-IV. The rest of the calculation just depends on a little book keeping consisting of two comparisons and one multiplication per iteration of the loop (3). The number of iterations of the loop (3) depending on

Table 5. Insert into Buy List (Figure 2) Results

N	Rounds	Pre-Processed Data (m,b)	Two Party Case		Three Party Case	
			Offline Time	Online Time	Offline Time	Online Time
10	10	(1561,1260)	1.454	0.002	0.060	0.005
20	10	(2871,2310)	2.675	0.004	0.073	0.010
50	10	(6801,5460)	6.338	0.010	0.135	0.021
100	10	(13351,10710)	12.443	0.022	0.238	0.042
200	10	(26451,21210)	24.652	0.054	0.437	0.090

precisely what quantities are matched in the main step of line 3-IV (i.e. essentially how many sell items are opened). The experimental results for this step are presented in Tables 6 and 7. The online run times in Table 6 do not depend significantly on N, thus we only give the range of values for the time for values of N in the range 10 to 500. The offline runtimes depend more on the value of N. Finally, the price calculations consists of one opening, and at most two comparisons, so the cost is negligible.

Table 6. Online Latency for Step 3-IV of Periodic Auction Bid Completion Phase (Figure 4)

Number of Opened Sells	Rnds	Two Party Case Latency	Three Party Case Latency
0	9	0.001 - 0.001	0.001 - 0.001
1	27	0.002 - 0.002	0.003 - 0.003
2	45	0.003 - 0.004	0.004 - 0.005
3	63	0.004 - 0.005	0.006 - 0.008

Table 7. Offline Latency for Step 3-IV of Periodic Auction Bid Completion Phase (Figure 4)

N	Pre-Processed Data (m,b)	Two Party Case Latency	Three Party Case Latency
10	(3610,3150)	3.364	0.073
20	(7220,6300)	6.729	0.137
50	(18050, 15750)	16.822	0.288
100	(36100,31500)	33.645	0.492
200	(72200,63000)	67.290	0.997
500	(180500,157500)	168.226	2.449

We see that the main cost in the periodic auction is the cost of inserting an incoming order into the sorted table. This increases as the size of the table increases. This looking at Table 5 we see that if in a given period we have a maximum of ten buy and ten sell orders, then the period length could be set to $2 \cdot 10 \cdot 0.002 = 0.040$ seconds (in the case of the two party setting), which would equate to a throughput (just considering this part of the computation) of $20/0.040 = 500$ orders per second. However, this is the best throughput one could achieve; the higher the number of orders per second the more processing is needed to sort them, and hence the less orders one can process. This leads us to consider the more naïve volume matching methodology, which we turn to in the next section.

4.3 Volume Matching Method

Volume match (a scheduled cross at some fixed time point, where buy and sell orders are matched only on volume, with no price information considered) is the simplest matching algorithm we consider. As far as we are aware, there is no prior work attempting a "secure" implementation of volume match. This is perhaps because the algorithm is too simple to be of interest from the theoretical perspective of auction mechanism design, since there is no price discovery process. However, despite (and, perhaps, because of) its simplicity, volume match (or scheduled cross) has been used consistently in real-world dark markets for more than thirty years: from the original Posit and Instinet platforms, where crosses occurred once, or several times, per day; to current incarnations such as Posit Match and LSE's Turquoise, where crosses occur at intervals between 10 and 45 seconds. Therefore, despite the lack of prior work in this area, we believe a secure MPC implementation of a volume match algorithm has potential for significant real-world impact.

As explained in the introduction, our final method we use is a volume matching algorithm. This allows us to dispense with the complex decision making related to prices of bids and offers in the previous two methods. In these dark market auctions the price of all completed trades does not depend on any input price by the bidders, but is simply taken from an external source; usually the price on an associated lit market.

Again, we take as input the set of orders, but this time we are only interested in the specific volume to be bought or sold, and each party only enters their respective volume for this period of trading. At the end of the computation, we want all buy and sell orders that can be matched to be opened. Matching is done on a first-come-first-served basis. If a party's order is output as zero, or less than the order's original volume, then this is the amount which gets satisfied.

