
**Market data encryption
optimization using GPU; based
on pre-defined granular
permissions**

**B. A. J. Karunaratne
2018**



Market data encryption optimization using GPU; based on pre-defined granular permissions

A dissertation submitted for the Degree of
Master of Science in Information Security

B. A. J. Karunaratne
University of Colombo School of Computing
2018



Declaration

The thesis is my original work and has not been submitted previously for a degree at this or any other university/institute.

To the best of my knowledge it does not contain any material published or written by another person, except as acknowledged in the text.

Student's Name: B. A. J. Karunarathne

Signature:

Date: 14.07.2018

This is to certify that this thesis is based on the work of Mr. Binoy Karunarathne under my supervision. The thesis has been prepared according to the format stipulated and is of acceptable standard.

Certified by:

Supervisor Name: Dr. Chamath Keppitiyagama

Signature:

Date: 14.07.2018

Abstract

Information security research and literature surveys has generally followed the views from practice where users are frequently seen as a weak link in the implementation of information security measures and their role in the overall security of the system should be maximized if possible since the dependency on data security of the business is vital to any type of organization.

Trade messages are the core of the stock market systems and it is essential to transmit the message with managing the delivery of trading applications and keeping latency low. Because of the low latency fact, currently the security of the messages are not considered and the message can be received by any sender and a receiver if they correctly connected to the trading gateways.

This research has taken a broad approach into the trading messages related security in a market data system, from the initial conceptualization of the security approaches through to the in depth security mechanism that can be gain by implementing the given solution. The results derived from this research show that there is a great need for focuses on the transmission and the transition of the market data from a secure perimeter to another secure perimeter. This thesis focuses on the secure mechanisms that can be applied on the market data in order to achieve the information security requirements within systems development.

After critically evaluating three candidate symmetric and asymmetric algorithms, three solutions were developed and then using qualitative and quantitative analytical methods, most suitable cryptographic solution has been presented for exchange system.

Acknowledgment

I would like to thank Dr.Chamath Keppitiyagama for serving as my mentor during the one year process of my study and for his comments guidance during earlier drafts of this thesis and the comments that were given while developing programs for providing this research process.

I would like to thank my family members specially my wife, Deshani Siyambalapi-tiya for the patience, attention and care that she gave me. I will not be able to achieve my Master of Science degree without her unconditional support.

Also a special thanks to Isura Nirmal, for his guidance that he gave me while I writing my thesis. He was humble enough to give his fullest support and comments whenever needed.

List of Abbreviations

3DES	Triple Data Encryption Standard.
AES	Advanced encryption Standard.
AMQP	Advanced Message Queuing Protocol.
ASCII	American Standard Code for Information Interchange.
ASIC	Application Specific Integrated Circuits.
DES	Data Encryption Standard.
FAST	FIX Adapted for STreaming.
FIX	Financial Information eXchange.
FPGA	Field Programmable Gate Arrays.
GPU	Graphics Processing Unit.
LDAP	Lightweight Directory Access Protocol.
MiB	Mebibyte.
MOM	Message Oriented Middleware.
NIST	National Institute of Standards and Technology.
POA	Point of Arrivals.
POD	Point of Departure.
PTX	Parallel Thread Execution.
XML	Extensible Markup Language.

Table of Contents

Abstract	i
Acknowledgement	ii
List of Abbreviations	iii
Table of Contents	iv
List of Figures	vi
List of Tables	vii
Introduction	1
1.1 Prolegomena	1
1.2 An introduction to the trading platforms	1
1.3 Proposed program aspect at glance	3
1.4 Functionalities of a trading platforms	4
1.4.1 Trade message dissemination process	4
1.4.2 Highlighted Performance Benchmarks	5
1.5 Overview of the thesis	6
1.6 Objectives Of this Thesis	7
1.6.1 Research Methodology	9
1.6.2 Empirical Study	9
1.7 Thesis Structure	10
Literature Review	11
2.1 Introduction	11
2.2 Current Trading System Implementation	11
2.3 Market Data Message Categories	12
2.4 Current Message Transmission Protocol Architecture	13
2.4.1 Performance of IBM MQ Technology	14
2.5 Security Policy Background Analysis	15
2.6 Security Overview of the Current Implementation	16
2.7 Granularity Overview of the Current Implementation	17
2.8 Objectives of an Adversary and Possible Attack Types	18
2.9 Threat Model	19
2.10 Information Security vs Adversary	20
2.11 Adversary and Attack Vectors	21
2.12 Evaluated and Achieved Objectives	22
2.13 Selection of Algorithms	23
2.14 Qualitative Analysis	24
2.15 Quantitative Analysis	24

Identify the attack types and attack vectors	25
3.1 Introduction	25
3.2 Identified attack types in this domain	25
3.2.1 Attack Types	25
3.2.2 Attack Vectors	26
3.3 Outcomes of a Successful Insider Attack	27
Implementation of Crypto System	29
4.1 Introduction	29
4.2 Test Environment Preparation	29
4.2.1 Order Matching Process	30
4.2.2 Test Environment Hardware Specification	31
4.3 RSA Implementation in Depth	32
4.3.1 RSA Pseudocode	33
4.4 AES Implementation in Brief	35
4.5 Triple DES Encryption in Brief	36
Performance Results	37
5.1 Introduction	37
5.1.1 Results analysis	37
5.1.2 RSA Encryption-Decryption Results	48
5.1.3 AES Encryption-Decryption Results	49
5.1.4 3DES Encryption-Decryption Results	50
5.2 Thread Based Performance Analysis using Amdahl's Law	53
5.2.1 Performance Consideration	53
5.2.2 Possible limitations in Amdahl's law based calculations	54
Evaluation of Results	56
6.1 Introduction	56
6.2 Vertical Analysis	57
6.3 Horizontal Analysis	57
6.3.1 Plain Text Encryption Process Flow	58
6.3.2 Cipher Text Decryption Process Flow	59
Future Work	60
Executive Summary	62
References	66
Appendix A. Class Diagram	70
Appendix B. Architecture Diagram	72
Appendix C. Data Flow Model	74
Appendix D. Sequence Diagram	76
Appendix E. Terms related to Stock Markets	78
Appendix F. RSA based solution	80

List of Figures

1.1	Proposed changes in current gateway map	8
2.2	Current Granularity categories	17
2.3	Possible attack regions	18
2.4	Example FIX message	20
2.5	Possible attack vectors	21
4.6	Order matching process	30
4.7	AES Implementation process flow	35
5.8	Average order execution on GPU and CPU	38
5.9	Average order execution rate in the current system	39
5.10	Average order execution rate in the RSA based system	40
5.11	Average order execution rate in the AES based system	41
5.12	Average order execution rate in the 3DES based system	42
5.13	Average data execution time of the trade data	43
5.14	Relationship between the data encryption rate and the time	44
5.15	Relationship between Resource usage and the time	45
5.16	Trade file execution rate in different time of the day	46
5.17	Encryption time for a fix sized trade file	47
5.18	Encryption-decryption rate of RSA based system	48
5.19	Encryption-decryption rate of AES based system	49
5.20	Encryption-decryption rate of 3DES based system	50
5.21	Amdhal's performance in different system specifications	54
6.22	Encryption process of the plain text data	58
6.23	Decryption process of the encrypted data	59
8.24	Characteristics score diagram of the current system	64
8.25	Characteristics score diagram of the proposed RSA based system	64
8.26	Characteristics score diagram of the proposed AES based system	65
8.27	Characteristics score diagram of the proposed 3DES based system	65
A.1	Highlighted changes in the class diagram	71
B.1	Highlighted changes in the Architecture diagram	73
C.1	Highlighted changes in the Data flow diagram	75
D.1	Highlighted changes in the Sequence diagram	77

List of Tables

1.2	Propitiatory gateway's subscription fees	7
2.3	Standard format of a FIX message	20
3.4	Break time of an average 128bit key	28
5.5	Execution rate of the trade messages on current system	39
5.6	Execution rate of the trade messages on RSA based system	40
5.7	Execution rate of the trade messages on AES based system	41
5.8	Execution rate of the trade messages on 3DES based system	42
5.9	Compared Machine details	51
5.10	AES performance in different system specifications	52

Introduction

1.1 Prolegomena

This chapter demonstrates the key points that have been addressed in the thesis by introducing the background of the motivation for the work. This chapter continues with presenting issues analyzed in this thesis, research goals, scope, research methodology and ends with a description of the remaining chapters with an introduction of each of them.

1.2 An introduction to the trading platforms

Since societies step forward from a barter economy to using a common currency as a medium of exchange, the societies have tried to devise systems that allow for rational ways to exchange value between multiple parties. In order to balance the process of making goods and services commensurable the Greek philosopher Aristotle defined four characteristics of a good form of money, durability, portability, divisibility and intrinsic value that help to execute a transaction from what is considered to be “good money”.

As the technology world is evolving into different dimensions, economies of the countries grew in an accelerated rate and the demand for a secure medium of exchange increased due to security, regulations.^[1] The currency exchange in the physical format was not a proper solution for the transactions which are high in both volume and market price. As a solution for those drawbacks, governments created more accessible and available medium of exchange that they could control and regulate. This

Businesses needed more capital to extend their territory in their business area. In order to raise money for further investments, companies either borrow capital from a bank or an individual investor or by selling part of the company by issuing stocks to the public. A collection of companies listed under a stock exchange and interested parties can buy and sell stocks. Matching process of buy and sell orders were previously done on a board manually. Currently with technology evolution, this particular medium of exchange has been executed in multiple venues using pre-programmed software using a brokerage that supports automated trading. The extreme demand is for secure and low latency data communication is leading the system developers, business analysts and solution architects to come up with new means of communication which are faster, cost efficient and widely available. Brokerage companies listed in stock exchange systems to are in vogue that distributed and secure communication is the most crucial factors that

need to be considered when multi party communication done. The primary goal of the stock exchange and stock brokerage companies are to facilitate the message dissemination of decentralized topologies without occurring a single points of failure at the middle of a trading day or a trading session, enabling fault tolerance and high availability. These characteristics make them the dominant trading solutions for do the stock transactions within distributed architectures.

However, a major problem with this kind of application is to securely transmit generated messages from the point of origin to the point of destination where the clearing process of the trade is finalizing. As the transmission part of this implementation, MOM is one of the key technologies in financial market data transmission process. Market data transmission using AMQP, is an open-standard for communication over MOM. This thesis intended to determine the ability to encrypting the trade message before connecting the message gateways to the MOM. In order to observe how encrypting the trade messages process performs in a real-life scenario, it was necessary to identify the existing architectures of the trade message delivery system. All the evaluations are performed over 1 Gigabit Ethernet assuming that the data rate over the channel was constant throughout the durations of executing the experiments.

In a typical message queue consists of a queue manager and an application to connect to the local bindings and client connections. There is a mechanism to do a connection authorization using LDAP. IBM MQ[®] uses public key techniques to protect data while traveling across a network. There is a channel authentication mechanism and it can be used to block invalid connections according to predefined filter rules.

1.3 Proposed program aspect at glance

Standardization development of a trading platform from the vendors perspective is a mandatory task since the connections to third party clients and message dissemination through gateway must properly align with legacy systems and existing systems. In this thesis, the efficiency of GPUs with a limited number of stream processors which were located in general use desktops and laptops were examined and the execution of algorithms were monitored with AES, RSA and Triple DES algorithms. For the implementation part of the changes in encryption algorithms were done using Microsoft Visual Studio 2015 and Code Blocks Release 17.12 , the OpenCL framework was used under OS X. The graphic cards tested were Nvidia® GTX 750 TI, NVIDIA GeForce® GT 755M, AMD Radeon® HD 6700M, AMD® RX 480, ZOTAC® 980TI Amp Extreme edition, NVIDIA®Tesla K40 12GB GPU Accelerator. The NVIDIA®Tesla K40 12GB GPU Accelerator was only used for test the selected encryption algorithm's maximum performances. Also GPU-Z 2.8.0 tool was used in order to collect the hardware utilization details while conducting the test.

GPU based solutions are very efficiently and can replace or assist CPU computations. At the testing level an average throughput of encryption rate 4.564 MB/s for RSA, 2.135 MB/s for AES and 2.655 MB/s for 3DES was achieved. Encryption and decryption results showed an acceptable output after using GPU based message encryption protocol. The combined execution of encryption and decryption on the GPU can improve security of the trade message by using 128 bit size key. The detailed analysis of the outcome has presented in the next chapters. Previous latency of a trade message dissemination from the system was 72000 μ s and there was a reduction in the latency caused by data transfers between CPU and GPU after implementing the crypto system. Since the average latency after encrypting the trade message was around 96000 μ s which is still can be considered as an acceptable latency. After conducting this experiment on a system with heterogeneous architectures of systems with GPUs and with different multi-core CPUs, contrary to expectations, this study did not find any significant drawbacks of using an encryption mechanism on trade message dissemination process in order to replace the existing mechanism of using a secure channel to connect secure parameters.

1.4 Functionalities of a trading platforms

Stock market software can be developed by different software vendors. Distributed processing, parallel processing, asynchronous high speed logging and spatial subscription can be categorized as subsections of trading system's performance matrices. Fault tolerance, system replay are the measurements that can be taken to measure system reliability. Distributed processing, parallel processing and partitioning are the sub categories of scalability measures of the system. In Current implementation, trade message security has defined from the transport layer. Protocol should provide confidentiality, inability of adversaries to alter the contents of the communication, inability of adversaries to impersonate as the sender or the receiver, can not inject new data during communication.

1.4.1 Trade message dissemination process

In order to overcome these issues, most of the stock exchanges are using IBM MQ[®] and Simple Object Access Protocol as there solution for message negotiation and transmission. There is a common agreement about critical activities related benchmarks between the clients and the trading system. In the system requirement specification document issued by the software company with related to the proposed trading system, these benchmarks had been clearly defined and it has been taken into consideration when analyzing the crypto program performance.

Parallel processing is an important terms since this thesis proposed the solution based on trade messages and its security maximization. Partitioning process of the system is also an important task since proper load balancing of the system can be gained by that. Proper resource utilization can lead a system to behave in a predictable manner. Dynamic schema, programmable components, event driven, distributed processing, regression testing, spatial subscription can consider as flexibility factors. These component will determine the system's ability to adopt to changes that are made after a system go-live. Off the shelf machines, experiments before actual development, reduce code redundancy are some of the actions that can be taken to minimize cost. Trading system can be classified according to various factors, including company size and company sector. Current market price of the stocks multiplied by the number of shares of stock measures the market value of the trading system, which is one measure of the company's size.

1.4.2 Highlighted Performance Benchmarks

Following benchmarks[2] were taken as key points to measure a trading system's performance.¹

- 1,000,000 - 200,000,000 Orders per Day
- 7,000 - 12,000 orders per second
- Peak order rate 18,000 - 24,000
- 2,750 accepted/ rejected trades per second
- Average order latency under 2,400 μs
- Average trade latency 72,000 μs
- Average market data dissemination latency of 720 milliseconds
- Estimated database size for 100,000 trades (0.25 GB)
- Settlement² is due in (T+3) period³
- Trade message will be considered as invalid after (T+3) days

¹This figures can vary according to the Service Level Agreement between the trading system developers and users. The benchmark has been taken as general acceptable ratios at the time of the submission of this thesis

²Settlement of a trade is the business process whereby ownership of the trade is transferred according to the buy party and the sell party, usually according to the payment of money

³T is the trade executed day and 3 is the number of days after execution

1.5 Overview of the thesis

In this master thesis, the possibility of improving the performance of the dissemination process of trade messages after applying a security mechanism based on encryption method has been addressed. Graphical Processing Units have become a vital component of current computational systems. Progress of GPU technology has been developing in an accelerated rate and it has become an achievable target because of the stagnation in traditional CPU clock speed. Scientists have started focusing on developing more powerful computation throughput for general purpose computing. In the past decade, there has been a significant improvement in the GPU performance and programmability. It has made GPU appropriate for certain types of application specific characteristics with large computation requirements and considerable parallelism.

The target in this research is to study, analyze and compare three different encryption algorithms' performance and then to design and make an implementation of an encryption system to encrypt FIX protocol based trade messages with GPU computation. A trade message has to be kept in the system until the process of updating the accounts of buy and sell trading parties and preparing the transfer of money and securities. This process is simply known as the clearing⁴ process of a trade. After a trade is being cleared, the exchange process of money between the parties of a trade on the settlement date after agreeing as earlier on the trade is known as settlement⁵. In this thesis, trades settled in 3 business days were considered and the security implications were tested and benchmarked. A successful attempt made by adversary to decrypt a trade message or to reveal secrecy on an encrypted trade message after 3 days from the point of origin considered as a successful security implementation on trade message since the trade message expiration time has reached and the message has become an invalid message after 3 days.

As an outcome of this thesis, the performance between GPU implementation and CPU implementation has been compared in order to determine the mechanisms of improving the performance of public key algorithms. Finally, the generalisability of these results have been critically evaluated and presented the optimal mechanism to transmit trade messages between communication parties after being encrypted.

Furthermore, in this research; the performance of algorithms has been compared between the GPU implementation and the CPU implementation. The comparison used to graphically represents the outcome of the experiment to demonstrate that the proposed solution is feasible and achievable with the available technology and hardware component.

⁴Clearing process of a trade represents all processes from the time a trade was ready for a transaction until that trade is settled by the parties

⁵Settlement of a trade is the business process whereby ownership of the trade is transferred according to the buy party and the sell party, usually according to the payment of money

1.6 Objectives Of this Thesis

This thesis was based on a multiple message transmission mechanism developed by different trading system vendors in order to standardization the existing message protocols into single message protocol after encrypting the content of these messages. From the various contributions added by experts in the trading platform related software development area, have given a solution to transmit fragmentary trade messages through different gateways according to one of the following properties.

- Importance of the field of the message
- Latency factor
- Relevance
- Message type

1. There is an additional charges due to the usage of message gateways and the thesis addressed the assignment to reduce the third party gateway service chargers which is recurring as an annual fee.

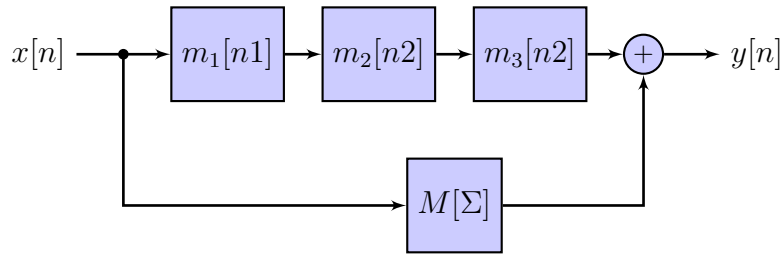
Due to security purposes while using a secure pipeline to do the message transmission, the fragmented messages are disseminating through different proprietary gateways after being prioritize information and it definitions. After a successful implementation of the crypto system, the number of gateways that are being used to transmit market data can be reduced to a less number.

When connecting the trading system into these gateways, a service charge will be added. Following table shows some of the service charges that a broker has to pay after connecting into proprietary trading gateways (Tab 1.2).

Name of the gateway	Subscription Fee
Light Speed Trading(https://www.lightspeed.com/)	
X_Trader	\$ 800.00 Per Month
X_Trader Pro	\$ 1700.00 Per Month
X_Risk	\$ 500.00 Per Month
TT Multibroker Pro	\$ 400.00 Per Month + Charges
Deutsche Börse(http://www.xetra.com/)	
ETI ETS Full Session	€500.00 Per Month
T7 EMDI	€3000.00 Per Month

Tab. 1.2: Propitiatory gateway's subscription fees

2. Another objective of this thesis was to reduce the number of gateway connections after introducing a substitute solution to the existence of a middleware message handling functionality. The proposed system must offers relevant functionalities with an acceptable rates that have been benchmarked after considering the standardized architecture for market data system(Fig1.1).
3. Since there are number of connection nodes in a trading management system due to confidentiality, Integrity and Availability of trade data, there is a lack of exerted security control over messages which are disseminating through the trading system. By introducing a security mechanism, the research looked into the possibility of reducing the number of connections that are connected to the trading system by combining multiple trade messages in to a concatenated single message.



Variables

- $x[n]$: Original Trade Message
- m_1 : Chunk of the trade message
- $[n1]$: Gateway for the message transmission of m_1
- m_2 : Chunk of the trade message
- $[n2]$: Gateway for the message transmission of m_2
- m_3 : Chunk of the trade message
- $[n3]$: Gateway for the message transmission of m_3
- M : Message dissemination after encrypted
- Σ : Single gateway with encrypted messages
- $+$: Message Destination for further processes
- $y[n]$: Message Database

Fig. 1.1: Proposed changes in current gateway map

4. Final and the most focused objective of this thesis was how to apply an additional protection layer over the disseminated trade message via the FIX gateway. The terms “Trade Message” and “FIX gateway” set up boundaries around the scope of the research are with a feasible coverage. After achieving the expected results incorporate with the predefined boundary, necessary code level adjustments can be used to tune up the system for other message protocols.

The thesis addressed this issues and proposed a solution for solving each of these identified issues.

1.6.1 Research Methodology

The research approach undertaken for this thesis as follows. Identified functionalities of the developed program should be tested in order to verify whether the proposed system is feasible to connect to the native trading system to apply cryptographic functions. Authentication, Availability, Confidentiality, non repudiation and integrity are the five aspects of information security and all development of the program and results analysis were carried out according to the standard guidelines mentioned under those categories.

The foremost concern in data transmission through a wireless channel is data security and integrity. Many researches have identified various properties that might impact the security of the data while transmitting the data through a medium[3].

1.6.2 Empirical Study

A brief background description of the empirical study that was conducted in order to select the most suitable encryption mechanism and then the empirical material and its findings are presented in next chapters. These findings in-line with different organizations to identify the best security solutions and measures are used to protect the trading data. Endmost results were classified after considering these facts and statics received from the executed set of tests.

1.7 Thesis Structure

This thesis describes the approach to the identified problem and a description of developed cryptographic program in the following chapters. The chapters are organized as follows.

- Chapter 1 presents the motivation, background on trade message encryption system based on GPU computation and the characteristics differentiate proposed solution from existing trade message transmission mechanism.
- Chapter 2 provides an overview of previous research on trade message security and cryptographic researches based on file encryption. The literature review attempted to describe the existing gap between an attack and the security of a trade message which can contain sensitive data by highlighting the most contributive projects a thorough description of reference architecture and justification for the cryptographic algorithms adopted for such systems.
- Chapter 3 describes the system requirements and the architecture of the proposed solution. A brief demonstration of the hardware specification is included in this chapter.
- Chapter 4 describes the implementation details and the technologies chosen in the proposed solution. Identified limitations, comparison of the results and necessary changes in the existing system are also addressed in this section. Three different program developed in 3 different cryptography algorithms were used in this process. All the test results are shown in graphically in this chapter.
- Chapter 5 describes the evaluation tests performed and the corresponding results. The results have been critically evaluated with the existing benchmark and presented the overall advantages and disadvantages.
- Chapter 6 summarizes the work developed and future work.
- Appendix section contains several additional information to the thesis writing such as, classes diagrams and definitions, detailed analysis tables and partial test-results.

Literature Review

2.1 Introduction

Chapter 2 begins by layering out the theoretical and related literature background to the identified research question. Foundation for the proposed functional background is also backing up through this chapter. In addition to that, the research question and the suggested solution has critically evaluated throughout this chapter. To conclude chapter 2, the literature revision and results were obtained to develop the cryptographic program and implement it in practice.

2.2 Current Trading System Implementation

A capital market may not have the resources to do the continuous monitoring of the market data dissemination via an encrypted channel. Algorithmic trading is the process of using a computer program that follows a defined set of instructions for placing a trade order. Identifying profitable trade combinations and place to do the trades in order to generate profits at a speed and frequency is impossible to identify by a human trader. The aim of an algorithmic trading platform is to identify the patterns that can be executed to generate a profit at the minimum cost[4] while all the parties have a clear and understanding view of the process.

The trade messages transmit through a secure or an insecure channel contain sensitive information that only authorized persons should get access[5]. Encrypting the trade messages before sending over a network is important. Therefore in the present situation, securing only the system as an individual entity combined with communication medium is not enough on a communication network[6]. Enabling strong security is an important precaution that needed to be followed. The proposed encryption mechanism on trade messages should present a perfect balance between security and latency. The encryption algorithm embedded application provides a security layer to the instant communication after doing the encryption and at the same time strong security and an acceptable latency can be gained. Users are allowed to select the security and the predefined granularity level[7] that they want and after encrypting the content of this message, they can exchange the trade messages.