The algorithm is presented in Figure 6. One can immediately see it is much simpler, and the number of operations are a deterministic function of N and M (unlike prior methods). Indeed, the algorithm requires $2 \cdot (N + M) + 1$ secure comparisons, $2 \cdot (N + M) + 1$ secure multiplications, plus some secure additions.

Algorithm for Clearing a Dark Market

On input of $\langle s_1 \rangle, \ldots, \langle s_N \rangle$ and $\langle b_1 \rangle, \ldots, \langle b_M \rangle$ the algorithm computes (we assume N and M are known)

1. $\langle S \rangle \leftarrow \sum_{i=1}^{N} \langle s_i \rangle$.
2. $\langle B \rangle \leftarrow \sum_{j=1}^{M} \langle b_j \rangle$.
3. $\langle f \rangle \leftarrow [S] > [B]$.
4. $\langle T \rangle \leftarrow \langle f \rangle \cdot (\langle B \rangle - \langle S \rangle) + \langle S \rangle$.
   [Note $\langle T \rangle$ holds the total volume which will be bought/sold in this execution].
5. $\langle L \rangle \leftarrow \langle T \rangle$.
6. For i from 1 to N do
   (a) $\langle z_1 \rangle \leftarrow \langle L \rangle \leq 0$.
   (b) $\langle z_2 \rangle \leftarrow \langle L \rangle < \langle s_i \rangle$.
   (c) $\langle s_i \rangle \leftarrow ((\langle L \rangle - \langle s_i \rangle) \cdot \langle z_2 \rangle + \langle s_i \rangle) \cdot (1 - \langle z_1 \rangle)$.
   (d) $\langle L \rangle \leftarrow \langle L \rangle - \langle s_i \rangle$.
7. $\langle L \rangle \leftarrow \langle T \rangle$.
8. For j from 1 to M do
   (a) $\langle z_1 \rangle \leftarrow \langle L \rangle \leq 0$.
   (b) $\langle z_2 \rangle \leftarrow \langle L \rangle < \langle b_j \rangle$.
   (c) $\langle b_j \rangle \leftarrow ((\langle L \rangle - \langle b_j \rangle) \cdot \langle z_2 \rangle + \langle b_j \rangle) \cdot (1 - \langle z_1 \rangle)$.
   (d) $\langle L \rangle \leftarrow \langle L \rangle - \langle b_j \rangle$.
9. Open $\langle s_i \rangle$ and $\langle b_j \rangle$ for all i and j.

Figure 6. Algorithm for Clearing a Dark Market

As an example, consider the values $s_1 = 3, s_2 = 4, s_3 = 5$ and $b_1 = 5, b_2 = 2, b_3 = 11, b_4 = 1$. Then, we see the following trace will be executed, resulting in sellers 1 to 3 selling all their items, while only buyers 1 and 2 manage to get exactly what they want. Buyer 3 has to make do with only obtaining five units and buyer 4 receives none.

- $S = 3 + 4 + 5 = 12$
- $B = 5 + 2 + 11 + 1 = 19$
- $f = 0$
- $T = S = 12$
- $L = 12$, $i = 1$, $z_1 = 0$, $z_2 = 0$, $s_1 = 3$
- $L = 9$, $i = 2$, $z_1 = 0$, $z_2 = 0$, $s_2 = 4$
- $L = 5$, $i = 3$, $z_1 = 0$, $z_2 = 0$, $s_3 = 5$
- $L = 12$, $j = 1$, $z_1 = 0$, $z_2 = 0$, $b_1 = 5$
- $L = 7$, $j = 2$, $z_1 = 0$, $z_2 = 0$, $b_2 = 2$
- $L = 5$, $j = 3$, $z_1 = 0$, $z_2 = 0$, $b_3 = 5$
- $L = 0$, $j = 4$, $z_1 = 1$, $z_2 = 1$, $b_4 = 0$

Experimental Results Again we performed experiments for different values of N and M, with random buy and sell bids in the range $[1, \ldots, 100]$. Our results are given in Table 8 for our two cases. The algorithm is relatively simple to analyse and it is clear that the number of multiplications and comparisons in one execution is given by $2 \cdot (N+M)+1$, and the number of rounds of communication is given by $9 \cdot (\max(M, N)+1)$. The online latency of this methodology is basically given by the online phase time t_o, whilst the throughput in terms of number of bids processed per second is given by $(N + M)/t_o$. As we can see from the table, the throughput remains fairly constant, with any variation due to our experimental setup. Importantly, the online throughput we are able to achieve are close to what one would need in a real market.