2.3 Market Data Message Categories

Stock exchange is a collection of pre trade data, trade data and post trade data. Submitted orders and Order requests send to the exchange system before trades take place. Trading activity related execution reports, trade submit messages, trade rejected messages are trade messages and trade settlement instruction messages are related to post trade actions. Thesis was mainly focused on FIX gateway related Trade message encryption since it basically designed based on the low latency market data dissemination factor. Number of existing stock exchanges[8][9]are using FIX as one of their primary message dissemination gateways. FIX is a self describing protocol used for transmitting stock market trade submit related information[10]. FIX is an ASCII message with a header, body and trailer. In order to check the integrity of Trade message sequence, there is a checksum embedded to message itself[11]. Type of FIX applications messages are as follows.

Once messages are being recorded in the system , automated background processes execute on top of the stored messages. Information received from the process can be categorized into two sub categories according to the client server architecture[12]. The clients of the message service are

- Order management System
- Client side FIX message engine
- Trading Frontend

Server side can be considered as

- Server side FIX message engine
- Order Matching engine
- Trade settlement system

2.4 Current Message Transmission Protocol Architecture

IBM MQ[®] is a middleware for messaging and queuing[13]. Most of the trading platform systems[13] are using IBM MQ[®] in order to transmit messages over the network. This middleware transports trade messages from one destination to another destination. It is secure and reliable since there is a process[14] of an encryption and decryption mechanism.

As an overview of the functionality of IBM MQ[®], a client channel definition table determines the probable network channel definitions and authentication information used by client applications to connect to the queue manager. There is a server hardening process with regard to the setting up client connection to achieve the security. Each of these client connections require the details of the server host name, server channel name, and listening port on the server.

IBM MQ[®] is the standard connection protocol of the trading system and third party clients' software. MQ Channel can be down during a trading period⁶. Also trade message queue can be full during a peak time of the day. IBM MQ[®] related compatibility problems can be occurred between IBM[®] and vendor distributed systems. Changes in the MQ configuration due to bug fixes of the stock exchange system and performance of the MQ protocol can be slowed down and below the expected values due to network, MQ or message broker and there can be problems[15] in connecting to broker's message queue manager applications.

⁶A trading session is a period of time of a business day, from the day opening notification to the day closing notification of a stock exchange.

2.4.1 Performance of IBM MQ Technology

The protocol performance can be limited by physical resources and application design[16]. IBM MQ[®] reduces the number of interconnections between multiple parties and simplify development through separation of integration logic from the applications front end. Because of the powerful publishing and receiving compatibility of the messages through exchange system's matching engines which have been designed by different vendors, real time based information and content to any endpoint can be disseminated. Transformation and validating of messages in any alphanumeric combination of different message formats, including XML, text based, web services and non-XML formats are supported by IBM MQ[®]. Business rules can be configured according to predefined criteria which can match information content and business processes. Business agility improvisation by dynamic configurations of information dissemination and end connection patterns without doing any complex code changes or a reprogramming of end-point applications is also an advantage of using IBM MQ[®] protocol. Access control to securely deliver the order and trade related data at a minimum latency is also a value added advantage of MQ.

Trade message is the ultimate outcome of a matched buy and sell order. There can be number of orders in a trading system at a given period of time. There is a set of predefined requirements according to the regulations and matching algorithm criteria that has been setup by the exchange system that has to be satisfied before it can be matched and execute with another order. Trade message can be disseminate in different formats[17]. Primary purpose of enabling a gateway controller of sending trade message is to receive real-time information on executed trades, to submit a trade for registered member firms. This interface can not be used to submit orders and it can only receive market data. While using IBM MQ[®], following problems can be occurred after an inside attack of network sniffing attack[18].

- Can the trade messages validated to be trusted as not altered by a third party before publishing it to the gateways?
- Who is the publisher of the trade messages? Whether the trading system itself or a third party pretending to be as the trading system.
- Is the publisher of the trade messages properly authorized to publish trade messages via MQ?
- What is the guarantee whether the sender is sending the trade publications to the authorized subscriber?
- Can the receiver be a person group who has hacked the receivers' end systems?
- Who defines the routine of the trade message through the pipeline?

2.5 Security Policy Background Analysis

In a company, a set of clearly defined rules which are directly affected on security is established by creating a security policy. Security policy documents contains rules that to be followed in written guidelines format. The same conditions apply to a stock exchange and the security policy should cover both front end and the back-end of the system. Policy document outline necessary precautions, steps and procedures to follow in the aspiration for security. A policy document outlines specific requirements on information security realm[19] [20]. Two different types of security policies with related to trade message were identified while implementing the crypto system. In order to maintain security standards (ISO27001⁷/ISO27002⁸), it is mandatory to maintain the security policy document denoting all the requirements and implications with relevant to management and security of Stock Exchange data assets.[21]. All these security standards concatenate essential three goals in security perspective.

- Create awareness of security issues with existing exchange system.
- Help exchange system operators and IT security officers design a security policy
- Entrust security mechanisms in order to achieve authentication, availability, integrity and response to security breaches.

While these security standards and the policies have placed significant importance on survivability of trading system while the system is under an attack. This thesis captured existing and contemporary problems that arise when trading system or its sub component are under attack. As a result of an attack or a delay in initiating network connections, there can be unusual activities with regard to trade disseminations process. These scenarios must also be covered by a security policy in order to secure and control data integrity and confidentiality[22]. This thesis was done considering the ordinary security policy points that can be relatively exhibit in every trading system platform developed by different vendors.

There is a specific “System security policy”, which defines how a trading system should be designed in order to support the security benchmark of a trading system. There are important policy documents that have been defined for a trading system. Economical, government regulations, investors’ guidelines are considered when there is an implementation of a security policy on line. Not only a trading system should be covered by these policies but also other information systems can be covered through this security policies after amending minor details. Following three documents must prevail and proper approval should be in position when developing, implementing, upgrading a trading system.

- Information security policy
- IT Policy
- Classification of information sources

⁷ISO27001 establishes requirements for an organization to certify its Information Security Management System (ISMS)

⁸ISO27002 more focused on the individual and provides a code of practice for use by individuals within an organization

2.6 Security Overview of the Current Implementation

Implementing security measures in the trade message transmission process is not a task that can be finished within a predefined period of time. It is a continuous process that should be a primary task of market data system recurrence tasks list. In a trading management system, task of the message transmission related activities monitoring is an infeasible task. Although a particular physically measurable parameter is secured from the attackers, the security and the secrecy of the trade message can not be guaranteed through out a predefined period of time[23]. Thus the security goal should be constant improvement against the attacks that can be occurred unexpectedly. The improvisation task of the secure trade message dissemination starts with understanding the risks and the threat landscape. Adversaries, objectives, and how these adversaries carry out their attacks. Identifying the risk and relate adversaries is a mandatory precaution that has to be taken in to consideration.

A company can be targeted and affected by a ransomware outbreak due to an unknown attack which can be originated from any destination. Motivation to the attack can also be vary by instance to instance. Affected company has to deal with a either a motivated targeted attack or an unintentional attack. But ransomware can be used as a cover to hide more chained activities[24]. In the market data system, clients and the clearing parties are interchanging message using a message protocol. Software architects are aware of the potential threats to an organization's technical infrastructure. As a common approach, identifying the threats can be identified which are significant in beefing up organizations' network security[25]. Implementing of a cryptosystem in a trading system is always a trade off between security, latency and efficiency. As mentioned in a previous topic, there are benchmarks that have to be maintain in order to get expected performance from a system.

All these security models are depending of an initial trust that is "completely" secure inside a clearly defined boundaries. The initial trust level is always hard to protect while safeguarding the invisible boundary lines. There is no guarantee that the interconnected systems which are with well defined secure boundaries will always secure since an attacker might have the ability to sniff through the communication pipeline due to the dependencies in parameter level security description.

To solve this secure transmission problem in a feasible way that the implementation can be done effectively, three prototype were developed. After doing a thorough statistical and functional set of analysis, RSA based mechanism has presented as the solution for the research questions. Next few chapters contain the detailed analysis on how the RSA based solution was derived after critically evaluating the results which were taken after conducting test with a wide range of possible objectives which were subjected to a number of constraints.[26]

2.7 Granularity Overview of the Current Implementation

As shown in the Fig 2.2, need of the granularity level based encryption arose due to the following requirement in a trading platform. A clearing member in the trading system can have a number of trading members and subsequently clients who are trading via the trading members.

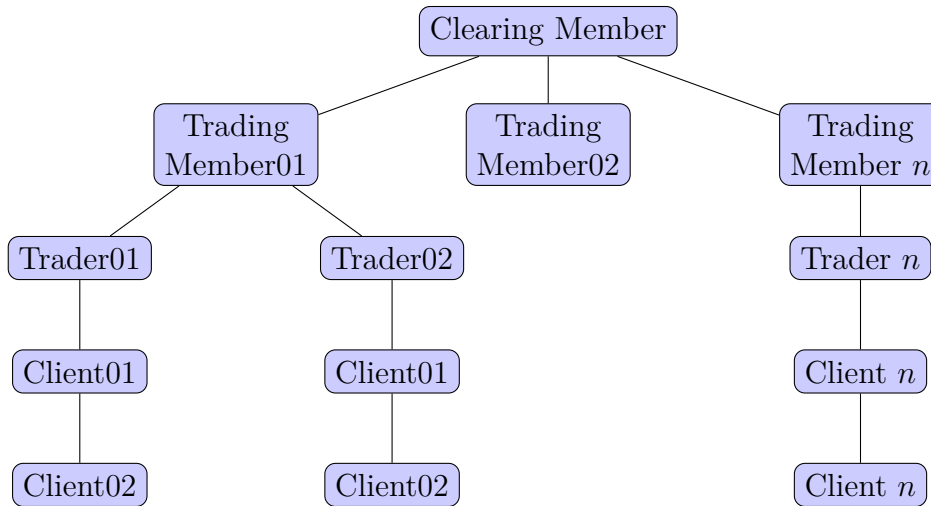


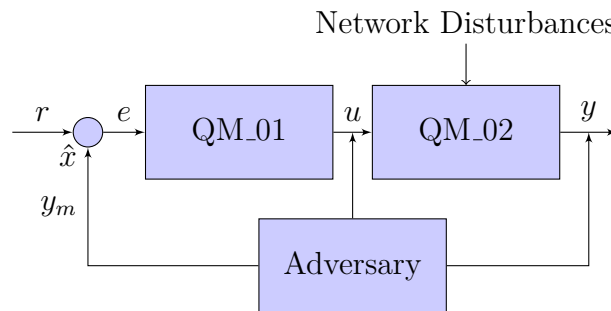
Fig. 2.2: Current Granularity categories

As multiple trading members can be trading under a single clearing member, the exchange generates execution messages to the clearing member with every details that can be sent to communicating parties in order to ensure that all the trading activities of the clearing member can view every action done by the trading members who are under the clearing members' authority. All the order details, execution details and order rejection details will be sent to connected parties and in order to define the scope of the research area, predefined multiple levels of granularity permissions were defined according to the user levels.

When interpret the term "granularity", system shows the ability to slit into a greater breakdown of the trade data related application programming interfaces into smaller parts. Granularity level is used to describe the extent to which a larger system and its information is subdivided[27]. In order to restrict access and visibility of orders and trade information within a clearing firm to a certain trading members. Order and trade messages are automatically being deleted permanently after predefined period of time. There is a certain doubt whether a sensitive field of or an information within a trade message accessed accidentally or maliciously by an adversary. Additionally, it is important to have a user level based permissions that have been defined in the system and added to restrict any trader, firm, client, network sniffer from publishing vital trade details to the general public[28].

2.8 Objectives of an Adversary and Possible Attack Types

Fig 2.3 shows the possible attack regions in the communication network between multiple parties. There are gateways where secure boundary releases the messages to be transferred to another secure boundary through a communication channel. In order to minimize the exposure, message transfers between communication parties have been disseminating through a communication pipeline to operate as the transport backbone across all environments in an IT Infrastructure provided by a third party vendor .



Variables

r	: Original Trade Message
e	: Edited Trade Message
\hat{x}	: Attack surface or attack vector
y_m	: Exploit over an identified vulnerability
QM_1	: Queue Manager server No 01
u	: Piplined communication network
QM_2	: Queue Manager server No 02
Network Disturbances:	Possible Network interferences
y	: Trade messages at an endpoint
Adversary	: Rogue user who has a wrong intention

Fig. 2.3: Possible attack regions

Message transmission between a secure boundary to secure boundary through IBM MQ® is depending on the mutual understanding of the security level of these connected parties. Modern day cyber attackers can comes in any form. Sniffing through network traffic and time based password cracking are some of the attacking modes that can activate on this domain[29]. Therefore hardening process of possible information leaking in every evasive systems is one of the most important activity that can be started from the requirement gathering phase of the system.

2.9 Threat Model

Threat model is a combination of three preliminary components[30]. Adversary should have the sufficient assistance in following three areas.

Resource

- Method of supplying the fundings
- Technology backup to conduct an attack
- Infrastructure
- Community or an organizational support
- Slow changing hardware or software components is an additional advantage for an attack

Oppertunity

- Method of entering must be aligned with the vulnerability
- Can be crated, discovered or developed by an adversary
- Can be consists of various potential and current access.
- Possible to make changes on an ongoing process

Motivation

- Adversary type
- Target event, occurence, duration
- Predefined mission goals and objectives
- Possible to make changes on an ongoing process
- political, economic, personal agendas.

2.10 Information Security vs Adversary

As per the current implementation, if there is a successful data breach in the system due to an attack at a data dissemination point, there is a high probability of occurring a information security related incident. Controls that have been applied in the live system guarantee the data integrity inside the secure perimeter and there is no control over an attack that can be occurred between the transmission process. Current security implementation will breach if an attacker can successfully alter trade data at the point of origin. There is a message sequence monitoring system which keeps the records of the trade data using an auto generated reference number[12]. Sudden changes in this sequence will trigger the limitations that have been set up in number of trading system components. Order matching system, gateway controllers, data servers and other programs will concurrently show the error in sequence of messages.

Fig 2.4 is a sample FIX message. All message fields are defined in ASCII and after identifying the header, body and a trailer, decoding a FIX message can be obtained by doing calculations as in Tab 2.3.

```
8=FIX.4.4|9=61|35=A|49=BRKR|56=INVMGR|98=0|34=1|52=20000426-12:05:08|108=30|10=143|||
8=FIX.4.4|9=61|35=A|49=BRKR|56=INVMGR|98=0|34=1|52=20000426-12:05:08|108=30|10=143||
8=FIX.4.4|9=61|35=A|49=BRKR|56=INVMGR|98=0|34=1|52=20000426-12:05:08|108=30|10=143||
8=FIX.4.4|9=61|35=A|49=BRKR|56=INVMGR|98=0|34=1|52=20000426-12:05:08|108=30|10=143||
```

Fig. 2.4: Example FIX message

Tag	Field	Value	Description
8	BeginString	FIX.4.1	
9	BodyLength	154	
35	MsgType	6	INDICATION OF INTEREST
49	SenderCompID	BRKR	
56	TargetCompID	INVMGR	
34	MsgSeqNum	238	
52	SendingTime	19980604-07:59:56	
23	IOIid	115686	
28	IOITransType	N	NEW
55	Symbol	FIA.MI	
54	Side	2	SELL
27	IOIShares	250000	
44	Price	7900	
25	IOIQltyInd	H	HIGH
10	Checksum	231	

Tab. 2.3: Standard format of a FIX message

The research area for this thesis has been limited only to implement security for trade messages which are transmitting through FIX[®] and FAST[®]. This was purposefully implemented since there are number of software vendors who are into stock exchange system development. FIX and FAST protocols are supporting all the trading venues since these protocols are considering as a standard protocol. An assumption was made that the code can also be applied on other protocols, if the proposed crypto program works accordingly in FIX environment.

2.11 Adversary and Attack Vectors

Fig 2.5 shows the inter-relationship between Adversary, Threat, Asset, Control and Attack vectors. Designing such diagram creates a visualization of the possible work-flow that can be conducted by an adversary to plan/execute an exploit against the trading system. Identifying this work-flow can be helpful in order to minimize the response time when an attack takes place. Even though an attacker is covering his tracks, security analysts will be able to make analytical decisions based on an activity work-flow of an adversary.

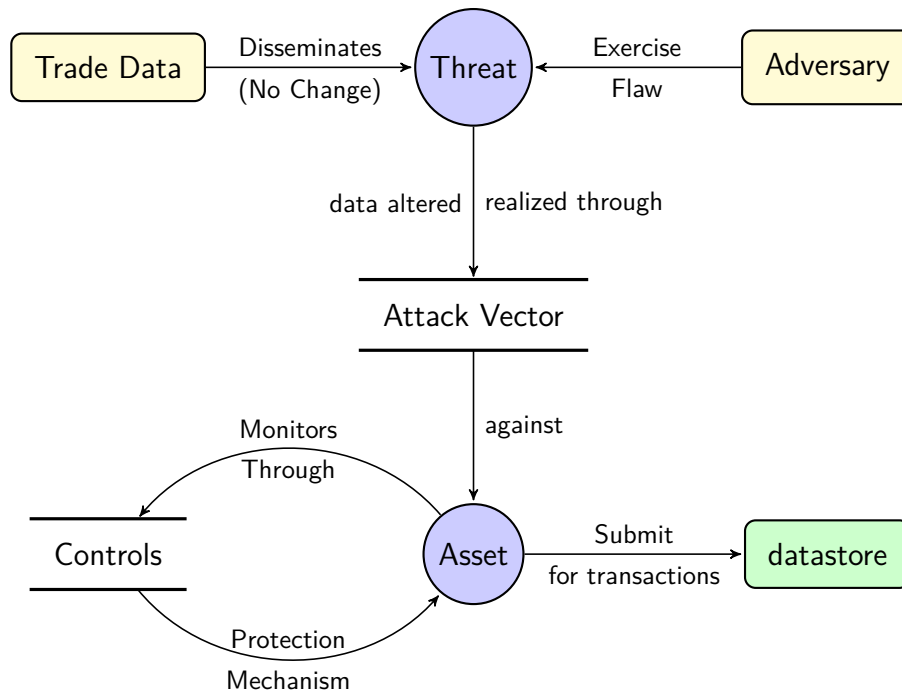


Fig. 2.5: Possible attack vectors

2.12 Evaluated and Achieved Objectives

The objective for doing literature review is to facilitate the distinguish the limitations, failures and to identify the probability of an operation triggered by an attack. Also, this thesis presented a set of desirable architectural features for trade management systems. The primary level goals include

- Identify process of most suitable algorithm based software from three programs that has been developed from three different cryptographic algorithms.
- The system executions and hiding data must be backward compatible and Adversary-proof. Backward compatibility is the process of reverting back an encrypted message to it's original plain text form. The development of the program must have covered the issue which identify as an attack vector.
- Maximize Flexibility
Crypto system must be flexible in order to cope with the market data system components which are running on different subsets of applications. As a results of the literature review, the most suitable location that the crypto program should be placed was identified in a trading system. All the results were calculated and collected by following required standards.
- Maximize Extensibility
Crypto system must offer extendibility for vendor based trade applications and data gateways. The trading system is a collection of different software components from different software vendors and the components are proprietary to one or more manufacturers. As a solution, the program developed in a multi platform compatible approach.
- Unified Interface
System must deliver functionalities by following mandatory policies and self-policing between various firms and vendor based components. Also newly introduced system must have a self deployment mechanism or it should securely managed across all trading platform by trigger from a single entity.
- Layered Architecture
The system must developed using a layered architecture. Since there are number of hardware components which are connected to the trading system by pre-programming or pre-configuration. There should be a solid support for the changes that can be arise in the future.
- Low bandwidth usage
The system has designed in order to use less bandwidth possible.

2.13 Selection of Algorithms

Security of a crypto system can be measured by the key size used while applying the cryptographic algorithm. It is a fact that greater the keys size, stronger the encryption. Since this thesis carried out in order to critically evaluate a candidate encryption mechanism while preserving the latency factor of trade dissemination, the length of the key is taken into consideration by balancing latency and security. When evaluating different algorithms, structure and the set of operations that algorithm performs can be considered as architecture of the algorithm. The implementation steps and the selection of algorithm are decisions of the developer according to the basic characteristics definitions of an algorithm, hardware and software availability and resource limitations.

These limitations and requirements provide evidence when selecting the algorithm and categorize the algorithm into asymmetric or symmetric . Asymmetric key cryptography is based on using different keys between the sender and the user while doing the encryption and decryption. Symmetric key system is differ from asymmetric system since the sender and receiver share a common key that is used in both encrypt and decrypt the message. Advanced Encryption Standard(AES) is a symmetric algorithm and it was selected for do the implementation on GPU based environment after considering the execution speed and the easily adaptability for text based systems[31]. Security aspect of the trading system is an affirmative measure of resisting a series of attacks and the govern the process of choosing an algorithm to use in such a crypto system, which was a key point taken into consideration.

Flexibility defines the algorithm can remain in existence after a minor or a major modifications according to the requirements. In this particular case, this characteristic considered as an important factor which monitored closely since most of the executions carries out in a GPU based environment. Candidate cryptographic algorithm must have the ability to adopt to changes that are being done in the code level changes. At the beginning, in search of the most suitable symmetric or asymmetric algorithm which the proposed system must based on, related changes were done for three different algorithms. Scalability is another major element on which encryption algorithms can be analyzed and categorized. It depends on parameters as memory Usage, encryption rate, software hardware performance and decryption rate.

2.14 Qualitative Analysis

Qualitative research approach can be broken down into identified problem, background analysis of the problem, rationale the factors with possible outcomes and objectives that can be addressed by introducing a solution. Research design and methodology which can be identified after successful iterations of the previous steps. Data analysis that can be collected from the test executions and observations. Qualitative analysis attempts to find an answer to the research questions by testing a hypothesis. Different types of research designs are there in order to use to measure the outcome of the research approach. Demonstrating, designing and interpreting of collected data are more meaningful and applicable for the process of qualitative analysis[32]. Trade data execution results that can not be straightforwardly converted in numbers and the analytical results directly related to notions, viewpoint, values and stress testing related results can be categorize as qualitative assessment of the system. System behavior observation while performing the encryption task in the background was the major consideration while doing this analysis. The variations of resource utilization can not be directly converted into a numbers based analysis since the different hardware combinations can produce unpredictable results as an output. In this thesis, following properties of a qualitative analysis were considered and analytical results were demonstrated.

- Hypothetical Analysis
- Interpretative Analysis
- Provocative Analysis

2.15 Quantitative Analysis

There are different classifications of quantitative research methods can be used in order to examine the research question and expected outcome of it. Namely research design, test and measurement procedures and statistical analysis are the key methods that are include in this analysis. The outcome of these analytical methods can be classified descriptive, experimental and causal comparative classification. Descriptive classification commonly based on observations and events. Experimental research approach is based on the solution and probable outcome after being applied the solution. In casual comparative classification, independent and dependent variables will be critically evaluated applying relationships between defined variables[33].

Implementing a commercial trading platforms is an exhaustive task and it be taken several years to complete a trading platform development. Since a real world trading platform is not allowing to do any production testing on it, the common method of validating tasks related to a trading system is done on a simulator system.

Identify the attack types and attack vectors

3.1 Introduction

In this chapter, there is a brief description of possible attack types that can be identified in the trading domain that was pointed out in the previous chapters in this thesis. An attack in information security threat perspective involves with an attempt to reveal, obtain, change, implant, destroy, remove or hide information without informing the authorized parties. An attack vector is a path which a hacker(or an intruder) can gain access to a computer or network server in order to deliver a payload or malicious outcome[34].

3.2 Identified attack types in this domain

3.2.1 Attack Types

The attacker's first goal is to identify potential targets for their mission. Attackers are often motivated by financial gain, access to sensitive information or damage to brand. They may even work in the company premises disguising as a genuine employee. This can be the first step of a planned attack over stock exchange. After that the attacker can set up a fake account with all the necessary view, read, write permissions as he deserved the permissions as an employee. Also if the employee is in managerial level who's intention is to harm the company's brand, he might have registered domains and create fake trade firms and brokerage companies for further hacking purposes. Once the bogus employee or the attacker determines when to deploy an attack and what defenses are in place in favor of attack. Since they have control up to certain level, selected attack vector is impossible to prevent or detect. It can be a zero-day exploit, a spear-phishing campaign or bribing an employee. Usually there is a minimal business impact. Finally, the attacker is ready to plan and venue for attack.

At the second step, the adversary tries to breach the corporate system and gain an access to the environment. After making an assumption that an adversary has the necessary access to the company to gain credentials of the system, using a valid credentials will not trigger any security precaution and it will allow to access to the and downloaded more tools to access the environment. If adversary has a sound knowledge in hacking, finding

these entry points are virtually untraceable. Due to the untraceability and undetectability of a planned hacking attempt, typically the targeted trading system is unable to detect, protect and respond on time. Even if the threat is successfully detected, possibility of minimizing the damage is very low. In practice, the attacker is always successful.

Attack types and attack vectors can be divided into two categories in order to identify the domain at a wider angle. In the trading system platform, there can be occurrences where an attacker can use multiple attack vectors and multiple attackers try to use the same attack vector. There are four steps in the life cycle of an attack from the attackers' perspective. The first and the second steps are related to the selection and executing of an attack. The steps which align with attack vectors are describe under next subsection.

3.2.2 Attack Vectors

Once the adversary has established a connection to the trading network, there is a possibility to compromise message transmission systems and user accounts. Their goal is to alter trade messages before it disseminate to the trading parities and for clearing purpose and create a connection to outside channel in order to do insider trading or communicate information which would affect the share price before it is released to the market and the price can be shifted in an unlawful manner.

As per the IBM MQ[®] configuration, an administrator who has permissions can use the WebSphere[®] MQ admin tool to create and configure new peers and add or destinations. The attacker can impersonate as an authorized user and there is a high possibility of deploying a successful attack by the intruder in this phase. As illustrated in Section 2.6, there is an actual possibility of an insider attack which can ended up with a high successful rate.

Forth step of an insider attack deploying by adversary is where attacker identify and establish the necessary level of privilege to achieve their objectives. If the adversary successfully executed the previous three stages, at this time, they have deployed the attack vectors over the communication channels and successfully impersonated with credentials acquired in the previous phases. As final activity of this stage, the attacker accesses to the target data, market data servers, trade management systems and trader and trading data and alter or compromised the details.

3.3 Outcomes of a Successful Insider Attack

When an adversary reached final stages of his mission, ex-filtrating the trade data which he was after, changing critical information and daily trade data and disrupting business operations are some of the critical hacking activities that he can perform. If adversary has an intention of monitoring the activities by staying at the company for a longer period of time, passive attack, traffic analysis and eavesdropping attacks are also possible. Previous studies have highlighted that there were successful data breaches in the firms listed on NASDAQ⁹, NSE¹⁰ or NYSE¹¹. The studies have been conducted on successfully executed data breaches, phishing attacks, information security breaches and website attacks[35].

In previous data breaches to dissolve the market's effect of a disclosure of an attack, an event window of three days was used focused on the date of the breach announcement. It was consider as a practice in most of the cases[36]. Proposed cypto-system must create an acceptable level of defense against above attack types in order to provide protection. Properly set up cryptographic system must be designed in order to make the system difficult to break as possible. Companies who are searching on a cryptographic security system will also consider the expenses and possible gain before implementing a security mechanism in the company. Sophisticated technology can be used by attackers to decipher encrypted trade information by investing on software and hardware . In theory there is a possibility of breaking any encryption algorithm by using exhausting mechanism by referring every possible key in a sequence. This brute-force method requires a large amounts of computing, electrical power that need to harness a key and the requirements will differ as length of the key increase.

Tab 5.10 show how the budget, time and cost with related to different type of adversaries. Hardware level solutions base on FPGA¹² and ASIC¹³. Brute-force, square attack, key attack, differential attack and improved square are some of the attack types that can be implied on a cryptography system which has been developed in AES, RSA or triple DES. Therefore the proposed mechanism should not breach under common attack types.

For reference purposes, the comparison has been done for a key size length of 64 bits and 128bits. One of the primary goal of encrypting the trade data is to safeguard the relevant data that has been recorded in the trade file until clearing process of the trade is finalized. Usually secrecy of the trade has to be kept in the database for three days following the trade day. If the adversary is unable to crack the encryption within three days following the trade message generation process, it can be consider as a successful approach to secure trade messages using a cypto system[37].

⁹ The second largest exchange in the world by market capitalization

¹⁰The National Stock Exchange of India Limited

¹¹The New York Stock Exchange

¹²These are programmable hardware which is specifically designed for encryption/decryption

¹³These are programmable hardware which is specifically designed for encryption/decryption that can test 200 million keys per second

Type of Adversary	Budget (USD)	Tool	Time & Cost/Key 64 bit	Time & Cost/Key 128 bit
Regular User	200	1 PC GPU unit	Not feasible	Not feasible
Team of individuals	2000	7 PCs 7 GPU units	38 years	Not feasible
Small Business	10000	12 PCs 10 GPU units 2 FPGA units	560 days	Not feasible
Corporate Department	50000	20 PCs 15 GPU units 5 FPGA units	30 days	Not feasible
Large Corporation	100000	50 PCs 40 GPU units 10 FPGA units	24 hours	Not feasible
Intelligence Agency	1000000	200 PCs 150 GPU units 50 FPGA units	10 mins	Not feasible

Tab. 3.4: Break time of an average 128bit key

In this example the adversary will not be able to breach before T+3 period of time after being encrypted the trade data with AES, RSA or 3DES. If the size of the key is 64 bit, there is a possibility of cracking the encryption with any known attack type. Since the implementation is done with a minimum key size of 128bit, the probability of cracking the key with in a period of 3 days is very limited.

In symmetric encryption algorithms, the most popular attack method used by attackers is the brute-force method . It is the generic method that theoretically works and in practice which is unfeasible break into the system using brute-force attack since the systems are developing using encryption keys that are having a large bit length after hardening keys to make it totally infeasible. As a common practice it is highlighted that an encryption algorithm is said to be secure if no better attack method is known[38].

The public key has a strong internal structure by its origin and there is a possibility to access into the private key if an attacker can reveal the private key from its public key. The main component of the public key is the modulus, which is a composite integer. The private key is equivalent to the knowledge of the prime factors of the modulus. This can be done a lot more efficiently than trying out possible private keys; in fact, trying prime factors one at a time is a method known as trial division. It is not feasible to break RSA through brute-force since RSA is an asymmetric encryption algorithm with in the time period in which the trade must keep as a non-disclosure agreement.

Implementation of Crypto System

4.1 Introduction

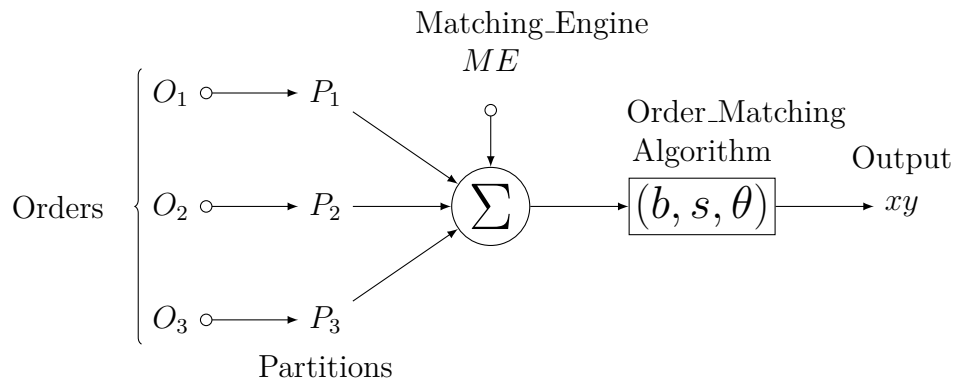
The most appropriate cryptographic algorithm for the implementation was chosen after developing three crypto systems in different encryption standards. Throughout this chapter, a brief description of each of these encryption standard and the programs that were built in C++ language.

4.2 Test Environment Preparation

The number of orders that are being transmitted through a channel per day can be a large number. Therefor in a live trading system, there is a concept called “partitioning” the order dissemination channels. As the name imply, order books will be uploaded into n number¹⁴ of machines which are then connected to a centralized machine which is more powerful than other machines. Algorithmic trading functions, trade clearing functions, order validation and rejection can be done through this machine. One advantage of implementing a cluster design is that the hardening process of the program and related hardware can be easily finalized. Also when collecting the research data, assumption was made that crypto system will display the same functionality either in individual machine or on an interconnected system.

¹⁴This number can be vary according to the size of the trading system

4.2.1 Order Matching Process



Variables

Orders	: Submitted orders at a given time
O_1	: Order sequent submitted to Partition_01
O_2	: Orders sequent submitted to Partition_02
O_3	: Orders sequent submitted to Partition_03
P_1	: Trading system partition_01
P_2	: Trading system partition_02
P_3	: Trading system partition_03
Partitions	: Predefined partitions to recieve orders
ME	: Matching Engine
Σ	: Comobined sequence of trades
Order_Matching Algorithm:	Developed algorithm to match orders
b	: Buy order
s	: Sell order
θ	: Exposition Function inside matching algorithm
bs	: Matched trade deetails

Fig. 4.6: Order matching process

According to figure 4.6 diagram, an assumption was made that either one of these partitions worked corresponding to given specifications and expecting benchmark. There is a low probability of a system failure after plugged in the remaining partitions in to the system. Algorithm should work accordingly in every partition and the calculations have to be follow the same flow in order to keep the integrity of trade information.

4.2.2 Test Environment Hardware Specification

After selecting the most suitable encryption algorithm, the executions were done in different types of machines with number of hardware specifications changes to analyze benchmark results. In order to collect data in more specific and standard manner, all the tests related to the process of selecting the most suitable encryption algorithm to develop the trade message encryption based on a crypto system were done using a laptop with following system specifications.

- HP Pavilion dv6-6c19tx Entertainment Notebook PC
- Intel Core i7-2670QM CPU @ 2.20GHz
- AMD Radeon[®] HD 6700M 1 GB GDDR5 graphic card
- Windows[®] 10 64-bit OS Build 15063.729
- 8GB DDR3 SDRAM

Multiple performance metrics were monitored and collected data. While implementing the security, it was notified that there is a trade off between the security and the latency factor[39]. When selecting the scope of the research, encryption was applied on the trade message and observed various information to keep up with the industry standards.

- Encryption time
- CPU utilizations
- GPU utilization
- Overall system hardware utilization
- Time taken to execute a set of instructions
- Number of instructions executed within a duration
- Current status vs After being encrypted status

4.3 RSA Implementation in Depth

First program was developed the RSA cryptography algorithm. Appendix A, B, C, D and F contains the diagrams which used to design and develop the AES based crypto system. The algorithm was implemented in the C ++ language in Windows 10 - 64 bit environment using Code::Blocks. It is an open source, cross platform program. Since the GPU cards had been installed in Windows, Apple OS X, Linux based computers, as a precaution, proprietary independent and free program were used. That decision was also supported to do the comparison between encryption algorithms in a horizontal manner. Code::Blocks supports number of compiler environments and it was an additional benefit of using the programs' capability to the comparison between algorithms.

In RSA methodology a and b are two prime numbers which can be selected randomly with large size.

- After selecting these numbers, one computation was done as $N = ab$.
- Another value was defined as $\phi N \equiv (a - 1)(b - 1)$.
- Another new number “x” introduced where x was such that $\text{gcd } x, \phi N \equiv 1$.
- Then there exists a “y” value that satisfies $xy \equiv 1 \pmod{\phi n}$.
- After considering above equations a conclusion can be made on as following

Encryption as $E(m) \equiv m^e \pmod{N}$

Decryption as $D(c) \equiv c^d \pmod{N}$

Encrypted trade messages using the RSA algorithm, in the key generation, N calculated as a large value, in that case the attacker breaks the value with a succession probability of $t = (p - 1) * (q - 1)$, it is very difficult. In order to make the RSA safety, it must choose large a and b ; use a longer key is beneficial and harmless. Users usually choose more than 100 decimal digits, where the attacker cannot decompose the N in polynomial time effective internal. The experiment results were

4.3.1 RSA Pseudocode

Following set of Pseudocodes were designed for the purpose of implementing RSA based program. Rabin-Miller preliminary test was also embedded to the main program in order to test whether the selected prime is a probable prime or not.

Algorithm 1: RSA algorithm designed for key generation

```

Result: RSA Key Generation
Input : 2 Prime Numbers
Output: Public key and Private key

1  $a \leftarrow$  Randomly generated prime number
2  $b \leftarrow$  Randomly generated prime number
3  $N \leftarrow a*b$ 
4 foreach Successesfully generated keypair do
    /* Generated Prime values will use to do the
       following calculations */
5  $\phi N \leftarrow (a - 1)(b - 1)$ 
6 if  $\phi N$  Generated correctly then
7      $x \equiv$  calculation of exponent  $(\phi N, N)$ 
8     there is a  $y \leftarrow (N, e)$ 
9     Using theories in Eucledian Algorithm,  $(ed \equiv 1 + \phi Nk)$  for private
       exponent d
10     $(d, k, gd) \leftarrow (e, \phi N)$ 
11    if  $d < 0$  then  $d \leftarrow (d + N)$ 
12
13    else
14    |  $pvtKey \leftarrow (e, \phi N)$ 
15    end
16    return  $(pvtKey, pubKey)$ 
17 end

```

From the beginning, the focus was to build a crypto system using an asymmetric key method. Number of connected users to the trading system backed up that decision due to maintainability related issues.[40]

In above step, there was a term called “calculation of exponent” and the Public Exponent generation algorithm is given by the following pseudocode.

Algorithm 2: RSA algorithm Public Exponent

Result: RSA Public Exponent Generation
Input : (ϕN) and N
Output: Encrypted exponent

- 1 Calculation of exponent $(\phi N, N)$
- 2 **while** *True* **do**
- 3 $r \leftarrow$ *Randomly generated number*
- 4 $e \leftarrow r(\text{mod } (N - 2)) + 2$
- 5 **if** $\text{gcd}(e, \phi N) = 1$ **then**
- 6 **return** (e)
- 7 **end**

Without a correct decryption of an encrypted data, the total can be disadvantageous if the interpretation of the data contains any errors. Encryption and decryption phases were developed focusing the correctness of the code itself. Key generation and the encryption process execute the main functions of the process while decryption takes place after a successful completion of mentioned processes.

Algorithm 3: RSA algorithm Encryption

Result: RSA Encryption Process
Input : Plain text and Public (N, e)
Output: Encrypted Text Cipher text

- 1 Encryption M, pubKey
- 2 **while** *True* **do**
- 3 $e \leftarrow \text{pubKey.getpubEXP}()$
- 4 $N \leftarrow \text{pubKey.getModule}$
- 5 $C \leftarrow \text{exponent}(M, e, N)$
- 6 **return** (c)
- 7 **end**

Algorithm 4: RSA algorithm Encryption

Result: RSA Decryption Process
Input : Plain text and Public (N, e)
Output: Encrypted Text Cipher text

- 1 Encryption M, pubKey
- 2 **while** *True* **do**
- 3 $e \leftarrow \text{pubKey.getpubEXP}()$
- 4 $N \leftarrow \text{pubKey.getModule}$
- 5 $C \leftarrow \text{exponent}(M, e, N)$
- 6 **return** (c)
- 7 **end**

4.4 AES Implementation in Brief

The symmetrical key algorithm was not taken as an exact solution for the considered research boundary. The key exchange was a problem since in a trading system, number of connection nodes can be an issue that can be hard to overcome. The simulator data fed into the AES algorithm assuming that the authentication correctly happened between the trading system and the connection node prior to data transmission.

Secure key management is users' liability and responsibility[41] and AES algorithm's main computation related fixed block size is with a size of 128 bits with variable key sizes. The key sizes are 128, 192 or 256 bits. Basic AES computation is iterated under number of rounds which depends on the key size. Number of iterations with respect to above key sizes are 10, 12 and 14. After the implementation of the program architecture for the AES encryption method, There was a slow execution rate due to this number of iterations after selecting the key sizes as 128 which is suitable for securing trade data in exchange system in real-time application in Fig (4.7).

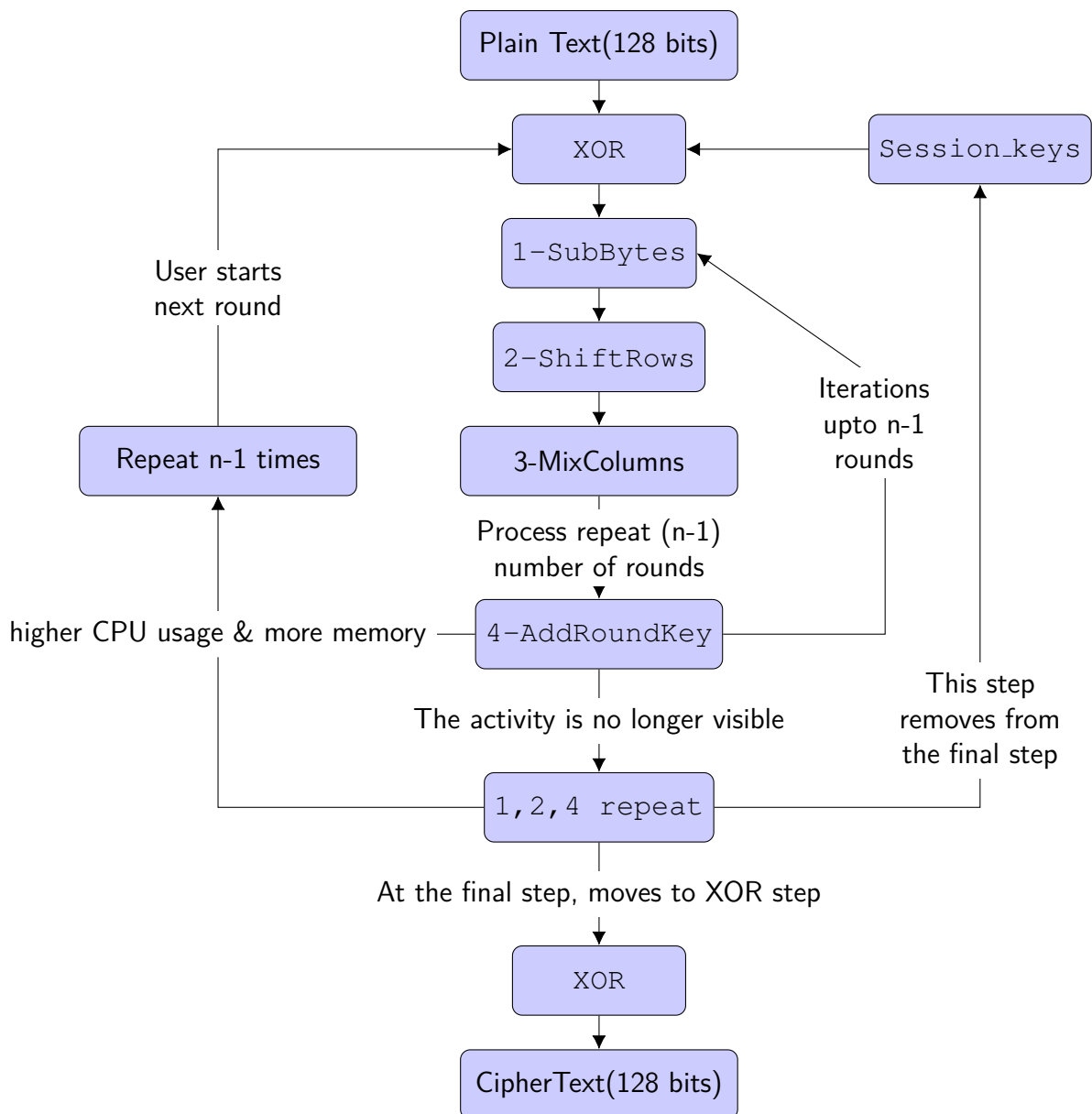


Fig. 4.7: AES Implementation process flow

4.5 Triple DES Encryption in Brief

The last selected algorithm to develop a sample program was 3DES algorithm. The payment card industry has been more relied on the security based on the solutions that were created using 3DES algorithm [42] [43]. The original development of DES was done by an IBM team and in 1977 it was adapted as a standard secure encryption mechanism by NIST. After that, it has always been a basis for performance comparisons and for lay a foundation for new encryption algorithms. Triple Data Encryption Standard, 3DES is a type of symmetric cryptography algorithm that work on block cipher algorithms which are applied three times to each data block. DES standard is the predecessor of 3DES and the key size has been increased in 3DES to guarantee additional security through encryption applications . 3DES has advantages and disadvantages over its predecessor. One major disadvantage of 3DES is, it is three times slower than regular DES. While doing the experiments, it showed significant slowness when encrypting the data set. But can be billions of times more secure if used properly. Since it is based on the DES algorithm, 3DES was easy to modify according to the security requirement in trading system.

Some of the most important properties of 3DES encryption are as follows.

- 3DES is a Block cipher with symmetric secret key
- Block length = 64 bits
- 3DES is easy to implement and can be accelerate in both hardware and software
- Provide easy and well security to trading system
- The detection of the fraud use of the card is found much faster than the existing system
- Possible Key lengths = 56, 112, or 168 bits

3DES is an extention of DES with two 56-bit keys applied. Given a trade message, the first key is used to DES encrypt the trade message. The second key is used to DES decrypt the encrypted trade message. Since the second key is not the right key to apply on the trade message to retrieve the original message. This decryption scrambled the trade data further. The twice-scrambled trade message is then encrypted again with the first key to yield the final encrypted trade message. This three-step procedure generate the encrypted trade message. The encryption process of the 3DES is more lengthier than other two procedures that were considered inside the scope.

Performance Results

5.1 Introduction

The RSA algorithm can be implemented both in hardware and software. Implementations based on software solutions without any customization take the smallest resources utilizations, These changes offer limited level of information security. The continuous growing enthusiasm for high speed, high volume secure communications combined with physical security, hardware implementation of cryptographic algorithms becomes essential. The proposed RSA was implemented on GPU technology. This RSA encryption based on hardware implementation was useful in securing message transmission. A successful implementation of this RSA algorithm on GPU can be useful in communication such as military communication where secure message transmission protocol is a greater emphasis on the integrity of communication.

5.1.1 Results analysis

Previous studies have showed that authentication and selecting keys for encryption decrease the normal rate of data transmission. These phenomena was observed in all the three programs that were considered in this study. There was a delay in execution at the start of the programs.

In a computer program, for the encryption process mathematical based matrix calculations, substitution and transformation are being used with a secure key. The key is independent from the original trade data and in multi party communication, asymmetric key based encryption performs better results than in the symmetric key based algorithms in security perspective. But due to the size of trade data message, we consider two symmetric algorithm based mechanisms and 1 asymmetric algorithm. DES, 3DES, and AES use the same key to encrypt the trade message and to decrypt the encrypted trade message and therefor 3DES and AES considered as possible cryptography encryption algorithms to develop two prototypes. RSA and Elliptic Curve cryptosystem use different keys in encryption and decryption processes and Only RSA encryption standard based prototype developed and Elliptic Curve based solution was not considered for this thesis due to its complexity in default nature.

Maximum number of execution that can be performed with in predefined period of time was collected since the maximum workable speed of the encryption was recorded since it is necessary to set up the benchmark of calculations to compare with the industrial and minimum requirement standards.

Field of cryptography must align with Confidentiality, Integrity and Availability. Up-to now the findings indicate that regular usage of cryptography could improve secrecy and integrity of trade data, but availability is also playing a vital part in the trade message transmission.

Samples were executed continuously throughout a time period of 100 seconds. While this experiment kept the time constant at a static rate, changing parameter was the cryptographic algorithms that were used in the previous examples. Encryption algorithm used in graphical processing unit connected systems and in RAM based systems. There was a significant improvement while the experiments were done in the graphical processing unit environment as shown in Fig 5.8.