Table 8. Volume Matching Run Times

N	M	Rounds	Pre-Processed Data (m,b)	Two Party Case			Three Party Case		
				Offline Time	Online Time	Online Throughput	Offline Time	Online Time	Online Throughput
10	10	99	(4961,4305)	4.623	0.010	2000	0.100	0.018	1111
10	20	189	(7381,6405)	6.879	0.015	2000	0.159	0.029	1034
20	20	189	(9801,8505)	9.134	0.020	2000	0.193	0.037	1081
20	40	369	(14641,12705)	13.645	0.041	1463	0.245	0.057	1052
50	50	459	(24321,21105)	22.667	0.045	2222	0.484	0.090	1111
50	100	909	(36421,31605)	33.944	0.087	1724	0.690	0.146	1027
100	100	909	(48521,42105)	45.221	0.099	2020	0.906	0.182	1098
100	200	1809	(72721,63105)	67.775	0.158	1898	1.233	0.287	1045
200	200	1809	(96921,84105)	90.330	0.239	1673	1.766	0.360	1111
200	400	3609	(145321,126105)	135.439	0.344	1744	2.508	0.583	1029
500	500	4509	(242121,210105)	225.656	0.500	2000	4.136	0.900	1111

5 Conclusion

We have presented the first full MPC proof-of-concept implementation and analysis of three common auction types used for dark pool financial trading venues. For each of the three algorithms presented, the two-party case has higher online throughput than the three-party case; but lower offline throughput. However, given that most primary markets open approximately eight hours per day, five days a week, and dark markets tend to follow similar opening hours, a real-world implementation would have plenty of time to perform these offline computations. Indeed, offline computations can also be performed in parallel. Therefore, the offline times presented are *not* a limiting factor in algorithm performance.

The two-party case is fully secure as long as *at least one* party is trustworthy. Therefore, we suggest a practical implementation of the two-party algorithms presented would have the venue operator as party one, and the regulator as party two (i.e., the FCA in UK, ESMA in EU, or SEC in USA). Thus, as the operator is unable to view order data in the clear, the algorithms *guarantee* no information misuse (such as that previously perpetrated by Pipeline [27], Liquidnet [28], and ITG [29]). In addition, as the regulator is also involved in the computation, this architecture ensures that a venue operator cannot illicitly vary the rules of the published trading mechanism (e.g., by using undisclosed order types that favour certain traders, such as perpetrated by UBS [30]).

To avoid abusive practices, regulation has attempted to encourage dark pool venues to concentrate on natural liquidity flow (volume investors looking to trade large positions with minimum market impact); and it appears that there has been some success. In Feb 2017, mean trade size and number of trades on European dark pools ranged from Liquidnet's large volume, negotiated matching platform (50 trades/hour; mean size €900,000) to UBS's mid-point matching MTF (16,000 trades/hour; mean size €8,000) [15]. Following the introduction of MiFID II in Jan 2018, the average trade size on European dark markets almost doubled in the first six months [22]. As trade size is negatively correlated with order flow, this suggests the throughput (orders per unit time) of dark venues has fallen.

Considering this, when viewed from a practical perspective, the maximum throughput of the three MPC algorithms presented in this paper are promising: the CDA can execute approximately 10-50 orders per second (depending on order book depth); the periodic auction, running every second, can clear approximately 20 orders on either side; while the scheduled cross volume match can execute 800 orders per second. For a real-world dark pool with relatively large minimum order size, these results suggest that MPC is ready for a practical implementation in financial markets, particularly if a simple matching mechanism is used (e.g., the volume match algorithm). Large investors are often prepared to sacrifice timeliness for privacy. Therefore, the privacy-preserving security guarantee an MPC dark pool provides may tempt enough users for it to be viable; even if executions times are longer than rival venues.

Acknowledgements

This work has been supported in part by ERC Advanced Grant ERC-2015-AdG-IMPaCT, by the Defense Advanced Research Projects Agency (DARPA) and Space and Naval Warfare Systems Center, Pacific (SSC Pacific) under contract No. N66001-15-C-4070, and by the FWO under an Odysseus project GOH9718N.