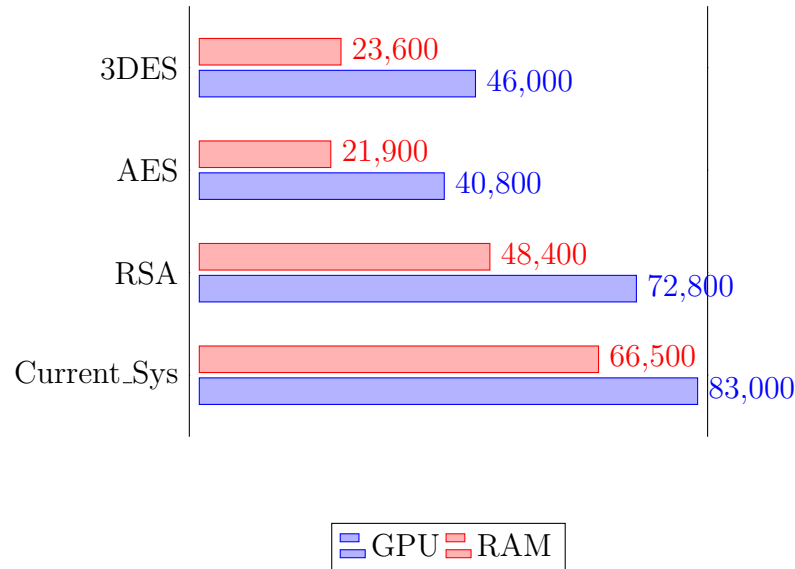


Fig. 5.8: Average order execution on GPU and CPU

After developing three crypto systems, the first test was executed to check the preliminary functionality of the program. In a trading system perspective, the main benchmark of the system is to monitor the execution rate of the trades. The basic results showed that RSA prototype was the fastest algorithm to encrypt trades with in a pre-defined period of time. RSA was slower than the existing mechanism but the extra security layer added to the trade file added an additional protection to the outcome. Subsection 1.4.2 presented that the average order rate of a trade system is around 7000-12000.

The current execution time of the trade messages showed in Fig 5.9 and Tab 5.5. The actual time can be deviate due to the network speed, system configuration and the delay that can be occurred due to the message routing. Number of execution was increased in the system when th trade messages received with a steady average.

This test was useful to check the availability of the system throughout a pre- defined period of time. In an actual system, number of trade messages with in 1000ms is around 60-70. The tested conditions covered the peak volume rate.

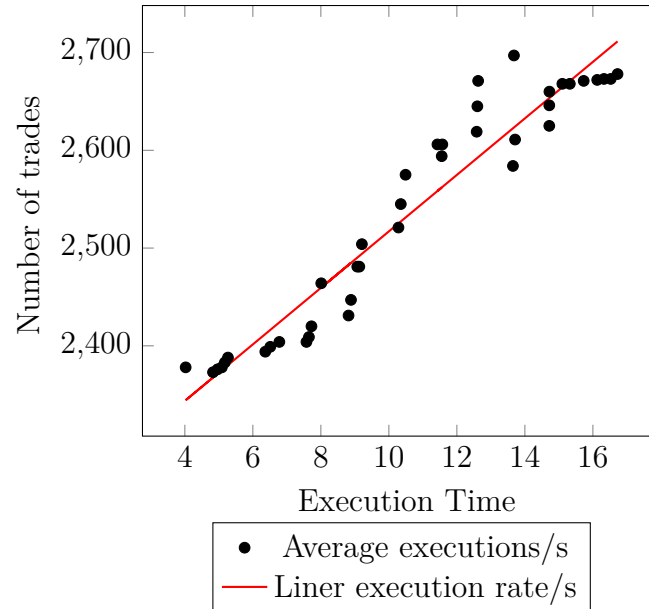


Fig. 5.9: Average order execution rate in the current system

Time	No. of Trades	Time	No. of Trades
4.828296	2373	10.4907	2575
4.962797	2376	11.55391	2594
4.023903	2378	11.42811	2606
5.087601	2378	11.5723	2606
5.176099	2383	12.58201	2619
5.2679	2388	12.60261	2645
6.365997	2394	12.62541	2671
6.513399	2399	13.64899	2584
6.779097	2404	13.6783	2697
7.575899	2404	13.7117	2611
7.649	2409	14.7178	2625
7.723602	2420	14.7178	2646
8.814098	2431	14.72139	2660
8.886108	2447	15.10169	2668
8.011303	2464	15.32141	2668
9.070704	2481	15.72979	2671
9.131198	2481	16.12699	2672
9.206599	2504	16.3264	2673
10.28229	2521	16.5246	2673
10.35089	2545	16.72478	2678

Tab. 5.5: Execution rate of the trade messages on current system

The execution time of the trade messages after being encrypted with the RSA cryptography algorithm is as follows in Fig 5.10 and Tab 5.6. The actual time can be deviate due to the network speed, system configuration and the delay that can be occurred due to the message routing. The execution rate was slower than the existing system after being encrypted by RSA encryption.

The speed of the encryption process was in the acceptable zone. RSA based program was the fastest algorithm among 3 prototypes.

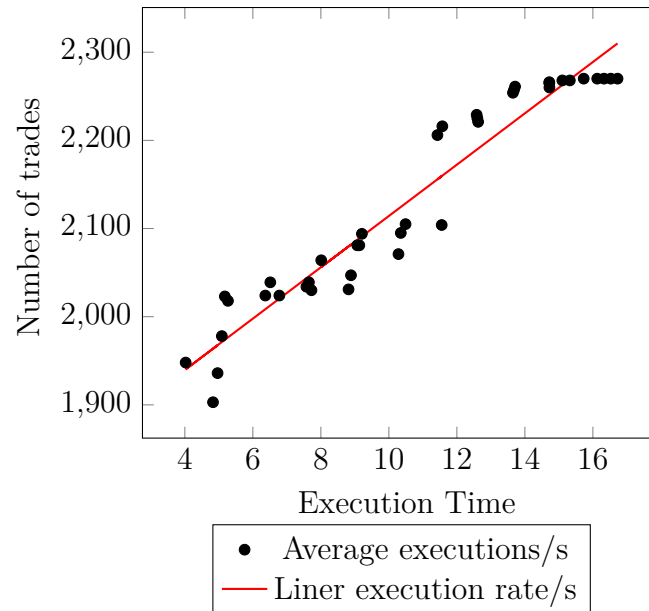


Fig. 5.10: Average order execution rate in the RSA based system

Time	No. of Trades	Time	No. of Trades
4.82827746	1903	10.4907	2105
4.96277277277	1936	11.5537271	2104
4.02272703	1948	11.4281771	2206
5.087677	1978	11.5723	2216
5.17677	2023	12.582721	2229
5.267579	2018	12.60267	2225
6.367777	2024	12.6254721	2221
6.5137757	2039	13.648979	2254
6.77977	2024	13.67873	2257
7.575879	2034	13.7857	2261
7.648578589	2039	14.71785778	2265
7.7236757	2030	14.715778	2266
8.8147757	2031	14.72139785789	2260
8.886187777	2047	15.1016859	2268
8.0113087575	2064	15.3217857	2268
9.07075785	2081	15.7275587	2270
9.13157857	2081	16.12685	2270
9.206557587	2094	16.325764	2270
10.28285875	2071	16.525746	2270
10.3578587	2095	16.72477878	2270

Tab. 5.6: Execution rate of the trade messages on RSA based system

The execution time of the trade messages after being encrypted with the AES cryptography algorithm is as follows in Fig 5.11 and Tab 5.7. The lowest performance among the three encryptions were shown by the AES encryption program. Number of rounds involved with the encryptions basically affected the performance of AES.

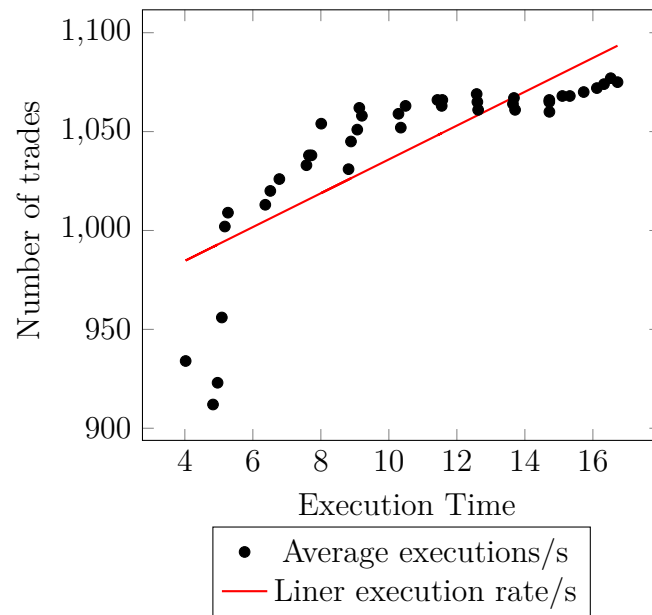


Fig. 5.11: Average order execution rate in the AES based system

Time	No. of Trades	Time	No. of Trades
4.828296	912	10.4907	1063
4.963497	923	11.34391	1063
4.9823903	934	11.424411	1066
5.08741	956	11.5433	1066
5.176099	1002	12.58601	1069
5.263322	1009	12.60261	1065
6.3653197	1013	12.62541	1061
6.5134699	1020	13.64899	1064
6.77657	1026	13.6783	1067
7.575199	1033	13.7145	1061
7.65569	1038	14.7548	1065
7.723102	1038	14.788	1066
8.856098	1031	14.72878	1060
8.88655	1045	15.188	1068
8.04513	1054	15.3248	1068
9.045634	1051	15.764	1070
9.155668	1062	16.18484	1072
9.25669	1058	16.3288	1074
10.286856	1059	16.5256	1077
10.350489	1052	16.72864	1075

Tab. 5.7: Execution rate of the trade messages on AES based system

The execution time of the trade messages after being encrypted with the TripleDES cryptography algorithm is as follows in the Fig 5.12 and Tab 5.8. The actual time can be deviate due to the network speed, system configuration and the delay that can be occurred due to the message routing. 3DES was faster than AES algorithm but was slow in performance than the RSA encryption.

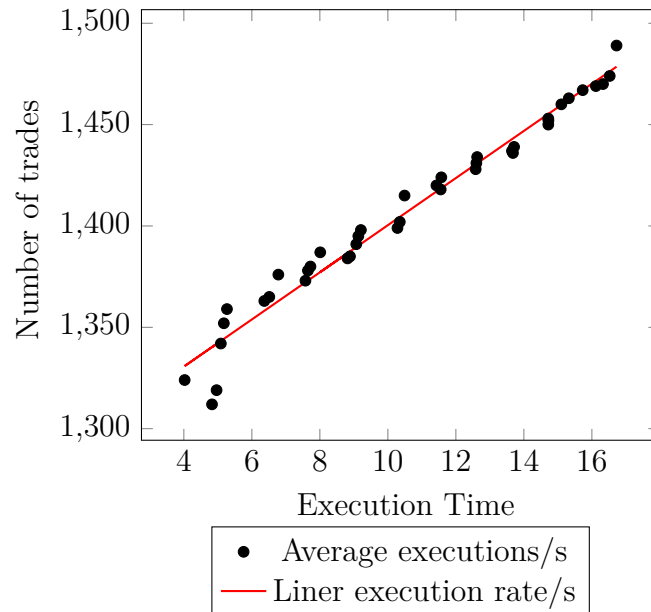


Fig. 5.12: Average order execution rate in the 3DES based system

Time	No. of Trades	Time	No. of Trades
4.82845656	1312	10.4907	1415
4.9627546	1319	11.553946461	1418
4.02395645	1324	11.4286341	1420
5.0876564	1342	11.5723463463	1424
5.176456	1352	12.58204341	1428
5.2675659	1359	12.602603636	1431
6.36554656	1363	12.625405645	1434
6.51356455	1365	13.6486456	1437
6.774645645	1376	13.67835645	1436
7.5755469	1373	13.71564	1439
7.64464549	1378	14.714564	1450
7.723564554	1380	14.71564568	1453
8.8140263463	1384	14.721456456	1452
8.886245162	1385	15.10145645	1460
8.0113464663	1387	15.32144564	1463
9.0704346346	1391	15.72956456	1467
9.131634634	1395	16.1170545645601	1469
9.2066356	1398	16.32645664	1470
10.2826546	1399	16.556456246	1474
10.35046556	1402	16.7247565454	1489

Tab. 5.8: Execution rate of the trade messages on 3DES based system

The comparison between the execution time of CPU based system and GPU based system is as follows. The comparison has been done considering the following conditions.

- Execution speed of the trade messages in the existing system in CPU environment
- Execution speed of the trade messages in the existing system in modified GPU environment
- Execution speed of the trade messages in the proposed system in GPU environment
- Industrial acceptable benchmark of the trade message execution in a system

The trade messages transmission is the most important process between the stock markets, clearing parties and the stock brokers. The transmission and parsing of trading messages take more time than the business logic execution[44] and outcome of the test were also showed the trade message transmission took more time than the execution period of the trade.

Fig 5.13 is a representation of the data execution rate calculated under MiB with related to execution time. The standard data transmission rate¹⁵ has plotted against the time and in this case the acceptable trade data rate has been considered as the information size of 3 mebibytes per second for day trade. The average rate has been calculated with the mebibyte values purely due to the calculation purposes. The usual notation of data size is given in decimals and when the units are converted into the actual binary format of the data representation, it can be an affective factor for the latency related calculations.

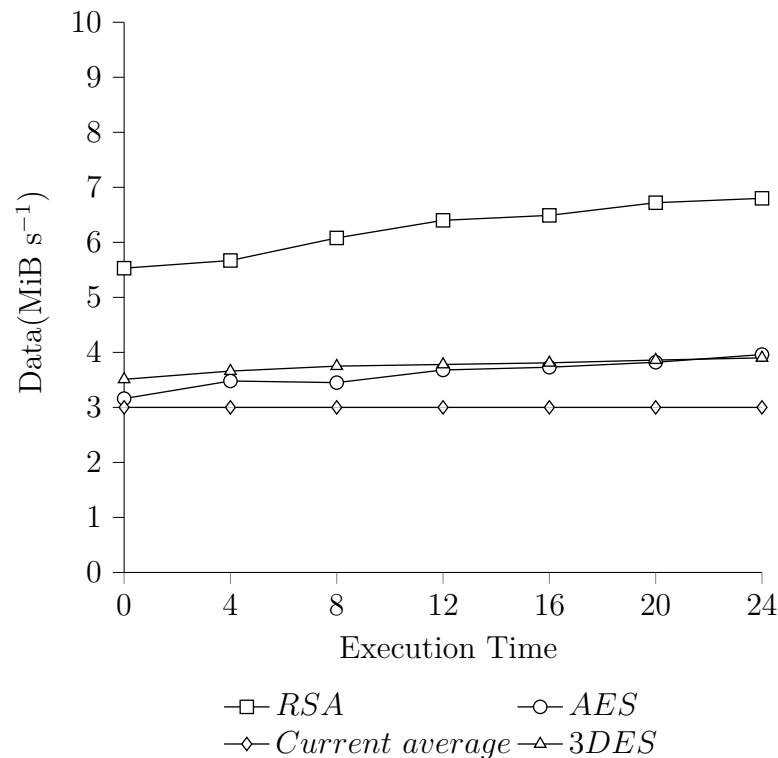


Fig. 5.13: Average data execution time of the trade data

¹⁵The rate may differ from a market venue to market venue

The time that has been taken to encrypt a set of trade messages is as in the Fig 5.14. The comparison has been executed under static conditions within the experiment time and the network speed throughout the period of time may vary due to the network condition. All other hardware conditions were set to a static parameter set since the comparison between the algorithms were done only in the code level of the desired system.

In order to have an indication of the results in both qualitative and quantitative methods, variables were kept under a static condition by only varying the time factor and the code level encryption algorithm changes.

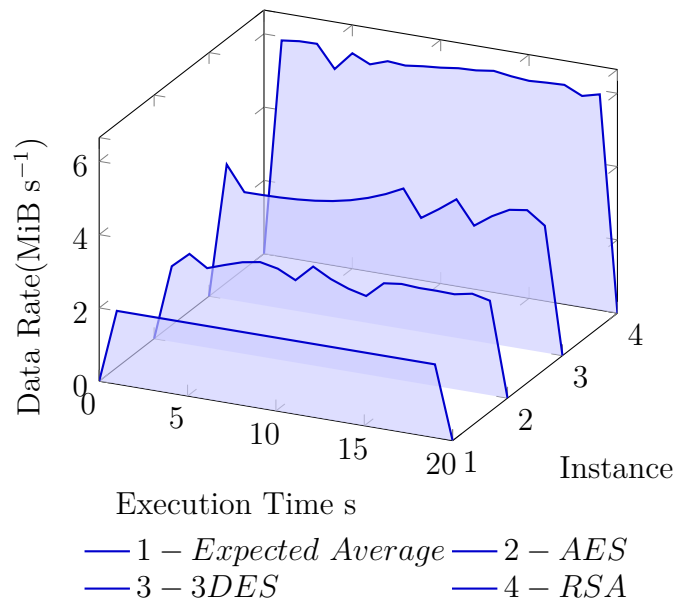


Fig. 5.14: Relationship between the data encryption rate and the time

The study of the execution speed under different algorithms were mainly carried out with the focus of identifying the algorithm with an acceptable accuracy with use of existing volatile memory components.

After conformational analysis of data execution rate of encryption algorithms, it was necessary to record the memory consumption of the RSA algorithm based system in order to identify whether the memory consumption of the system has not been altered by the modifications that were made prior to the experiment.

The next section of the experiment was concerned with the resource usage of the system with in the duration of the experiment. Hardware utilization monitoring as in fig 5.15 is another important factor in-order to test the compatibility of the existing systems after being introduced by a crypto system. The changes should be cost effective since it is necessary to implement the changes on the production system with minimum budget allocation.

This test was only conducted for the RSA based system since it was showing positive results than other methods and testing were prioritized mainly on RSA based program.

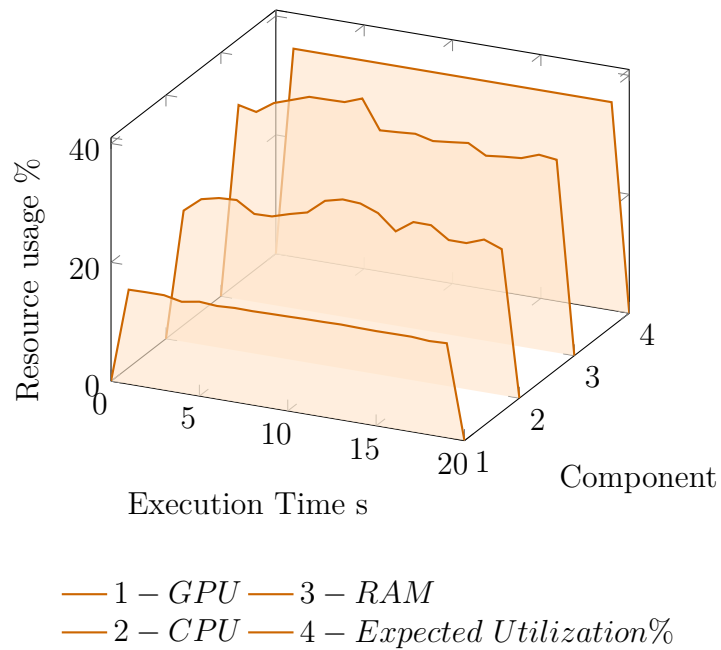


Fig. 5.15: Relationship between Resource usage and the time

In contrast to the early findings, however the experiment was based on the comparison of time which was taken to encrypt the files by varying the file size. The next calculations were based on the encryption duration with different file encryption standards. There are several trading statuses after a market start and the data dissemination file size is depending on the size of data in the trading cycle. It is a must to identify that whether the encryption can stand up to the required data transferring limit after applying the encryption. These results therefore need to be interpreted with standards since the calculations have to be completed with in the given period of time Fig 5.16.

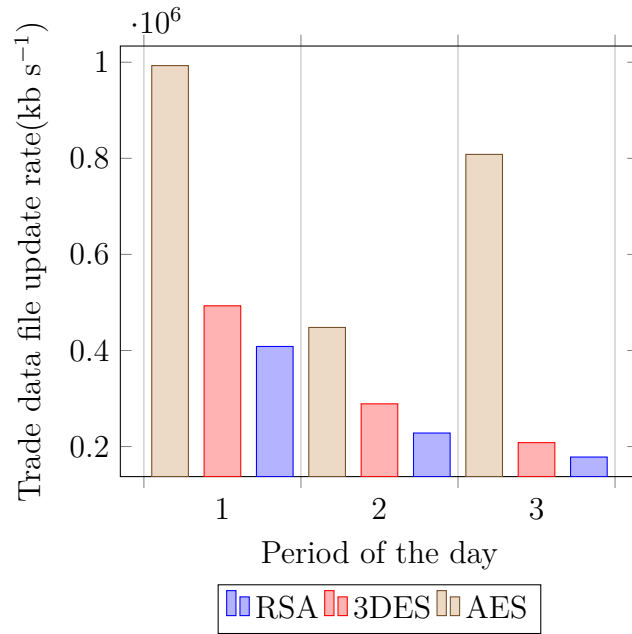


Fig. 5.16: Trade file execution rate in different time of the day

The numbers in X-axis represent the following sessions of a regular market.

- 1 - Start of the day (*Market start*)
- 2 - Regular trading session
- 3 - End of Day

At the start of a day, system is disseminating details about the instruments, firms and previous day trading details that has been successfully recorded in the market. File size is considerably high when instruments and firm data recored in the file. Inside the regular trading period, files will only contain the orders and trade data and the order submission data. The file size is considerably smaller than the start of the day and end of the day market data dissemination files.

In a particular trading session of a market data system, it is identifiable that there are sudden changes in the trading data dissemination due to unusual behaviors[45] in the stock markets.

Behavioral patters of movement in variations of trades was identified as follows. The changes were monitored and logged against three encryption algorithms as in Fig 5.17

Number of trades	RSA	AES	3DES
100	$1 \cdot 10^{-2}$	0.32	0.21
250	$1.8 \cdot 10^{-2}$	0.49	0.2
500	$3.5 \cdot 10^{-2}$	0.87	0.8
1,000	$5 \cdot 10^{-2}$	1.15	1.02
2,000	0.22	1.8	1.4
3,000	0.4	2.6	2.6
4,000	0.43	3.7	3.98
5,000	0.51	4.21	4.7
6,000	0.84	4.92	5.21
7,000	1.41	6.2	6.62
8,000	1.89	8.1	8.73
9,000	2.4	9.4	9.9
10,000	4.1	11.02	12.67

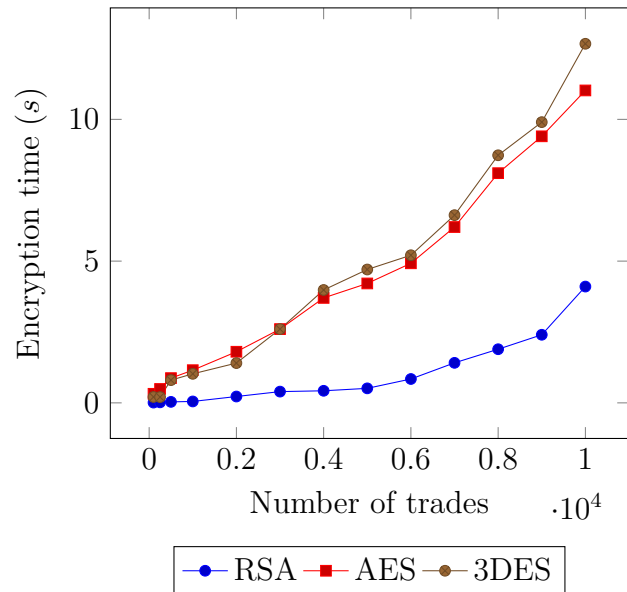


Fig. 5.17: Encryption time for a fix sized trade file

5.1.2 RSA Encryption-Decryption Results

RSA encryption and Decryption rate on a GPU based system is as follows in Fig 5.18.

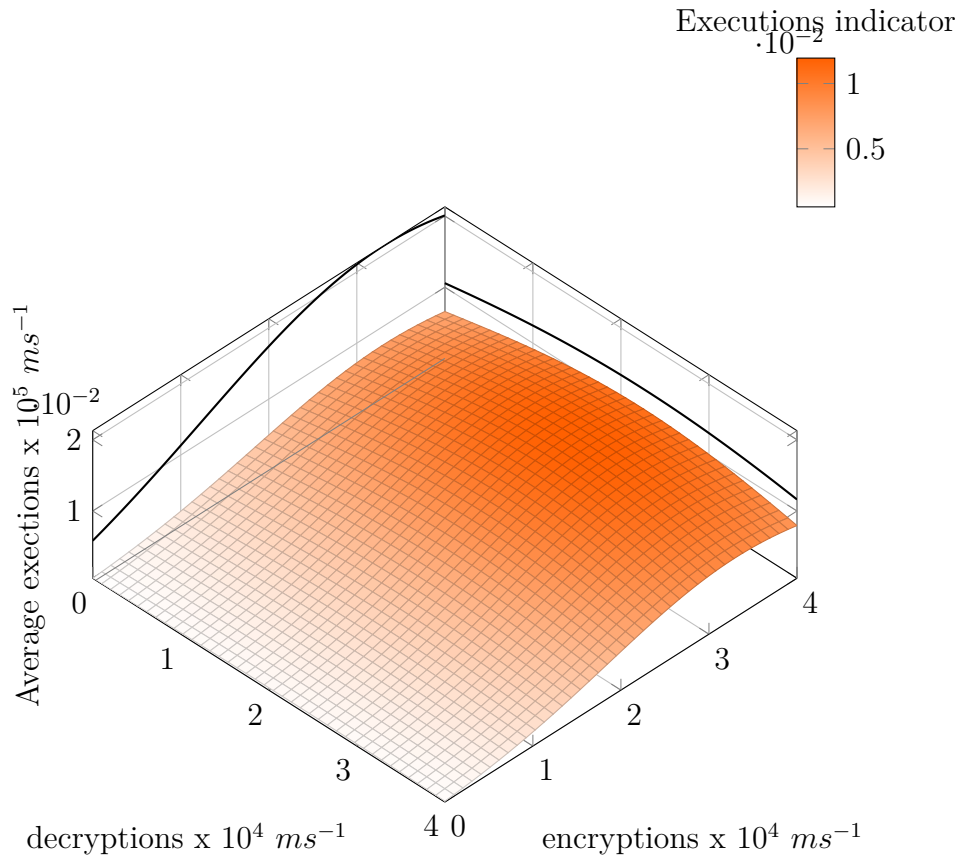


Fig. 5.18: Encryption-decryption rate of RSA based system

5.1.3 AES Encryption-Decryption Results

AES encryption and Decryption rate on a GPU based system is as follows in Fig 5.19

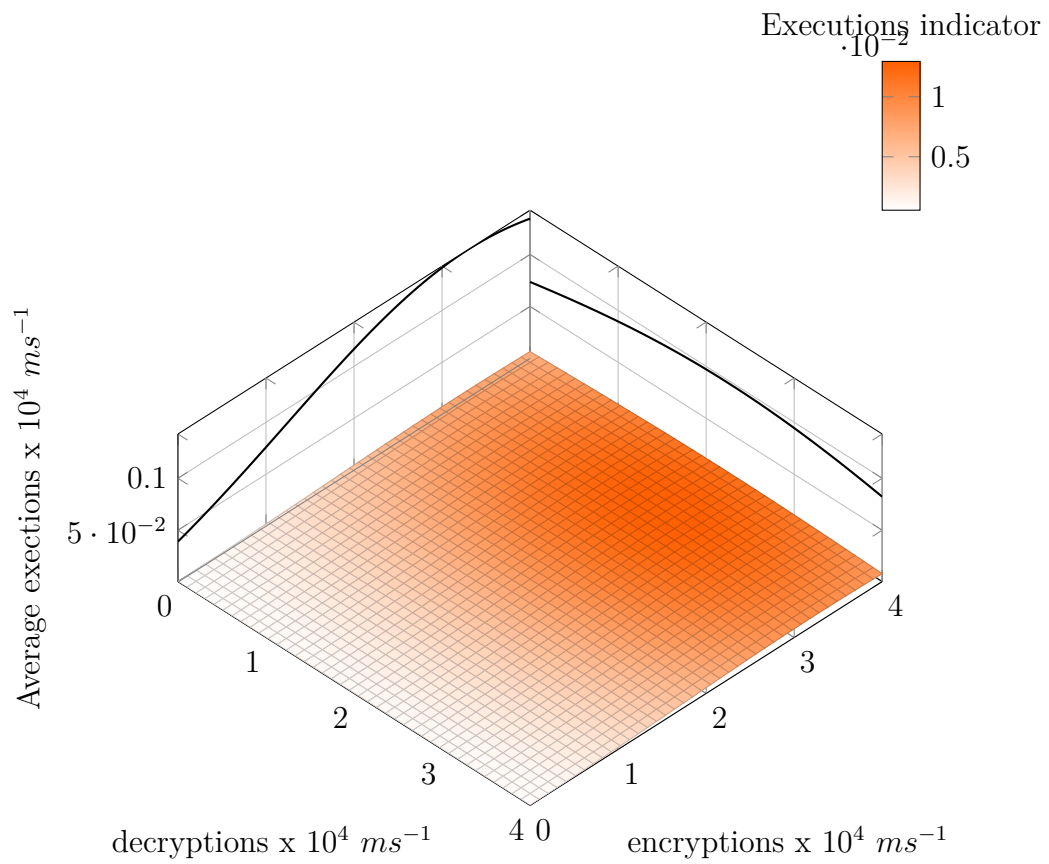


Fig. 5.19: Encryption-decryption rate of AES based system

5.1.4 3DES Encryption-Decryption Results

The Triple DES encryption and Decryption rate on a GPU based system is as follows in Fig 5.20

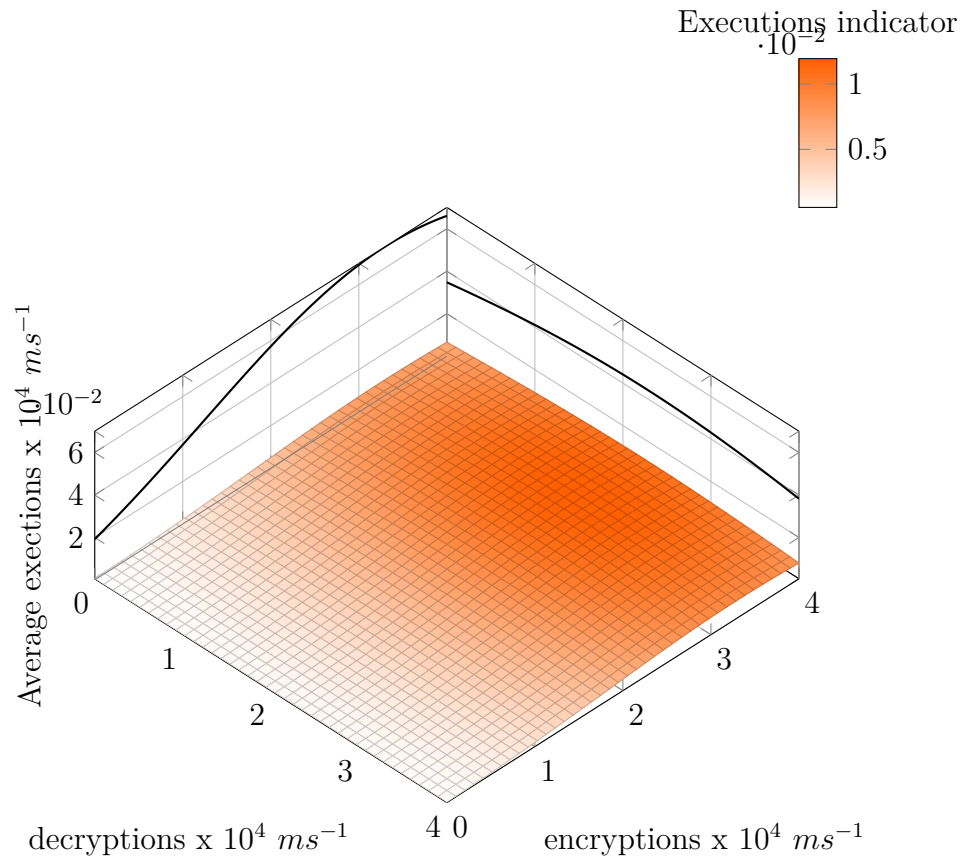




























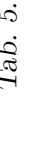
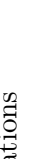


Fig. 5.20: Encryption-decryption rate of 3DES based system

The details of the machine set is in fig 5.9 and the results are shown in Tab 5.10.

System	System Specification	RAM(GB)	GPU(GB)
Machine A	Intel® Core i5-4310U CPU @ 2.00GHz Intel® HD Graphics 4400 :1696MB Windows® 7 OS Service Pack 1	4	-
Machine B	Intel® Core i5-4310U CPU @ 2.00GHz Intel® HD Graphics 4400 :1696MB Windows® 7 OS Service Pack 1	8	-
Machine C	Intel® Core i7-2670QM CPU @ 2.20GHz Intel HD Graphics 3000 Windows® 10 OS Build 15063.729	4	-
Machine D	Intel® Core i7-2670QM CPU @ 2.20GHz Intel HD Graphics 3000 Windows® 10 OS Build 15063.729	8	-
Machine E	Intel® Core i7-2670QM CPU @ 2.20GHz AMD Radeon® HD 6700M Windows® 10 OS Build 15063.729	4	1
Machine F	Intel® Core i7-2670QM CPU @ 2.20GHz AMD Radeon® HD 6700M Windows® 10 OS Build 15063.729	8	1
Machine G	Intel® Core 2 Extreme QX9650 @3.00 GHz 6M Cache Nvidia® GTX 750 TI Windows® 7 professional SP 1	8	2
Machine H	Intel® Core 2 Extreme QX9650 @3.00 GHz 6M Cache Nvidia® GTX 750 TI Windows® 7 professional SP 1	16	2
Machine I	Intel® Core i5 iMac 3.2GHz Quad-Core NVIDIA GeForce® GT 755M OS X El Capitan®	8	1
Machine J	Intel® Core i5 iMac 3.2GHz Quad-Core NVIDIA GeForce® GT 755M OS X El Capitan®	16	1
Machine K	Intel® Core i5-7400 CPU @ 3.00GHz AMD® RX 480 Windows® 7 professional SP 1®	8	4
Machine L	Intel® Core i5 -7400 CPU @ 3.00GHz AMD® RX 480 Windows® 7 professional SP 1®	16	4
Machine M	Intel® Core i7-6700k CPU @ 4.00GHz ZOTAC® 980TI Amp Extreme edition Windows® 7 professional SP 1®	16	4
Machine N	Intel® Core i7 -6700k CPU @ 4.00GHz ZOTAC® 980TI Amp Extreme edition Windows® 7 professional SP 1®	32	4
Machine O	Intel® Xeon® CPU E5-2699Av4 @2.40GHz SuSE Enterprise Server 11 NVIDIA® Tesla K40 12GB GPU Accelerator	128	12

Tab. 5.9: Compared Machine details

Performance Comparison of Encryption Algorithms on Different Technological Factors

System Specification	% of Total memory usage	Min _x	Average _x	Max _x	Trade Encrypted	Average _y	Max _y
Machine A		31.5	45.5	83.8		2.0	1.8
Machine B		26.8	48.8	91.8		1.2	1.9
Machine C		30.0	87.5	100.00		2.1	2.9
Machine D		23.6	58.5	95.7		3.6	2.1
Machine E		18	56.4	99.4		2.0	4.5
Machine F		22.3	58.4	89.3		8.4	3.8
Machine G		25.5	47.0	90.7		1.8	1.9
Machine H		26	67.9	87.6		3.2	1.5
Machine I		37.7	56.5	94.0		2.5	3.0
Machine J		25	39.8	92.4		1.9	2.6
Machine K		19.2	59.7	82.1		4.0	2.5
Machine L		24.4	46.1	89.3		3.5	1.5
Machine M		29.9	52.5	81.1		1.5	1.8
Machine N		24.4	35.4	83.2		0.9	1.1
Machine O		11.6	66.6	91.1		2.0	1.3

Tab. 5.10: AES performance in different system specifications

^x The calculations related to the system memory usage

^y Encrypted trade executions related qualitative analysis

5.2 Thread Based Performance Analysis using Amdahl's Law

5.2.1 Performance Consideration

The performance of the trading system of a parallel processor depends on complex and hard to define ways on the system's architecture and degree of parallelism in programs it executes. There are several simple performance measures such as speedup and processor utilization (efficiency) which provides rough performance estimates. The system interconnection structure and main-memory architecture also have a significant impact on performance.

Amdahl's Law is a formula for estimating the maximum speedup from an algorithm that is partially sequential and partially parallel. Following equation presented the maximum speedup that can be gained due to the parallelism that introduced from the GPU based cryptography mechanism.

From this equation speedup S_{max} can be calculated directly:

$$S_{max} = \frac{1}{(1 - P) + \frac{P}{S}}$$

- P is the percent of task that can be improved
- S is the speedup factor for the improvable part of the task

The performance results that has been previously observed were used to determine or calculate how much a computation gain that can be received using GPU environment. In trading data encryption process using parallelism procedure, it may not be mandatory for GPU based parallel computations to share data with the different partitions during the encryption of the trades in the trading system.

Newly introduced trade encryption software software is able to utilize multiple GPU cores and threads and prediction of an actual speedup gain was very difficult to estimate since the performance of different GPU models is dissimilar.

There are three options for communicating between various processes/threads running in parallel in the GPU environment. A complex mathematical calculation determine the actual speedup numbers to directly find the parallelization fraction which resulted by the newly introduced GPU device.

- Message passing - communicating with basic operations send and receive to transmit information from one computation to another.
- Shared memory - communicating by reading and writing from local memory locations that are accessible by multiple computations
- Distributed memory - some parallel computing systems provide a service for sharing memory locations on a remote computer system, enabling non-local reads and writes to a memory location for communication

5.2.2 Possible limitations in Amdahl's law based calculations

There are possible limitations in the calculated results and the reasons for the deviations are as follows.

- Not every action done in a program will have the same amount of parallelization
- Amdahl's Law only applies if the CPU is the bottleneck
- Many programs are hard-coded to use a certain number of cores
- Estimating the performance of a CPU will only be accurate for CPUs based similar architecture

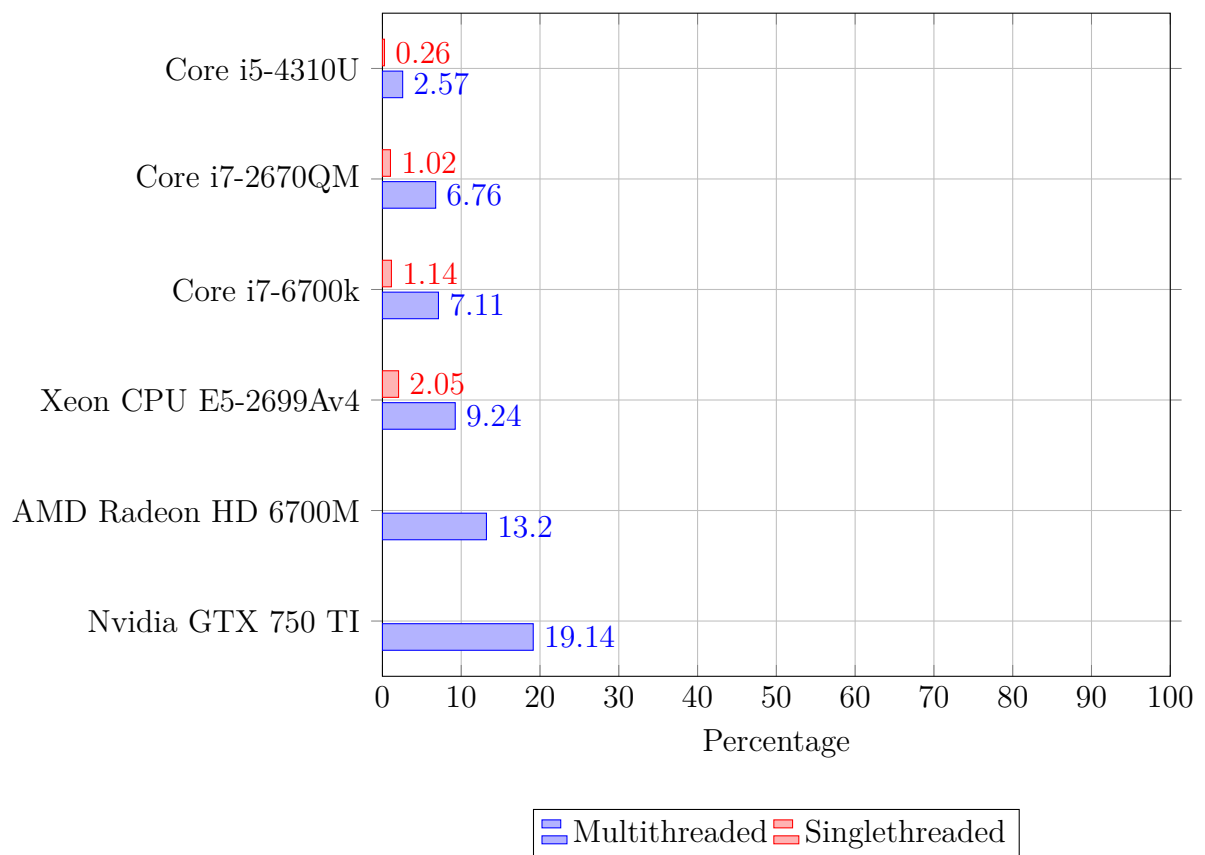


Fig. 5.21: Amdahl's performance in different system specifications

Fig 5.21 diagram shows the speed gain in multi threaded environment as follows. There are possible limitations in the calculations with related to Amdahl's law and the calculations has to be done considering the possible limitations as well.

- CPU which has multi cores were tested using “Process affinity” by enabling the least number of cores in “single threaded” testing and by enabling the most number of cores in “multi threaded” environment
- GPU performance evaluation over Amdahl's theory was tested only with the multi threaded scope

In order to visualize the results that were represented with Amdahl's law, correct assumptions has to be made as there is no communication/synchronization lag between trading firms and trade users, no copying trading data between nodes, there is a perfect load balancing and no time losses while executing the efficient parallelizations of the algorithm.

Amdahl's Law presumes that the correctness of the equation is acceptable when comparing the speedup of n processors over 1 processor of the exact same type. While calculating the results on GPU and CPU, it was considered that the multiprocessors and current multi-cores and multi-threads on GPU satisfy this property. In the real world scenario, this is not universally true. Current graphical processing systems have a host with a small number of cores of one architecture, and a highly parallel accelerator with many cores of a different architecture, often with a different instruction set.

Evaluation of Results

6.1 Introduction

At the beginning of this thesis, research has subjected to three most popular encryption algorithms according to the encryption type. AES, RSA and 3DES were the selected algorithms for the evaluation criteria. When selecting these algorithms, following key points were considered. In order to maintain acceptable security and prevention methods in this mechanism, rigorous evaluation of the system has to be done by varying multiple parameters which are directly related to the system functionalities and critical benchmarks standards that has been set up. These determinations should ensure assessing data security risk, therefore consideration should be given to the risks after presented while encryption process take place, accidental or unintentional levelling of data, loss, alteration, unauthorized disclosure of trade message fields details should be enclosed[46].

6.2 Vertical Analysis

In this section, the existing system's performances and the same system after being embedded with cryptographic component was tested in order to create a visual of market data. For the visualization of data, within the boundaries that considered for the research, data from simulated automated trading system which is employed for do the testing of various system functionality was used.

Primary focal point of doing evaluation with a vertical analysis was to ensure that all the existing functionalities in the current system are working accordingly after being connected with a cypto system. Also the latency factor was monitored rigorously throughout every test. There was an impact on the latency in a negative manner with all three crypto system. Still the rates were better than the latency rates that had been considered as the minimum required rates which were implemented by stock market experts. Some of the trading platform latency benchmarks are setup by the trading system developed vendor or the trading platform expertise themselves.

Another advantage of vertical analysis was it helped to oversee the hardware configurations that had to do in the system. 90% of the existing hardware was compatible with the new system. But in-order to gain the suggested performances, it was recommended to convert the system into GPU calculations compatible systems. Initial results showed a huge improvement in calculation done by the system after connecting the GPU hardware components.

6.3 Horizontal Analysis

Simulation results for this comparisons are shown under different parameter variations. In this case a clear vision of the superiority of the best algorithm over other algorithms in terms of the processing time, encryption rate, resource usage and speed. RSA encryption requires less time than all algorithms since the simplicity of the key generation process over multi-party communication. The key generation of the AES took more time than other two instances due to the number of rounds in the encryption process. Time consumption of RSA related executions were also less than the other 2 encryption algorithms. It was found that RSA has better performance and high throughput when compared with other two algorithms in spite of the key size used.

Encryption duration is used to calculate the throughput of an encryption scheme. It indicates the speed of the encryption algorithm used. The throughput of the encryption scheme is calculated by dividing the total plain text in Megabytes encrypted on the total encryption time for each algorithm. As the throughput value is increased, the power consumption of this encryption technique is decreased. 3DES simply extends the key size of DES by 3 times in succession with 3 different keys. Since the security perspective of the 3DES is solid due to the key size. RSA related mechanism was straightforward and provided acceptable level security with minimum latency.

6.3.1 Plain Text Encryption Process Flow

The flow chart of a trade data file while its status changed into the encrypted status is as follows in Fig 6.22. There is no data differentiate according to the trade data at this stage. the work flow remained same in all three encryption mechanisms prior to the encryption process.

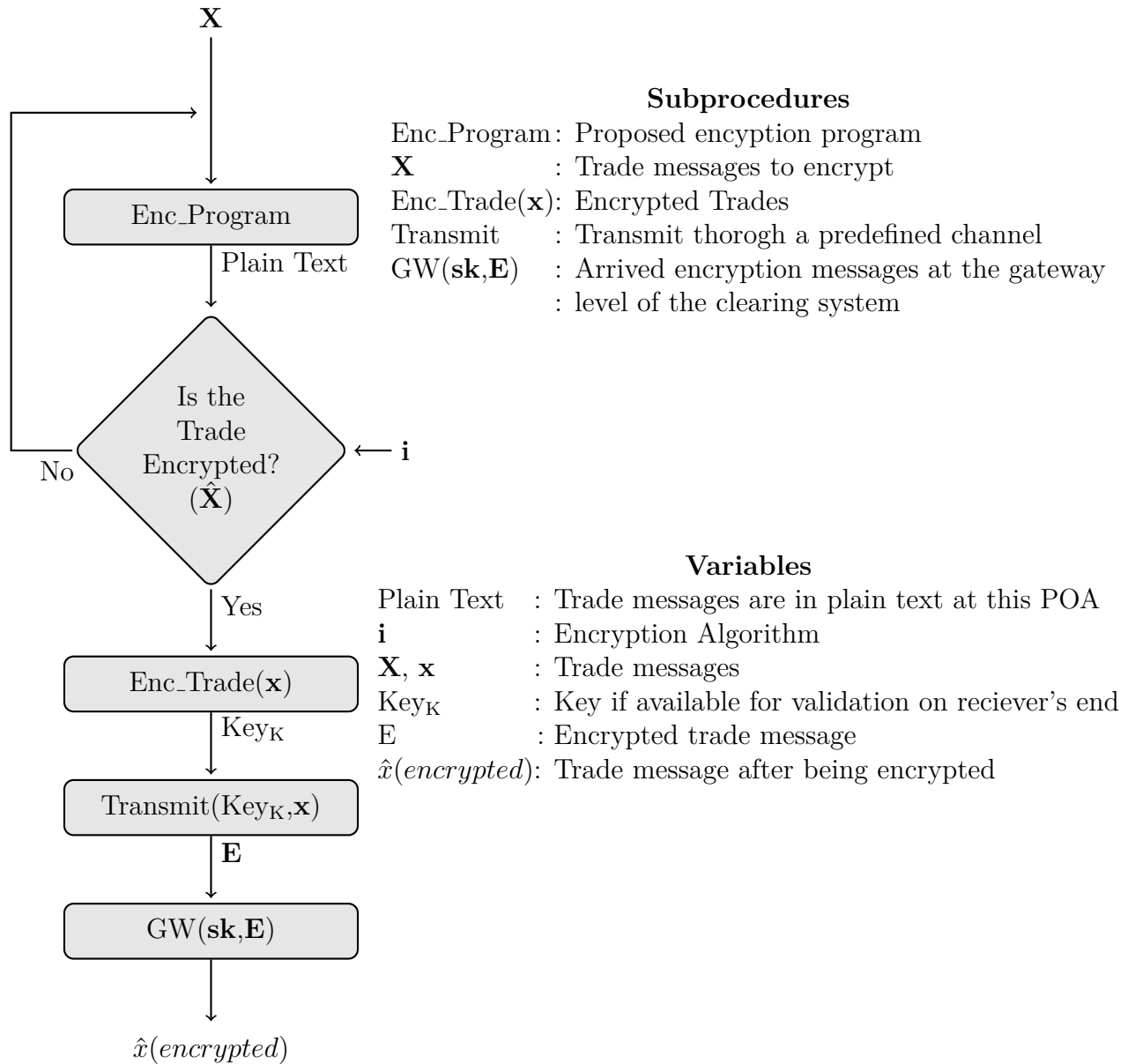


Fig. 6.22: Encryption process of the plain text data

6.3.2 Cipher Text Decryption Process Flow

The flow chart of a encrypted trade while its status changed into the decrypted status is as follows in the Fig 6.23. At this stage, original trade data has been treated with the encryption process and related decryption process triggered according to the mechanism.