References

1. Aly, A., Keller, M., Orsini, E., Rotaru, D., Scholl, P., Smart, N.P., Wood, T.: SCALE and MAMBA documentation (2018), https://homes.esat.kuleuven.be/~nsmart/SCALE/Documentation.pdf
2. Bogetoft, P., Damgård, I., Jakobsen, T., Nielsen, K., Pagter, J., Toft, T.: A practical implementation of secure auctions based on multiparty integer computation. In: Di Crescenzo, G., Rubin, A. (eds.) Financial Cryptography and Data Security. FC 2006. No. 4107 in Lecture Notes in Computer Science, Springer, Berlin, Heidelberg (2006), https://doi.org/10.1007/11889663_10
3. Bouchaud, J.P., Bonart, J., Donier, J., Gould, M.: Trades, Quotes and Prices: Financial Markets Under the Microscope. Cambridge University Press, Cambridge, UK (2018), https://doi.org/10.1017/9781316659335
4. Bouchaud, J.P., Farmer, J.D., Lillo., F.: How markets slowly digest changes in supply and demand. In: Hens, T., Schenk-Hoppe, K. (eds.) Handbook of Financial Markets: Dynamics and Evolution, pp. 57–156. Elsevier: Academic Press (2008)
5. Catrina, O., de Hoogh, S.: Improved primitives for secure multiparty integer computation. In: Garay, J.A., Prisco, R.D. (eds.) SCN 10: 7th International Conference on Security in Communication Networks. Lecture Notes in Computer Science, vol. 6280, pp. 182–199. Springer, Heidelberg, Germany, Amalfi, Italy (Sep 13–15, 2010)
6. Chida, K., Genkin, D., Hamada, K., Ikarashi, D., Kikuchi, R., Lindell, Y., Nof, A.: Fast large-scale honest-majority MPC for malicious adversaries. In: Shacham, H., Boldyreva, A. (eds.) Advances in Cryptology – CRYPTO 2018, Part III. Lecture Notes in Computer Science, vol. 10993, pp. 34–64. Springer, Heidelberg, Germany, Santa Barbara, CA, USA (Aug 19–23, 2018)
7. CME Globex: Marketing brochure. https://www.cmegroup.com/globex/files/globexbrochure.pdf (2018)
8. Damgård, I., Pastro, V., Smart, N.P., Zakarias, S.: Multiparty computation from somewhat homomorphic encryption. In: Safavi-Naini, R., Canetti, R. (eds.) Advances in Cryptology – CRYPTO 2012. Lecture Notes in Computer Science, vol. 7417, pp. 643–662. Springer, Heidelberg, Germany, Santa Barbara, CA, USA (Aug 19–23, 2012)