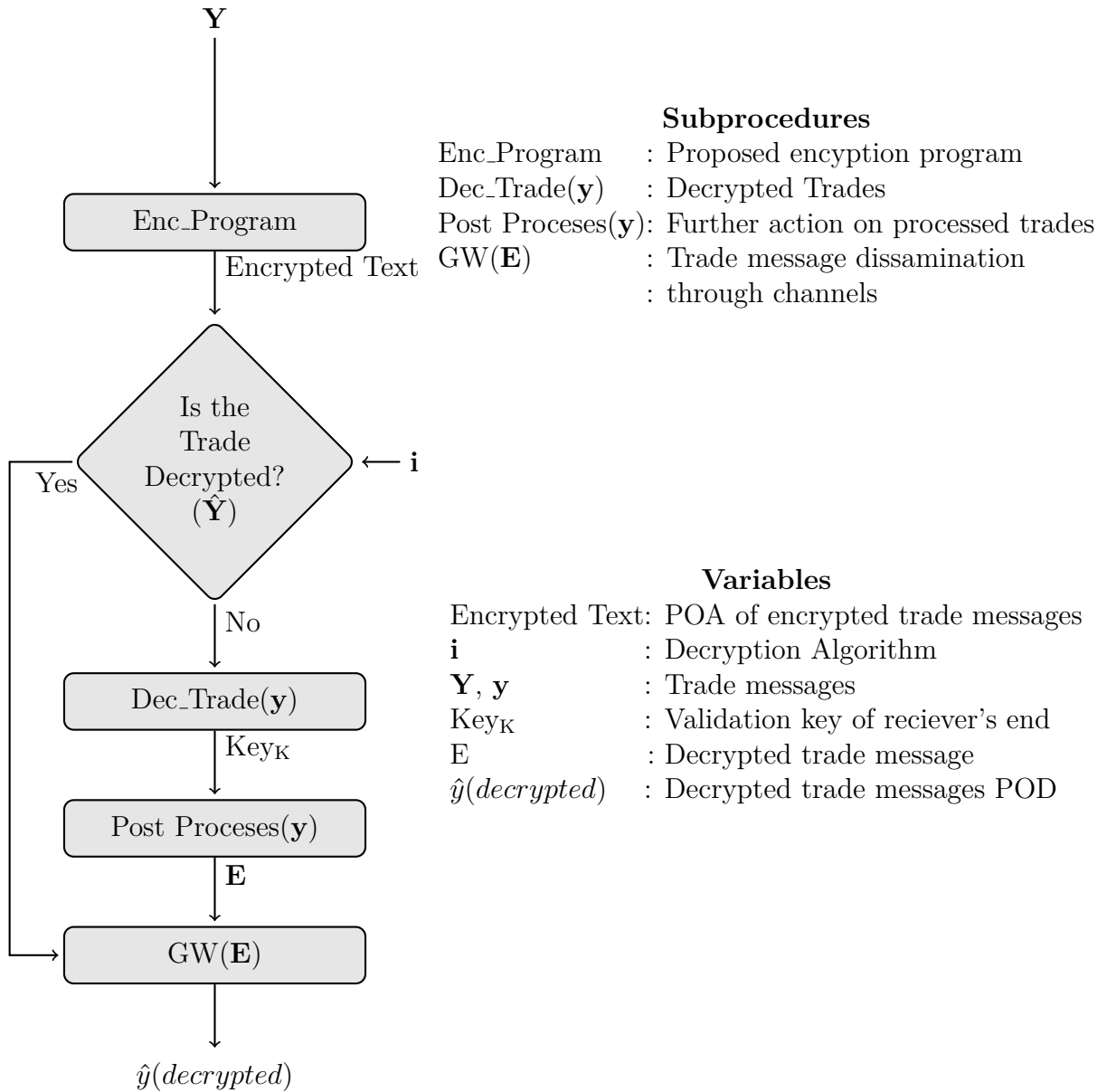


Fig. 6.23: Decryption process of the encrypted data

Future Work

When considering the trade message encryption in order to secure the transmission and to protect data, there are a number of interesting concepts which were not observed or which are only partially observed at for this research, but in order to merge this type of a solution with the production environment should perhaps be investigated in the future. As a priority work, it can be check the ability of utilizing the possibilities of using the computation power of the GPUs global memory to manage the overall calculation services of a trading platform which does not utilize the system's main memory at all, but instead to calculate the market data copies data directly to the GPU memory and save the CPU memory from some external source. One such case would be having encrypted data stored on a high speed storage solution which directly copies trade message stored memory dump from disk to the GPU, either utilizing local storage in the Peripheral Component Interconnect Express commonly known as PCI-e [47] slot connected nonvolatile memory solid state storage disks, or over high speed PCI-e network interconnects.

Non Volatile memory is another uncharted area in this document which can be study with GPU related researches. Most of the time, it can be study by common code adaptation among class libraries and high performance applications which can utilize GPUs computational power in the calculations related applications. The output data which resulted from the encryption or decryption routine could then be copied directly to the same source, or on a different high speed data storage or transmission endpoint on the GPU host, in this way it can be effectively using the GPU as a destination between two points which store and require two different states of the data, the data can be in the status of either in encrypted or decrypted. This implementation is similar to the implementations that can be found on the CPU, The CUDA libraries and the parts of the runtime may be closed source, the assembly language utilized by CUDA enabled GPUs, known as PTX instruction set, It can be another successful AES implementation as described in this thesis. This was not explored at all in this research due to the limitations with related to the resources, If there is any available resources, the research area will be an interesting task for a future scholar or a researcher who is inclined to utilize PTX.

Analysis of the PTX assembly output, available from the Nvidia CUDA C Compiler, could be analyzed to determine if there could be a performance benefit to writing the PTX manually. Also shown was the incredibly consistent throughput of GPU kernels, with low to non-existent variance in runtime given a particular input size and algorithm. Low variance means that the throughput for a given data size can be predicted to within a small margin of error, and then planned for accordingly. Financial systems, networks and communications and real-time operating systems all rely to some degree on such properties, and it would be a very interesting endeavor to explore applications for GPUs in general, not just AES, in these fields, and others. Lastly, while not strictly a research

problem, a robust system for error checking the status of kernel launches and memory copies would be a welcome addition to this application code base. For the purposes of research, a system with no error checking has been sufficient due to the completely deterministic and preplanned input sizes. However, in the real world this cannot always be assumed, and as such methods to ensure that the library alerts developer when an error has occurred is an important addition.

As a future analysis, this cryptography implementation can be applied into different real-life scenarios. After doing code level analysis furthermore, military units can adopt this message encryption system into their mission critical activities. If there is a need of delivering a command all round a military special forces unit, the same message can be transmit through the channels by defining the granularity l3v

Executive Summary

Trade management system related security has recently drawn massive public attention fuelled by the crypto currency related new technological influenced by CPU hardware evaluation and the tremendous increases in trading volumes of stocks due to the innovations powered by different stock market system vendors. By definition, there are number of connections to the market data system in order to disseminate market data to the connected parties. Most of the gateways are propitiatory gateways and a service charge or a subscription fee included when establishing the connections.

Indisputably, smooth flow of day today stock exchange system, zero security related incidents within a trade date is an important factor in stock exchanges that are driven by sophisticated technology while keeping the industry standards in place on all trade related activities on all layers of the trading data dissemination value chain. Previous discussions and researches based on the trade message security topic often based on the security of the data communication channel and there is a lack of sufficient and precise information based on the methods of encrypting data with prior to the data dissemination while minimizing the number of propitiatory trade gateways.

In order to select the appropriate encryption algorithm to develop a system base on the that, three algorithms were used and created prototype programs to observe results by varying number of factors that does have an impact on a latency factors that could be an issue and a reason for a possible system failure event. Critical analysis showed that an asymmetric encryption algorithm will befit for this scenario than a symmetric key encryption method. The number of connections that are being connected to the system and the process of key generation and key sharing mechanism backed up the selection. Between considered encryption mechanism, there was a significant gap between the results of performance and resource utilization of asymmetric key encryption and its perceived impact on market data encryption in the the trading system.

The research at hand aims to provide a solid background analysis to a problem that can be probable security gap in a market data system with emerging technology with related to hardware optimizations for calculations. This research includes identify definitions, existing functionality of the system, strategies and policies that are affected on current implementation of the system. Analytical discussion of components based on the results which have been taken by identified gaps and limitations helped to critically evaluate the new crypto based mechanism over the existing functionality. Identified impacted area through the literature review can be considered as factors to the ongoing discussions by evaluating certain proposed encryption mechanism, trying to offer new perspectives to maximize the security in existing functionality and deliver solution proposals.

After executing performance based set of test, following results were observed:

- There is a significant momentum in performance while using the RSA encryption mechanism.
- RSA encryption is less complex than other considered encryption algorithms
- Simple implementation steps
- Significant performance increase
- Less impact on latency than other encryption algorithms

Trade message encryption is an evolution of the market data dissemination over a secure information channel instead of a completely new phenomenon. There is a clear evolutionary process in the adoption of new encryption mechanism which is capable enough to give a competition, innovation and regulation to the most of the existing solutions. Market data and trade data encryption enable sophisticated market data communication mechanism can be used achieve legitimate gain on clients investments in reducing the propitiatory gateway cost while maximizing the security that in the existing system, compensation for any security breach that can be occurred in market, counter-party and operational risk exposures based on the security improvements in the trading mechanism.

Academic researches based on literature survey mostly shows positive and negative effects of encryption based strategies on market data. The majority of papers have focused on communication channel based security without considering the data encryption mechanism. On the contrary, some of the researches have argued that generally introducing a security mechanism can slow down the performance of a trading system. As trade message encryption for a stock exchange system has not been directly tested by any interest party and finding exact results for the scenario restricted by a lack of accessible and reliable data, further research on this area is highly desirable due to the technology changes.

In contrast to internalization of the trading system, encryption strategies were developed considering the relevant ASCII conversion with parallel to special character requirements. Any further assessment of encryption based strategies have to take an adaptation process in stock exchanges in different countries rather than reprogramming the approach with a code duplication. Since this mechanism is applied for different groups of market clients from firms specialized third party stakeholders, it is mandatory to communicate these proposed changes before apply it on an actual system.

Further compatibility and performance testing will be needed before performing a successful adaptation of the algorithm in to the existing production system. Therefore,

- Encryption strategies need to monitored, log and record data input and output parameters for developers testing, supervisory investigations and front-end validation
- Proposed strategy must be able to handle peak trade volumes and have capabilities of protecting data against technical failures in due to algorithm failures
- Regulators must ensure a 100% coverage for the potential systemic risks triggered by data breach and require people with specific skills and tools to introduce additional changes to the algorithm

A diagram Fig 8.24 was used to compare the indispensable properties of the market data system using a scoring mechanism before and after the changes added by the encryption program.

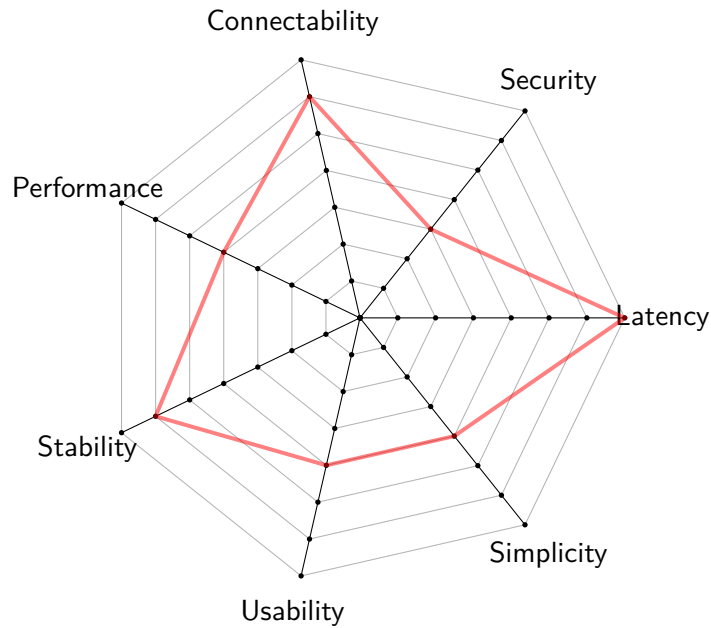


Fig. 8.24: Characteristics score diagram of the current system

After applying the cryptography encryption program on the data set. The scoring changed as follows in the Fig 8.25, Fig 8.26, Fig 8.27.

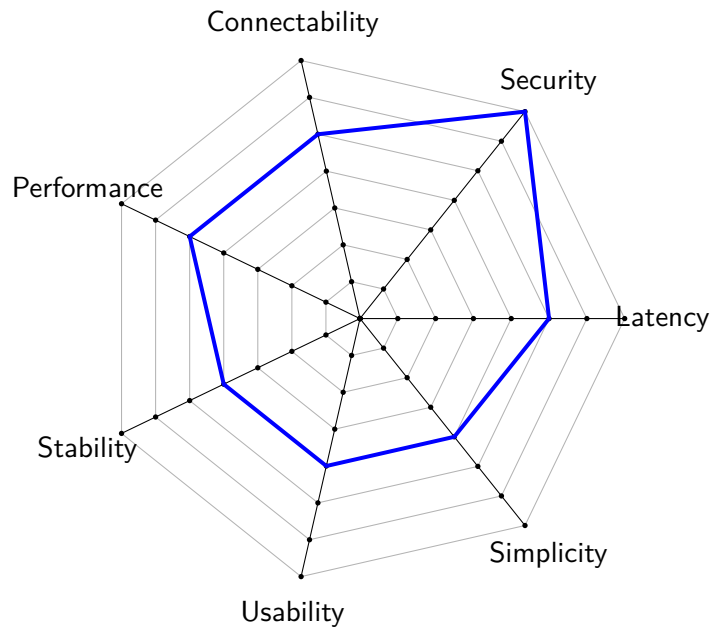


Fig. 8.25: Characteristics score diagram of the proposed RSA based system

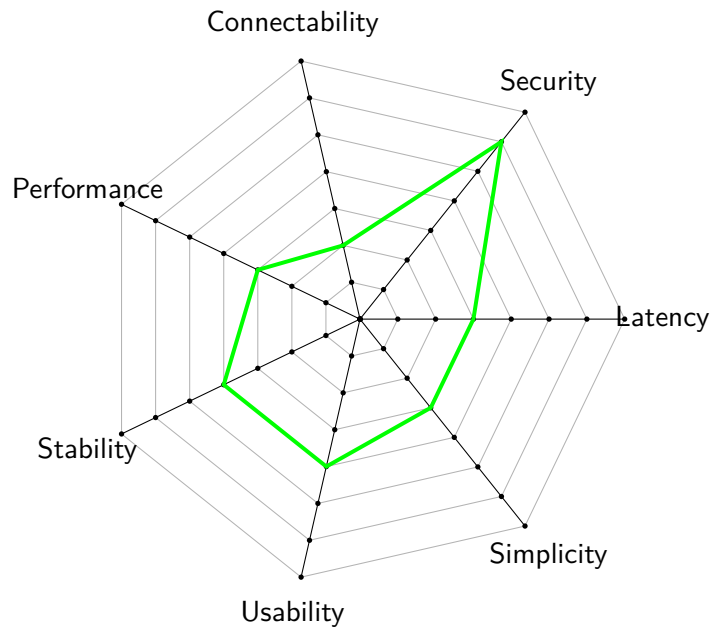


Fig. 8.26: Characteristics score diagram of the proposed AES based system

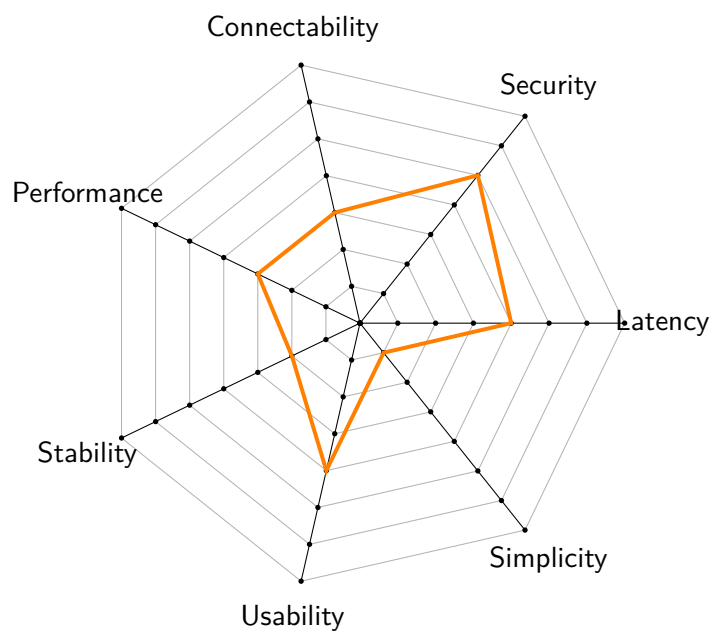


Fig. 8.27: Characteristics score diagram of the proposed 3DES based system

References

- [1] G. Kabelac, *Cyber money as a medium of exchange*. Frankfurt am Main, Federal Republic of Germany: Economic Research Group of the Deutsche Bundesbank, October, 1999, pp 18-21.
- [2] Millennium Information Technologies, “Millennium Exchange: Fast, Flexible, Multi-asset Trading Engine,” *Int. J. of Computer Science and Mobile Computing*, 2013.
- [3] J. Stapleton, *Security without Obscurity: A Guide to Confidentiality, Authentication, and Integrity*. Taylor & Francis Group, 6000 Broken Sound Parkway NW, Boca Raton: CRC Press, 2014, pp 5-10.
- [4] P. Gomber, A. Björn, M. Lutat, and T. Uhle, “High-Frequency Trading,” P.O Box 69 D-60629 Frankfurt/Main, Mar 2011, Goethe University Frankfurt.
- [5] S. William F, “Risk, Market Sensivity, and Diversification,” *Financial Analysis Journal*, 1995, pp 84-88.
- [6] T. Wyatt, G. Baddeley, N. Casey, L. Nguyen, J. Pedersen, M. Sætra, and I. Redbooks, *Secure Messaging Scenarios with WebSphere MQ*, ser. IBM Redbooks. IBM Redbooks, 2013, pp 02-08.
- [7] S. I. Inc, *SAS[®] 9.4 Intelligence Platform: Security Administration Guide*, 3rd ed. SAS Institute Inc., Cary, NC, USA: Economic Research Group of the Deutsche Bundesbank, 2016, pp 39-60.
- [8] LSEG, “Guide to the FIX 5.0 Interface to Tradelect[™],” *Associated Technical Specification Publications*, pp. 15–24, February 2010.
- [9] NASDAQ, “FIX for Orders Programming Specification for FIX 4.2,” vol. 2005, no. 07, pp. 97–105, Sept 2005.
- [10] D. Barkert, “Technical Whitepaper Kdb+ and FIX Messaging,” Tech. Rep., 2013, pp 02-08.
- [11] BSE, “BSE Exchange’s New Trading Architecture : BSE Market Data Interfaces,” Mar 2014, Manual v1.3.4.
- [12] TGHL, “Turquoise Equities TQ201 - FIX 5.0 Trading Gateway,” pp. 04–10, Sept 2013, Turquoise Global Holdings Limited.
- [13] L. Thyagaraj, C. Aranha, K. Darbha, D. Marski, R. Nicholson, J. Squibb, D. Ware, and I. Redbooks, *IBM MQ as a Service: A Practical Approach*. IBM Redbooks, 2016, pp 114-117.

-
- [14] The Depository Trust and Clearing Corporation, *MQ Standards and Technical Specifications*. DTCC, Oct 2007, pp 04-12.
- [15] B. D. Lamkin, "The Top Issues in IBM MQ and IIB," *Capitalware's MQ Technical Conference*, Sept 2013, MQ Technical Conference 2016 in Cleveland Ohio.
- [16] M. Taylor, "IBM MQ Better Application Performance," Feb 2015, IBM InterConnect Conf. held at the Mandalay Bay and MGM Grand Convention Centers in Las Vegas.
- [17] L. JSE, "Volume 00D - Trading and Information Overview for the Derivative Markets," Nov 2017, Version 1.02.
- [18] S. Fischer-Hübner, E. de Leeuw, and C. Mitchell, *Policies and Research in Identity Manage.: Third IFIP WG 11.6 Working Conference, IDMAN 2013, London, UK, April 8-9, 2013, Proceedings*, ser. IFIP Advances in Information and Communication Technology. Springer Berlin Heidelberg, 2013.
- [19] PSX, "Information Technology Security Policy for IBTS Pakistan Stock Exchange Limited," Nov 2013, Version 1.00.
- [20] TSE, "Taiwan Stock Exchange: Rules and Regulations Directory," May 2016, Information Security Policy:CC-12000 Version 1.00.
- [21] A. Caldern and S. Watkins, *SIT Governance: An International Guide to Data Security and ISO27001/ISO27002*, 5th ed. 1518 Walnut Street, Suit 1100, PA 19102,USA: Kogan Page Publishers, 2012, pp 25-29.
- [22] J. Hasbrouck, G. Sofianos, and D. Sosebee, "New York Stock Exchange Systems and Trading Procedures," April 1993, NYSE Working Paper No:93-01, pp 45-46.
- [23] P. Söderholm, "Maintenance and Continuous Improvement of Complex Systems Linking Stakeholder Requirements to the use of Built-in Test Systems," Master's thesis, Luleå University of Technology, 2005.
- [24] K. Thakur, M. Qiu, K. Gai, and M. Ali, "An Investigation on Cyber Security Threats and Security Models," *Conference: 2015 IEEE 2nd International Conference on Cyber Security and Cloud Computing (CSCloud 2015)*, 11 2015.
- [25] M. Walton. (2001, Oct) Five ways to secure your organization's information systems. Accessed 08 January 2018. [Online]. Available: <https://www.techrepublic.com/article/five-ways-to-secure-your-organizations-information-systems/>
- [26] S. Türpe, "Security Testing:Turning Practice into Theory," *IEEE Int. Conf. on Software Testing Verification and Validation Workshop*, April 2008.
- [27] L. Bauwens, W. Pohlmeier, and D. E. Veredas, *High Frequency Financial Econometrics Recent Developments*, ser. Studies in Empirical Economics. Physica-Verlag HD, 2007, pp 114-117.
- [28] Q. Tang, "Public Key Encryption Schemes Supporting Equality Test with Authorization of Differentgranularity," *Int. J. Applied Cryptography*, 2009.
- [29] L. James, "Global Information Assurance Certification paper," *SANS Institute*®, 2002.

-
- [30] S. Jajodia and C. Mazumdar, *Information Systems Security: First International Conference, ICISS 2005, Kolkata, India, December 19-21, 2005, Proceedings*, ser. Lecture Notes in Computer Science. Springer-Verlag Berlin Heidelberg: Springer Berlin Heidelberg, 2005.
- [31] S. Sharma and M. A. Hasamnis, “Enhancing Security of Text using Modified Encryption Algorithm,” *Int. J. of Eng. Trends and Technology (IJETT)*, vol. 36, no. 3, pp. 111–115, June 2016.
- [32] S. R. Masadeh, S. Aljawarneh, N. Turab, and A. M. Abuerrub, “A Comparison of Data Encryption Algorithms with the Proposed Algorithm: Wireless Security,” in *The 6th International Conference on Networked Computing and Advanced Information Management*, Aug 2010, pp. 341–345.
- [33] C. Williams, “Research Methods,” *Journal of Business & Economic Research*, vol. 5, no. 3, pp. 65–72, Mar 2007.
- [34] M. V. Pawar and J. Anuradha, “Network Security and Types of Attacks in Network,” *Procedia Computer Science*, vol. 48, pp. 503 – 506, 2015, International Conference on Computer, Communication and Convergence (ICCC 2015).
- [35] S. Du, J. Wang, and K. Gwebu, “Stock market reaction to data breaches: The moderating role of corporate social responsibility,” *Int. Conf. on Cyber Situational Awareness, Data Analytics and Assessment (Cyber SA)*, pp. 1–2, 06 2017.
- [36] K. D. Washington, “The Impact of Data Breaches on Market Value of Firms in the E-Commerce Marketplace,” Master’s thesis, Capella University, January 2016, Ph.D. dissertation.
- [37] H. O. Alanazi, B. B. Zaidan, A. A. Zaidan, H. A. Jalab, M. Shabbir, and Y. Al-Nabhani, “New Comparative Study Between DES, 3DES and AES within Nine Factors,” *Journal of Computing*, vol. 2, no. 3, pp. 152–157, Mar 2010.
- [38] T. S. Landgrave and N. D. Jorstad, “Cryptographic Algorithm Metrics,” *20th National Information Systems Security Conference*, 10 1997.
- [39] P. Jindal and B. Singh, “Study and Performance Evaluation of Security-Throughput Tradeoff with Link Adaptive Encryption Scheme,” *I.J. Computer Network and Information Security, 2013*, vol. 1, pp. 49–55, 11 2013.
- [40] V. Nazaruk and P. Rusakov, “Implementation of Cryptographic Algorithms in Software: An Analysis of the Effectiveness,” *Scientific Journal of Riga Technical University Computer Science. Applied Computer Systems*, vol. 43, pp. 97–105, 2010.
- [41] K. Kalaiselvi and A. Kumar, “Implementation Issues and Analysis of Cryptographic Algorithms Based on Different Security Parameters,” *International Conference on Current Trends in Advanced Computing (ICCTAC)*, no. 2, pp. 23–28, May 2015.
- [42] S. Aishwarya and K. R. Dhivya, “Online Payment Fraud Prevention Using Cryptographic Algorithm TDES,” *Int. J. of Computer Science and Mobile Computing*, vol. 4, no. 4, pp. 317–323, April 2015.
- [43] T. Farhana, M. I. I. Bappy, S. C. Prince, and D. Yasmeen, “Secured Message Transaction System with Strong Security Services,” *Int. J. of Computer Science and Mobile Computing*, vol. 5, no. 2, pp. 139–145, Feb 2016.

-
- [44] I. Kashkynbek and B. Umurzakov, “Increasing a speed of data exchange in the processes using the protocol FIX,” *The 13th International Scientific Conference Information Technology and Management 2015*, vol. 10, p. pp 94, April 16-17, 2015, Information Systems Management Institute, Riga, Latvia.
- [45] P. Wang and T. Moore, “Sudden Changes in Volatility: The Case of Five Central European Stock Markets,” *Journal of International Financial Markets, Institutions and Money*, vol. 19, pp. 33–46, 02 2009.
- [46] CSE, *Listing Rules*. Colombo Stock Exchange, 2016, Section 8 Corporate Disclosure.
- [47] R. Bittner, E. Ruf, and A. Forin, “Direct GPU/FPGA communication via PCI express,” *Cluster Computing*, vol. 17, no. 2, pp. 339–348, Jun 2014.

Appendix A

Class Diagram

The key purpose of developing this class diagram is to highlight the key relationships between systems' critical data classes within the trading system boundaries. Most of the additional feature classes have been omitted since the primary focus of drawing this diagram was to derive the changes which were introduced by the newly developed components.

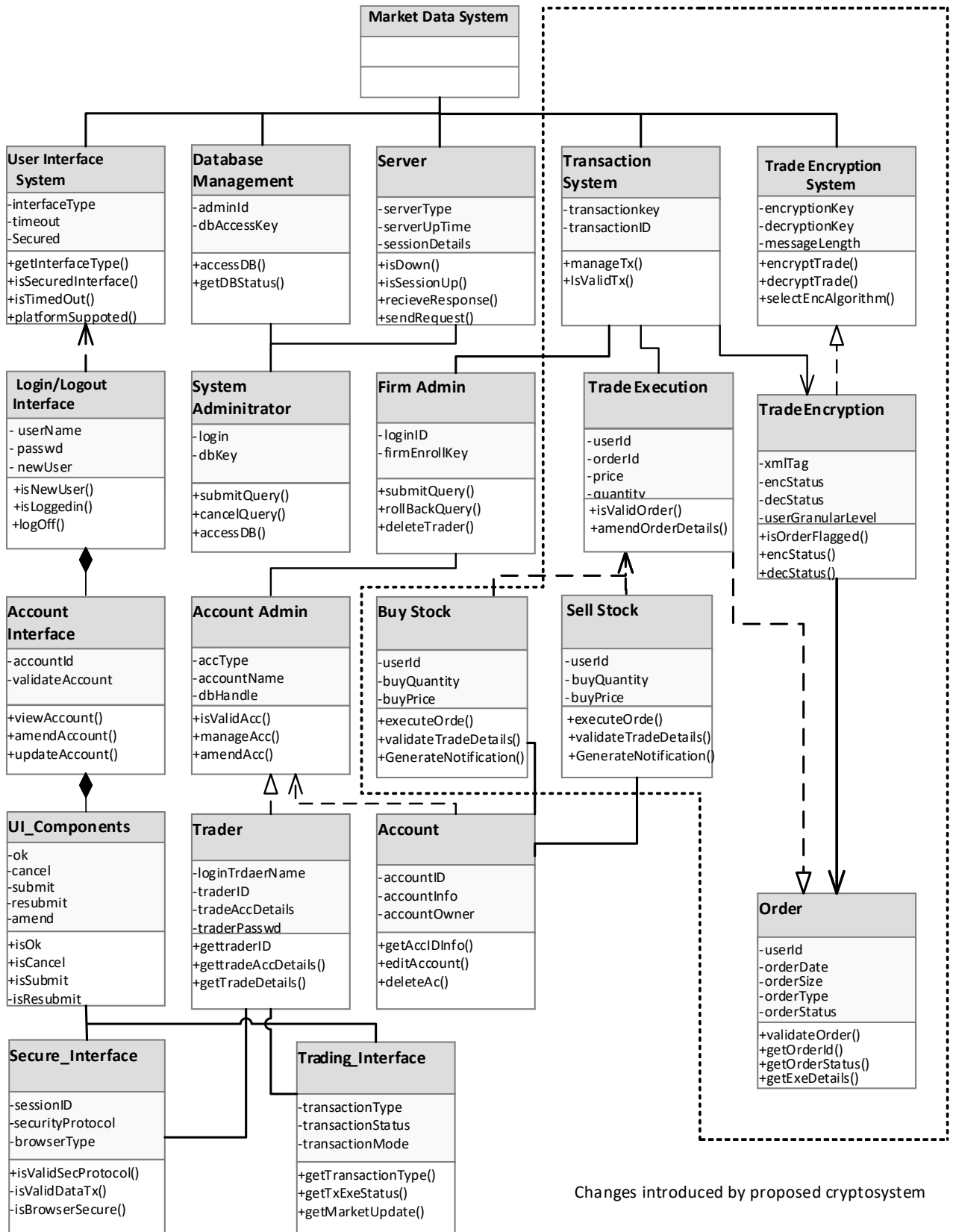
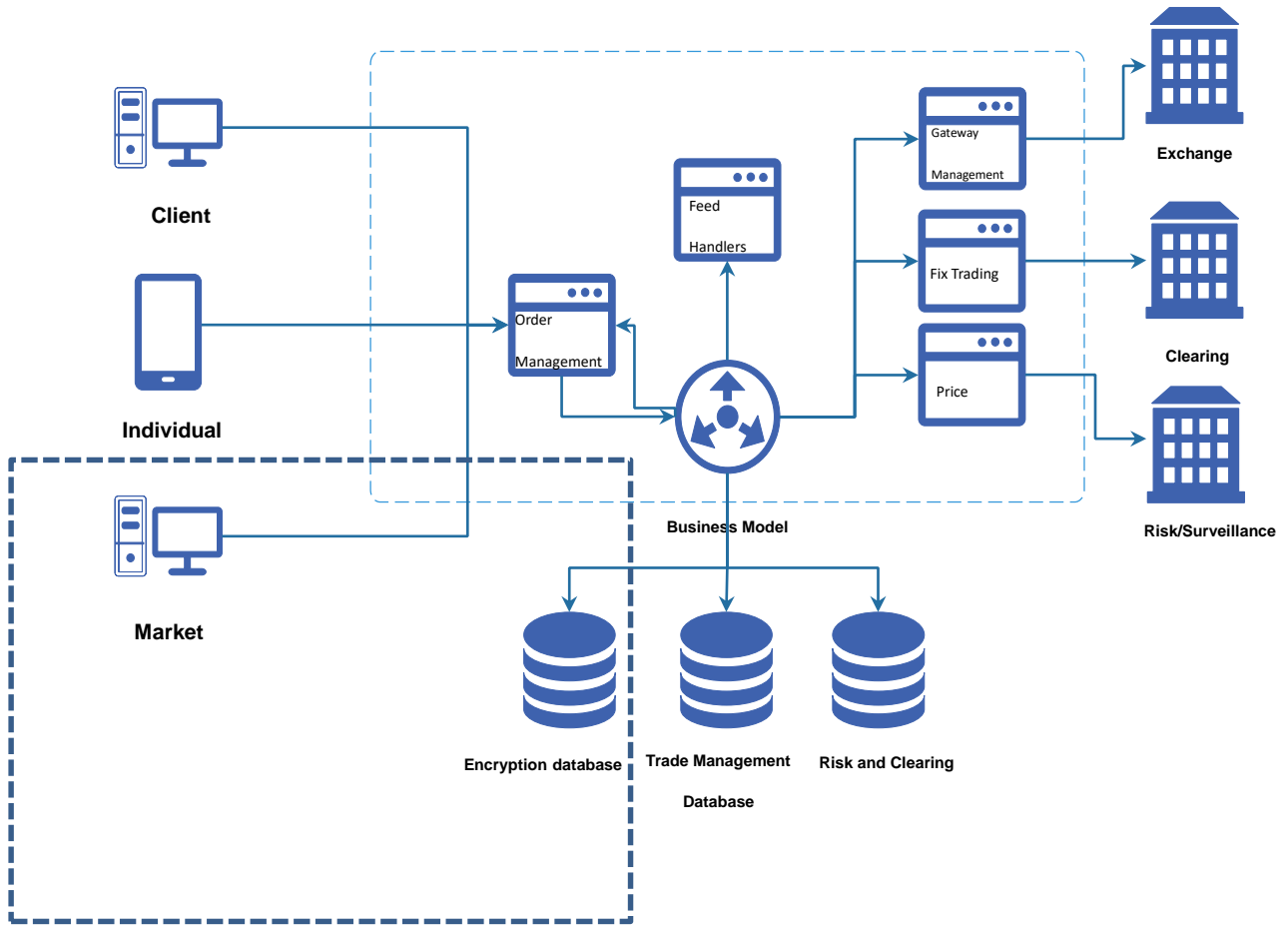


Fig. A.1: Highlighted changes in the class diagram

Appendix B

Architecture Diagram

In order to create a visibility and consensus across different parties in a trading system, an architectural diagram was designed. There are few changes that has been introduced by the encryption program to the architecture of the existing trading system. In software development project, there is a need to create architectural diagrams in order to forecast the additional resources that might be needing for the implementation of the new components.



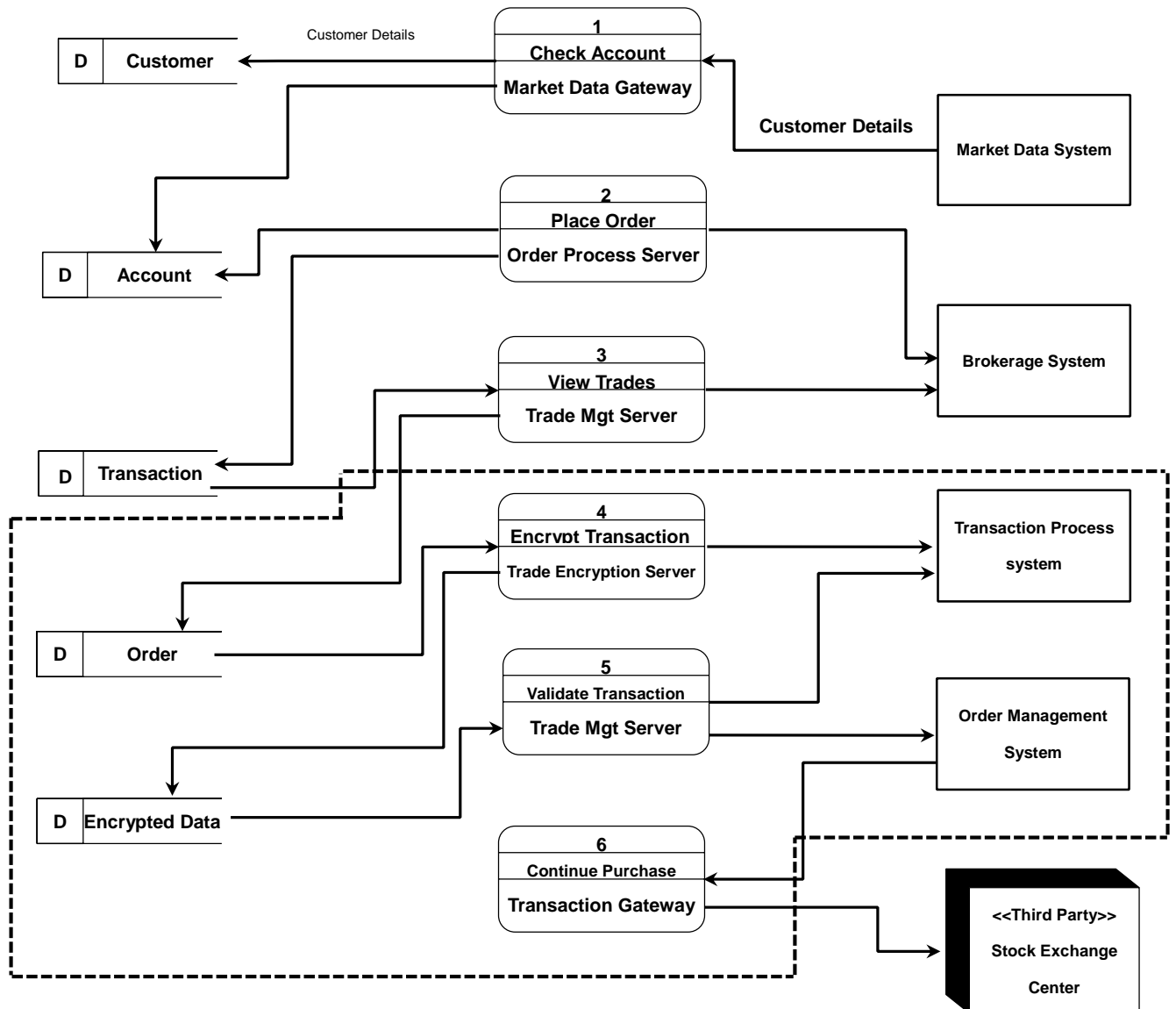
Changes introduced by proposed crypto system

Fig. B.1: Highlighted changes in the Architecture diagram

Appendix C

Data Flow Model

This data flow diagram interpret the flow of data in a trading system as well as the successive transformations of the trade messages between several system component. The data flow diagram is a graphical representation that arrange the flow of data in an understandable manner in a non-technical way. This diagram was used to select the most suitable system area that can use to apply the security mechanism. The sensitivity of the data was considered while developing the algorithm.



Changes introduced by proposed crypto system

Fig. C.1: Highlighted changes in the Data flow diagram

Appendix D

Sequence Diagram

Previously defined use case diagram describes how the external actors interact with trading system. Each interaction generates events requesting some operation in response. Changes were introduced in the existing test case model showed the new processes which involved with the changes. Behavioral driven modelling is a newly introduced development technique and it helps to design conditions, logic and code behavior additional techniques. Investigation process in which input and output events the systems used is shown in the Sequence Diagram. This is an extended representation of the details which were not represented by the class diagram. Due to the new changes in process, message sequence has also been affected. This diagram visualize the changes in the sequence.

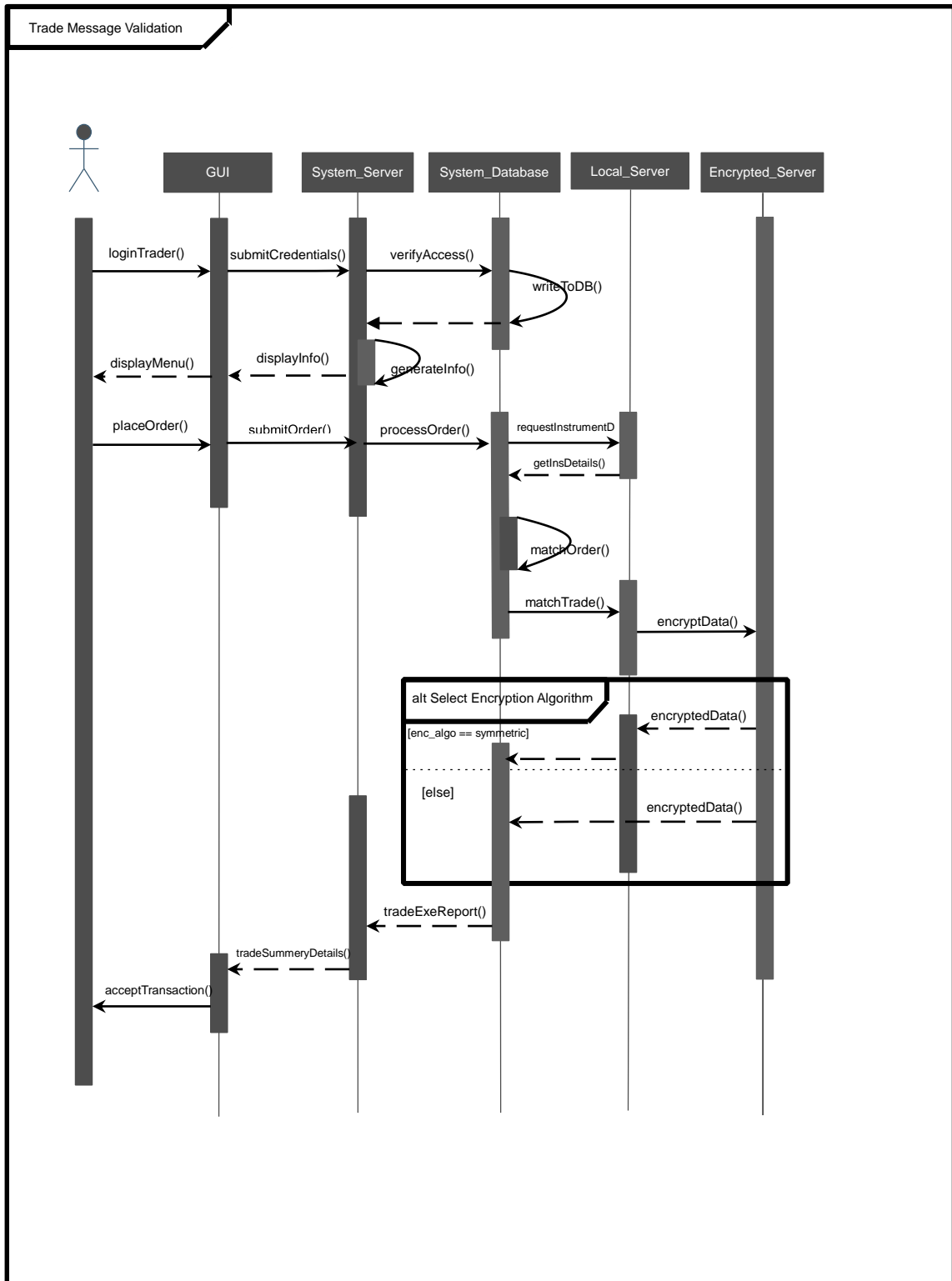


Fig. D.1: Highlighted changes in the Sequence diagram

Appendix E

Terms Related to Stock Markets¹

- Trade

The transfer of ownership of stocks from one person or entity to another entity. Trading process takes place on all days of the week (except Saturdays and Sundays and holidays declared by the Exchange in advance).

- Clearing

Clearing firm handles the back-end details of transactions between buy and sell parties. This makes the settlement process streamlined and efficient.

- Settlement

The settlement of trades takes place on T +0, T+3 working day based on the selection of the traded parties.

- Trading system/Exchange system

Trading System provides an automated trading platform for screen-based, the concept of floorless trading and an online surveillance, monitoring and clearing mechanism. The system supports transparency of trading operations on an order driven market . Opening Auction call- This session occurs before start of trading

- Market Start time

0900 hrs

¹Example times may vary from stock exchange to stock exchange and time zones

- Market close time

16030 hrs

- Trade modification and market end time

1615 hrs

- Order

A set instructions received from customers to broker firm to buy or sell stocks.