9. Farmer, J.D., Gerig, A., Lillo, F., Waelbroeck, H.: How efficiency shapes market impact. Quantitative Finance 13(11), 1743–1758 (2013), https://doi.org/10.1080/14697688.2013.848464
10. Financial Conduct Authority: Periodic auctions (June 2018), https://www.fca.org.uk/publications/research/periodic-auctions
11. Gresse, C.: The effect of crossing-network trading on dealer market's bid-ask spreads. European Financial Management 12(2), 143–160 (2006), https://doi.org/10.1111/j.1354-7798.2006.00314.x
12. Harris, L.E.: Order exposure and parasitic traders (1997), unpublished working paper, University of South Carolina, Los Angeles, CA
13. Jutla, C.S.: Upending stock market structure using secure multi-party computation. IACR Cryptology ePrint Archive (2015), https://eprint.iacr.org/2015/550
14. Keller, M., Rotaru, D., Smart, N.P., Wood, T.: Reducing communication channels in MPC. In: Catalano, D., De Prisco, R. (eds.) SCN 18: 11th International Conference on Security in Communication Networks. Lecture Notes in Computer Science, vol. 11035, pp. 181–199. Springer, Heidelberg, Germany, Amalfi, Italy (Sep 5–7, 2018)
15. LiquidMetrix: Guide to European Dark Pools - February 2017. Intelligent Financial Systems Limited. https://www.liquidmetrix.com (Feb 2017)
16. MacKenzie, D.: A sociology of algorithms: High-frequency trading and the shaping of markets (June 2014), http://www.sps.ed.ac.uk/__data/assets/pdf_file/0004/156298/Algorithms25.pdf, Univ. of Edinburgh, Working Paper.
17. Massacci, F., Ngo, C.N., Nie, J., Venturi, D., Williams, J.: FuturesMEX: Secure, distributed futures market exchange. In: IEEE Symposium on Security and Privacy. pp. 335–353. San Francisco, CA (May 2018), https://doi.org/10.1109/SP.2018.00028
18. P. Bogetoft et al.: Secure multiparty computation goes live. In: Financial Cryptography and Data Security. FC 2009., Lecture Notes in Computer Science, vol. 5628. Springer, Berlin, Heidelberg (2009), https://doi.org/10.1007/978-3-642-03549-4_20
19. Partisia: Secure order matching. Webpage (2018), https://partisia.com/order-matching
20. Petrescu, M., Wedow, M.: Dark pools in European equity markets: emergence, competition and implications. European Central Bank: Occasional Paper Series, No. 193 (Jul 2017), https://www.ecb.europa.eu/pub/pdf/scpops/ecb.op193.en.pdf
21. Redburn Execution: List of approved trading venues EMEA (28 Sep 2018)
22. Reid, H.: Light or dark? Six months on, MiFID 2 rules divide equity traders. Reuters Business News (29 June 2018), https://www.reuters.com/article/us-eu-markets-mifid-analysis/light-or-dark-six-months-on-mifid-2-rules-divide-equity-traders-idUSKBN1JP0LP
23. Stein, K.M.: Commissioner's statement on adoption of rules to increase the operational transparency of Alternative Trading Systems (ATS). U.S. Securities and Exchange Commission, Public Statement (18 July 2018), https://www.sec.gov/news/public-statement/statement-stein-071818-1
24. Thorpe, C., Parkes, D.C.: Cryptographic combinatorial securities exchanges. In: Dingledine, R., Golle, P. (eds.) Financial Cryptography and Data Security. FC 2009., Lecture Notes in Computer Science, vol. 5628. Springer, Berlin, Heidelberg (2009), https://doi.org/10.1007/978-3-642-03549-4_18
25. Thorpe, C., Parkes, D.C.: Cryptographic securities exchanges. In: Dietrich, S., Dhamija, R. (eds.) International Conference on Financial Cryptography and Data Security. FC 2007. pp. 163–178. No. 4886 in Lecture Notes in Computer Science, Springer, Berlin, Heidelberg (2007), https://doi.org/10.1007/978-3-540-77366-5_16
26. Thorpe, C., Willis, S.R.: Cryptographic rule-based trading. In: 16th International Conference on Financial Cryptography and Data Security: FC 2012. Lecture Notes in Computer Science, vol. 7397, pp. 65–72. Bonaire (Feb 2012), https://fc12.ifca.ai/pre-proceedings/paper_74.pdf
27. United States of America before the Securities and Exchange Commission: In the matter of Pipeline Trading Systems LLC, et al., Exchange Act Release No. 65609. https://www.sec.gov/litigation/admin/2011/33-9271.pdf (24 Oct 2011)
28. United States of America before the Securities and Exchange Commission: In the matter of Liquidnet, Inc., Exchange Act Release No. 72339. https://www.sec.gov/litigation/admin/2014/33-9596.pdf (6 Jun 2014)
29. United States of America before the Securities and Exchange Commission: In the Matter of ITG Inc. and Alternet Securities, Inc., Exchange Act Release No. 75672. https://www.sec.gov/litigation/admin/2015/33-9887.pdf (12 Aug 2015)
30. United States of America before the Securities and Exchange Commission: In the Matter of UBS Securities LLC, Exchange Act Release No. 74060. https://www.sec.gov/litigation/admin/2015/33-9697.pdf (15 Jan 2015)
31. United States Securities and Exchange Commission: SEC institutes enforcement action against 20 former New York Stock Exchange specialists alleging pervasive course of fraudulent trading. Press Release, https://www.sec.gov/news/press/2005-54.htm (12 April 2005)

32. Yadav, Y.: Insider trading and market structure. UCLA Law Review 63, 968–1033 (2016)
33. Zhu, H.: Do dark pools harm price discovery? Review of Financial Studies 27(3), 747–789 (Mar 2014), http://dx.doi.org/10.1093/rfs/hht078