Appendix F

RSA based solution

Listing F.1: main.cpp

```
/*  
|Binoy Karunaratne |  
|2015mis014 |  
|Trade Message Enc via RSA encryption |  
|UCSC |  
|2017 |  
|_____*/  
#define _WIN32_WINNT 0x0500  
//_WIN32_WINNT is a preprocessor token, which is replaced by 0x0500  
//The preprocessor just scans the whole file and replaces _WIN32_WINNT with (0x0500)  
→ everywhere it is found.  
#include <iostream> //This header is part of the Input/output library.  
#include <fstream> //this defines three new data types.(ofstream, ifstream, fstream)  
#include <tchar.h> //Previously there was only ASCII codes (OR Multi byte character set).  
//Now, All programming languages allows coding to be in Unicode for Internationalization Issue  
→ .  
#include <stdlib.h> //The stdlib.h header defines functions involving memory allocation,  
→ process control, conversions and others  
#include <windows.h> // declarations for all of the functions in the Windows API, all the  
→ common macros used by Windows programmers.  
#include columns"columnsEncProtocol_01columns.columnshcolumns" //By Including this file, we  
→ will be able to use the functions inside the file  
#include columns"columnsEncProtocol_02columns.columnshcolumns" //By Including this file, we  
→ will be able to use the functions inside the file  
#include columns"columnsAuxiliarcolumns.columnshcolumns" //Include the main functions auxiliar  
using namespace std; // used to define a scope and allows us to group global classes,  
→ functions and/or objects into one group.  
int main(){  
_tsetlocale(LC_ALL, _T(columns"columns")); //Can define a language that we need to convert the  
→ data into ("en_US.utf8", or any other .utf8)  
signed long long int key, mdc=0, e=4, phi, d, msgNUM[10000], encode[10000], decode[10000], blocks  
→ [10000], de_blocks[1000], i, j, p, q, op, newblocks[10000];  
//target type will have width of at least 64 bits.  
char msgTEXT[10000], msgDECO[10000], temp_val[10000];  
ifstream input(columns"columnsinputcolumns.columnstxtcolumns");  
ofstream temp_dec_blocks(columns"columnsblockscolumns.columnstempcolumns");  
ofstream encrypted_blocks(columns"columnsenccolumns.columnstempcolumns");  
cout<<endl<<columns"columns====columns_columnsrSAcolumns_  
→ columnsimpementationcolumns_columnsbycolumns_columnsbinoycolumns_columnskarunaratne  
→ columns_columns====columns"<<endl<<endl;  
cout<<endl<<columns"columns====columns_columnsenccolumns_columnsprocesscolumns_  
→ columnsdescribedcolumns_columnsincolumns_columnsenccolumnsEncProtocol_02columns_  
→ columns====columns"<<endl<<endl;  
main_function_EP01(&key, &p, &q); //Calling out the main function from EndProtocol_02 file.  
// system("cls"); // if the display is making the scrip slow, this can be removed at each time  
→ when the screen is moving on to a cmd window.  
cout<<endl<<columns"columns====columns_columnsrSAcolumns_  
→ columnsimpementationcolumns_columns====columns"<<endl<<endl;  
cout<<endl<<columns"columns====columns_columnsgeneratedcolumns_columnspriescolumns:columns_  
→ columnspcolumns_columns=columns_columns"<<p<<columns"columns_columnsecolumns_columnsq  
→ columns_columns=columns_columns"<<q<<columns"columns, columns_columnsncolumns_columns=
```

```

    ↪ columns_columnspcolumns_columns*columns_columnsqcolumns_columns,columns_columnsn
    ↪ columns_columns=columns_columns"<<key<<columns"columns_columns====columns"<<endl;
cout<<endl<<columns"columns==columns_columnsApplyingcolumns_columnssthecolumns_columnsEuclidean
    ↪ columns_columnsAlgorithmcolumns_columnsfromcolumns_columnssthecolumns_columns
    ↪ columns_EncProtocol_01columns_columnssoncolumns_columnssthecolumns_columns_keycolumns_
    ↪ columnstocolumns_columnscalculatecolumns_columns'columnsPhicolumns'columns_columns
    ↪ columns_columns'columnsecolumns'columns_columns=columns"<<endl<<endl;
phi=(p-1)*(q-1);
while(mdc!=1){
e++;
mdc=euclidean_alg_EP_01(phi,e);
}
d=calculation_d(e,phi);
if(d<0){
d=d+phi;
}
cout<<columns"columnsPhicolumns_columnsincolumns_columns(columns"<<key<<columns"columns)
    ↪ columns_columns=columns_columns"<<phi<<columns"columns"<<endl<<endl;
cout<<columns"columns_keycolumns_columnsPubliccolumns_columns(columnsncolumns,columnsecolumns)
    ↪ columns\columnsncolumns_columns_columns(columns"<<key<<columns"columns_columns,columns_
    ↪ columns"<<e<<columns"columns)columns_columns_columns_columns_columns_columns_
    ↪ columns_columns_columns_columns_columns->//columns_columns'columns"<<e<<columns"
    ↪ columns'columns_columnsiscolumns_columnsPubliccolumns_columnsvaluecolumns_columnsssuch
    ↪ columns_columnssthatcolumns_columnsmdccolumns(columns"<<e<<columns"columns,columnsphi
    ↪ columns)columns_columns=columns_columnsiscolumns"<<endl<<endl;
cout<<columns"columns_keycolumns_columnsPrivatecolumns_columns(columnsncolumns,columnsdcolumns)
    ↪ columns\columnsncolumns_columns_columns(columns"<<key<<columns"columns_columns,columns_
    ↪ columns"<<d<<columns"columns)columns_columns_columns->//columns_columns"<<d<<columns"
    ↪ columns_columnsiscolumns_columnssthecolumns_columnsinversecolumns_columnssofcolumns_
    ↪ columns'columns"<<e<<columns"columns'columns_columnssemcolumns_columns(columnsmod
    ↪ columns_columns"<<phi<<columns"columns)columns";
cout<<endl<<endl<<columns"columns====columns_Presscolumns_columnsEntercolumns_columnsstocolumns_
    ↪ columns_readcolumns_columnssthecolumns_columns_Inputcolumns_columns_Filecolumns====columns"
    ↪ <<endl<<endl;
system(columns"columnspausecolumns");
if(input==NULL){
cout<<endl<<endl<<columns"columns###columns_columnsInputcolumns_columnsfilecolumns_columns'
    ↪ columnsinputcolumns.columnstxtcolumns'columns_columnsnotcolumns_columnsfoundcolumns_
    ↪ columnsorcolumns_columnswithoutcolumns_columnsstextcolumns_columns###columns"<<endl<<
    ↪ endl;//This has been defined to handle the exception cases when there is no files.
system(columns"columnspausecolumns");
exit(0);
}else{
system(columns"columnsciscolumns");
//REDIMENSION THE ALGORITHM EXECUTION SCREEN #####
HWND console = GetConsoleWindow();
RECT r;
GetWindowRect(console, &r);
MoveWindow(console, r.left, r.top, 650, 650, TRUE);
cout<<endl;
//#####
i=0;
printf(columns"columnsMessagecolumns_columnsreadcolumns_columnsfromcolumns_columnsfilecolumns:
    ↪ columns_columns\columnsncolumns\columnsncolumns");
while(input.get(msgTEXTO[i])){
cout<<msgTEXTO[i]; //Read the MESSAGE FROM THE FILE AND AT THE SAME TIME PASS IT TO THE VECTOR
    ↪ msgTEXT AND PRINT ON THE SCREEN
i++;
}
cout<<endl<<endl<<columns"columns====columns_Presscolumns_columnsEntercolumns_columnsstocolumns_
    ↪ columnsenryptcolumns_columnssthecolumns_columnsabovecolumns_columnsmessagecolumns====
    ↪ columns"<<endl<<endl<<endl<<endl<<endl<<endl<<endl<<endl<<endl<<endl<<endl;
system(columns"columnspausecolumns");
system(columns"columnsciscolumns"); //REPAIR: YOU MUST MAKE A PRE-CODE, THE BLOCKS CAN NOT BE
    ↪ THE CHARACTERS IN ANSI, THE BLOCKS MUST BE THE UNION, SINCE THE UNION IS <key,
i=0; //zero counter
do{//PROCESS TO CONVERT THE TEXT TO WHOLE AND APPLY IN THESE ENTITIES THE RSA ENCRYPTION
    ↪ ALGORITHM
msgNUM[i]=convert_text_to_number(msgTEXTO[i]); //CONVERT THE INSERTED MESSAGE TO WHOLE ANSI
    ↪ TABLE
blocks[i]=msgNUM[i]; //message encoded in a file
i++;
}while(msgTEXTO[i]); //CONTINUE TO CONTINUE IN THE PROCESS WHILE IT IS NOT THE END OF THE FILE
i=0; //zero counter
j=0; //zero counter
while(blocks[i]){ //THIS PROCESS WILL GENERATE THE BLOCKS LESS THAN N

```

```

if(!blocks[i+2]){
newblocks[j]=(blocks[i]*100)+(blocks[i+1]);
}else if(!blocks[i+1] && !blocks[i+2]){
newblocks[j]=blocks[i];
}else{
newblocks[j]=(blocks[i]*10000)+(blocks[i+1]*100)+(blocks[i+2]);
}i=i+3;
j++;
}
system("columns"columns"columns");
cout<<endl<<columns"columns===columns_columnsGENERATEDcolumns_columnsBLOCKScolumns_columns===
↳ columns"<<endl<<endl;
i=0;
while(newblocks[i]){ //PRINTING THE BLOCKS GENERATED ON THE SCREEN
cout<<newblocks[i]<<columns"columns_columns";
temp_dec_blocks<<newblocks[i]<<columns"columns_columns";
i++;
}
cout<<endl<<endl<<columns"columns==columns_columnsTHESEcolumns_columnsABOVEcolumns_columnsARE
↳ columns_columnsGENERATEDcolumns_columnsBLOCKScolumns_columns==columns"<<endl<<endl<<
↳ endl<<endl<<endl<<endl<<endl<<endl<<endl<<endl<<endl;
Sleep(5000); //Wait 5 seconds and then go to the next step
i=0;
do{ //PROCESS TO CODE GENERATED BLOCKS
encode[i]=encoded_value(newblocks[i],e,key); //CODE THOSE WHOLE NUMBERS
i++;
}while(newblocks[i]);
i=0;
system("columns"columns"columns");
cout<<endl<<columns"columns===columns_columnsCODEDcolumns_columnsBLOCKScolumns_columns===
↳ columns"<<endl<<endl;
while(encode[i]){
cout<<encode[i]<<columns"columns_columns";
encrypted_blocks<<encode[i]<<columns"columns_columns";
i++;
}
cout<<endl<<endl<<columns"columns=====columns_columnsTHESEcolumns_columnsABOVEcolumns_
↳ columnsAREcolumns_columnsCODEDcolumns_columnsBLOCKScolumns_columns=====columns"<<endl<<
↳ endl;
cout<<endl<<endl<<columns"columns=====columns_columnsINITIATINGcolumns_columnsDECODINGcolumns_
↳ columnsPROCESScolumns_columnsOFcolumns_columnsTHEcolumns_columnsABOVEcolumns_
↳ columnsBLOCKScolumns_columns=====columns"<<endl<<endl<<endl<<endl<<endl<<endl<<endl<<endl;
Sleep(5000); //Wait 5 seconds and then go to the next step
system("columns"columns"columns");
i=0;
j=0;
cout<<endl<<endl<<columns"columns=====columns_columnsTHIScolumns_columnsIScolumns_
↳ columnsTHEcolumns_columnsMESSAGEcolumns_columnsBEINGcolumns_columnsDECODEDcolumns_
↳ columns=====columns"<<endl<<endl;
system("columns"columnscolorcolumns_columns0columnsAcolumns");
do{//PROCESS TO DECODE THE VALUES GENERATED ABOVE AND CONVERT THEM TO TEXT
decode[i]=decoded_value(encode[i],d,key); //DECODES BLOCKS ENCRYPTED IN BLOCKS NOT ENCODED
//PROCESS FOR DIVIDING THE GREAT BLOCKS IN COMBINATIONS OF 2 VALUES THAT WILL BE RECONVERTED
↳ TO THE CHARACTERS
if(decode[i]/100>=0 && decode[i]/10000>0 ){
de_blocks[j]=decode[i]/10000;
de_blocks[j+1]=(decode[i]/100)-(de_blocks[j]*100);
de_blocks[j+2]=(decode[i])%100;
}else if(decode[i]/100>=0 && decode[i]/10000<=0 ){
de_blocks[j]=decode[i]/100;
de_blocks[j+1]=(decode[i])%100;
}else if(decode[i]/100<=0 && decode[i]/10000<=0 && decode[i] >=0){
de_blocks[j]=decode[i];
}
msgDECO[j]=convert_numbers_to_text(de_blocks[j]); //CONVERT THE MESSAGE OF INTEGERS TO
↳ CHARACTERS
msgDECO[j+1]=convert_numbers_to_text(de_blocks[j+1]); //CONVERT THE MESSAGE OF INTEGERS TO
↳ CHARACTERS
msgDECO[j+2]=convert_numbers_to_text(de_blocks[j+2]); //CONVERT THE MESSAGE OF INTEGERS TO
↳ CHARACTERS
cout<<msgDECO[j]<<msgDECO[j+1]<<msgDECO[j+2]; //IMPRIMI FEATURES CHARACTERS
j=j+3;
i++;
Sleep(5);
}while(encode[i]);

```

```
cout<<endl<<endl<<endl;
system(columns"columnspausecolumns");
}}
```

Listing F.2: Auxilar.cpp

```

#include <string> "columnsEncProtocol_01columns.columnshcolumns" //This will make EncProtocol to
    ↪ use with all the functions that has been defined in the file
//all of the mentioned functions are addressed from the EncProtocol_01.cpp file.
signed long long int calculation_d(signed long long int e, signed long long int phi)//define
    ↪ the bit type as signed integral type which is having at least 64 bits.
{
signed long long int d;
d=euclidian_alg_encl(phi,e);
return d;
}
//Function that calculates the mod n, this is a ^ e mod n
signed long long int encoded_value(signed long long int msg, signed long long int e, signed
    ↪ long long int key)
{
signed long long int encode;
encode=power_calc_encl(msg,e,key);
return(encode);
}
//Function that calculates the decoded values in the received files.
signed long long int decoded_value(signed long long int encode,signed long long int d, signed
    ↪ long long int key)
{
signed long long int decode;
decode=encoded_value(encode,d,key);
return decode;
}
signed long long int convert_text_to_number(char converted_msg)
{
signed long long int convert_txt_to_number,cont=1;
convert_txt_to_number=converted_msg;
return convert_txt_to_number;
}
char convert_numbers_to_text(signed long long int converted_code)
{
char msg;
msg=converted_code;
return msg;
}

```

Listing F.3: EncProtocol 01.cpp

```

#include <columns>"columnsEncProtocol_01columns.columnsshcolumns"
using namespace std;
signed long long int euclidean_alg_EP_01(signed long long int a, signed long long int b)
{
    signed long long int x[99],y[99],quoc[99],rest_val[99],i=2,cont=0,temp_val=1,k;
    //step which is defining the initial statement variables
    quoc[0] =0;    quoc[1]=0;
    rest_val[0]=a;    rest_val[1]=b;    rest_val[2]=1;
    x[0]    =1;    x[1]=0;
    y[0]    =0;    y[1]=1;
    while(temp_val>0)
    {
        //From this point onward, the mathematical operations will start and alpha and beta values are
        ↪ counting
        rest_val[i] = rest_val[i-2] % rest_val[i-1];
        quoc[i] = rest_val[i-2] / rest_val[i-1];
        x[i] = x[i-2] - quoc[i] * x[i-1];
        y[i] = y[i-2]-quoc[i] * y[i-1];
        temp_val = rest_val[i];
        cont++;
        i++;
    }
    return(rest_val[cont]);
}
signed long long int power_calc_encl(signed long long int a,signed long long int e,signed long
    ↪ long int n)
{
    signed long long int A,P,E; //Declaration of Whole Variables
    A=a; P=1; E=e; //Variables receiving values
    while(E!=0){ //Tie for the algorithm's operations occur until the condition 1 (E == 0) is met
        if (E%2!=0){ //It checks if the Value of E is inparate according to the return of the rest
            P=(A*P)%n;
            E=(E-1)/2;
        }else{
            E=E/2;
        }
        A=(A*A)%n;
    }
    return(P);
}
signed long long int euclidian_alg_encl(signed long long int a, signed long long int b)
{
    signed long long int x[99],y[99],quoc[99],rest_val[99],i=2,cont=0,temp_val=1,k; //Initial
    ↪ Statement of Variables
    quoc[0] =0;    quoc[1]=0;
    rest_val[0]=a;    rest_val[1]=b;    rest_val[2]=1;
    x[0]    =1;    x[1]=0;
    y[0]    =0;    y[1]=1;
    while(temp_val>0)
    {
        //Start of the operation to find out the MDC and the values of alpha and beta.
        rest_val[i] = rest_val[i-2] % rest_val[i-1];
        quoc[i] = rest_val[i-2] / rest_val[i-1];
        x[i] = x[i-2] - quoc[i] * x[i-1];
        y[i] = y[i-2]-quoc[i] * y[i-1];
        temp_val = rest_val[i];
        cont++;
        i++;
    }
    return(y[cont]);
}

```

Listing F.4: EncProtocol 02.cpp

```

#include <iostream>
#include <fstream> //Call the functions required for the
#include <time.h>
#include <windows.h>
#include <math.h>
#include columns"columnsEncProtocol_02columns.columnshcolumns"
using namespace std;
void main_function_EP01(signed long long int *n,signed long long int *m,signed long long int *
    ↪ k)
{
signed long long int fator,q; //Declaration of the variables of the main function
cout<<columns"columnsGeneratingcolumns_columns2columns_columnsprimecolumns_columnsvalues
    ↪ columns_columnsforcolumns_columnssthecolumns_columnscalculationscolumns_columns_
    ↪ columnspleasecolumns_columnswaitcolumns_columns.columns_columns.columns_columns.
    ↪ columns"<<endl;
fflush(stdin);
*m=generator(100); //Calls the function that will generate the random values for m and
    ↪ k
*k=generator(10000); //Note that m will be > 1,000,000 and > 80,000,500
while(check_corr(*m)){ // LOOP function that will check if m can be divided
*m=generator(1000);
} //cousin <5000. Whenever this occurs, a new m will be generated.
while(check_corr(*k)){
*k=generator(34238);
} // Same previous function, but applied to k.
while(!testMillerRabinPrim(*m)){ // Running Miller's test for m, if the test points out that m
    ↪ is composed
*m=generator(1000);
} // a new value of m will be generated
while(!testMillerRabinPrim(*k)){ // Miller parameter k tested
*k=generator(33389);
}
cout<<columns"columnsRandomcolumns_columnsnumbercolumns_columnsgeneratedcolumns:columns"<<endl
    ↪ <<*m<<endl<<*k<<endl; //Message printing to the user
*n = (*m) * (*k); // Calculation of n as a function of m and k
cout<<columns"columnskeycolumns_columnsgeneratedcolumns_columnssuccessfullycolumns_columns_
    ↪ columnspubliccolumns_columnskeycolumns_columnsiscolumns:columns_columns"; //Message
    ↪ printing to the user
cout<<endl<<endl<<*n<<endl<<endl; //Screen printing of new lines and the value of n
}
signed long long int generator (long long int time_curr)
{ //defining the function that is related to the random number generation
signed long long int a;
srand(time(NULL)); //Sudo Random function responsible for generating a new random tree, so
    ↪ that
a=rand() %100000 + (time_curr); //assuring the numbers that are generated are always different
if (a%2!=0){ // Process to verify if the generated number is even or odd, if imp
return a; //the generated value will be returned to the main function
}else{ // otherwise, that even number will be sent - that is, the immediately preceding odd
return (a-1);}
}
bool check_corr(signed long long int x)
{ // function to verify whether the divider is less than 5000
//array which is having 669 primaries from 1 to 5000
int i, prime_values
    ↪ [669]={2,3,5,7,11,13,17,19,23,29,31,37,41,43,47,53,59,61,67,71,73,79,83,89,97,101,103,
107,109,113,127,131,137,139,149,151,157,163,167,173,179,181,191,193,197,199,211,223,227,229,
233,239,241,251,257,263,269,271,277,281,283,293,307,311,313,317,331,337,347,349,353,359,367,
373,379,383,389,397,401,409,419,421,431,433,439,443,449,457,461,463,467,479,487,491,499,503,
509,521,523,541,547,557,563,569,571,577,587,593,599,601,607,613,617,619,631,641,643,647,653,
659,661,673,677,683,691,701,709,719,727,733,739,743,751,757,761,769,773,787,797,809,811,821,
823,827,829,839,853,857,859,863,877,881,883,887,907,911,919,929,937,941,947,953,967,971,977,
983,991,997,1009,1013,1019,1021,1031,1033,1039,1049,1051,1061,1063,1069,1087,1091,1093,1097,
1103,1109,1117,1123,1129,1151,1153,1163,1171,1181,1187,1193,1201,1213,1217,1223,1229,1231,
1237,1249,1259,1277,1279,1283,1289,1291,1297,1301,1303,1307,1319,1321,1327,1361,1367,1373,
1381,1399,1409,1423,1427,1429,1433,1439,1447,1451,1453,1459,1471,1481,1483,1487,1489,1493,
1499,1511,1523,1531,1543,1549,1553,1559,1567,1571,1579,1583,1597,1601,1607,1609,1613,1619,
1621,1627,1637,1657,1663,1667,1669,1693,1697,1699,1709,1721,1723,1733,1741,1747,1753,1759,
1777,1783,1787,1789,1801,1811,1823,1831,1847,1861,1867,1871,1873,1877,1879,1889,1901,1907,
1913,1931,1933,1949,1951,1973,1979,1987,1993,1997,1999,2003,2011,2017,2027,2029,2039,2053,
2063,2069,2081,2083,2087,2089,2099,2111,2113,2129,2131,2137,2141,2143,2153,2161,2179,2203,
2207,2213,2221,2237,2239,2243,2251,2267,2269,2273,2281,2287,2293,2297,2309,2311,2333,2339,
2341,2347,2351,2357,2371,2377,2381,2383,2389,2393,2399,2411,2417,2423,2437,2441,2447,2459,
2467,2473,2477,2503,2521,2531,2539,2543,2549,2551,2557,2579,2591,2593,2609,2617,2621,2633,
2647,2657,2659,2663,2671,2677,2683,2687,2689,2693,2699,2707,2711,2713,2719,2729,2731,2741,

```



```

2749,2753,2767,2777,2789,2791,2797,2801,2803,2819,2833,2837,2843,2851,2857,2861,2879,2887,
2897,2903,2909,2917,2927,2939,2953,2957,2963,2969,2971,2999,3001,3011,3019,3023,3037,3041,
3049,3061,3067,3079,3083,3089,3109,3119,3121,3137,3163,3167,3169,3181,3187,3191,3203,3209,
3217,3221,3229,3251,3253,3257,3259,3271,3299,3301,3307,3313,3319,3323,3329,3331,3343,3347,
3359,3361,3371,3373,3389,3391,3407,3413,3433,3449,3457,3461,3463,3467,3469,3491,3499,3511,
3517,3527,3529,3533,3539,3541,3547,3557,3559,3571,3581,3583,3593,3607,3613,3617,3623,3631,
3637,3643,3659,3671,3673,3677,3691,3697,3701,3709,3719,3727,3733,3739,3761,3767,3769,3779,
3793,3797,3803,3821,3823,3833,3847,3851,3853,3863,3877,3881,3889,3907,3911,3917,3919,3923,
3929,3931,3943,3947,3967,3989,4001,4003,4007,4013,4019,4021,4027,4049,4051,4057,4073,4079,
4091,4093,4099,4111,4127,4129,4133,4139,4153,4157,4159,4177,4201,4211,4217,4219,4229,4231,
4241,4243,4253,4259,4261,4271,4273,4283,4289,4297,4327,4337,4339,4349,4357,4363,4373,4391,
4397,4409,4421,4423,4441,4447,4451,4457,4463,4481,4483,4493,4507,4513,4517,4519,4523,4547,
4549,4561,4567,4583,4591,4597,4603,4621,4637,4639,4643,4649,4651,4657,4663,4673,4679,4691,
4703,4721,4723,4729,4733,4751,4759,4783,4787,4789,4793,4799,4801,4813,4817,4831,4861,4871,
4877,4889,4903,4909,4919,4931,4933,4937,4943,4951,4957,4967,4969,4973,4987,4993,4999
};
for (i=0;i<=699;i++){
return ((x%prime_values[i]) == 0); //Checks whether the rest is 0. If it is, it will return
    ↪ True, otherwise FALSE
}
}
bool testMillerRabinPrim(signed long long int n)
{ // Function to apply Miller test
signed long long int s,d,b,e,x,base[10]={2,3,5,7,11,13,17,19,23,29}; //defining the first 10
    ↪ primary numbers
int i=0;
while(i<=9){//LOOP Required to test the 10 bases
for(s = 0, d = n - 1; !(d & 1); s++)//Factor n-1 in 2 ^ s, with increasing s from 0
d >>= 1;
for(x = 1, b = base[i] % n, e = d; e >>= 1)// base [i] ^ d = x (mod n) using exposition
{
if(e & 1) x = (x * b) % n;
b = (b * b) % n;
}
// Process to check whether base [i] ^ (d 2 ^ [0 ... s-1]) mod n! = 1
if(x == 1 || x == n-1) return true;//Process to check whether base [i] ^ (d 2 ^ [0 ... s-1])
    ↪ mod n! = 1
while(s-- > 1){ //will return true
x = (x * x) % n;
if(x == 1) return false; //Returns False if the number has been shown Compound
if(x == n-1) return true;
}
return false;
i++;
}
}
signed long long int fermat (signed long long int n)
{ //This is the function for calculate the Fermat theorem
signed long long int i=n, j=0, k=0; //Declaration of the variables used in the function
do{
i += j;
k = (int) sqrt((double)i);
j += ((!j)?1:2);
}while (i-k*k>0);
k += (j-1)>>1;
n /= k;
return (n>k?k:n);
}
void factorization (signed long long int n)
{ //Function to make the simple factoring for the equation
int b,cont=0;
b=n-2; //Since we are dealing with cousins, we only need to make the divisions by odd numbers
while(cont<2){ // Given that the only prime pair is the number 2, out of our range
if (n%b==0){
cout<<b<<"columns_columns_columns_columns"; //Print the value of b if it is a divisor
    ↪ of n
cont++; //The Counter can only reach the limit of 2, because this is the maximum number of
    ↪ dividers
}
//which it can not have, obviously excluded its own No. 1
b=b-2;// b It will always be reduced by 2, that is, it will always be odd.
}
cout<<endl<<endl; //Changing the Print Line on the Screen
}
}

```