View metadata, citation and similar papers at core.ac.uk
brought to you by CORE
provided by University of Missouri: MOspace

MANAGING MALICIOUS TRANSACTIONS IN
MOBILE DATABASE SYSTEMS

A THESIS IN

Computer Science

Presented to the Faculty of the University
of Missouri-Kansas City in partial fulfillment of
the requirements for the degree

MASTER OF SCIENCE

by
ABHIRUCHI SINGH

B.E., NATIONAL INSTITUTE OF TECHNOLOGY, 2005

Kansas City, Missouri
2013

MANAGING MALICIOUS TRANSACTIONS IN

MOBILE DATABASE SYSTEMS

Abhiruchi Singh, Candidate for the Master of Science degree

University of Missouri- Kansas City, 2013

ABSTRACT

Database security is one of the most important issues for any organization, especially for financial institutions such as banks. Protecting database from external threats is relatively easier and a number of effective security schemes are available to organizations. Unfortunately, this is not so in the case of threats from *insiders*. Existing security schemes for such threats are some variation of external schemes that are not able to provide desirable security level. As a result, still authorized users (insiders) manage to misuse their privileges for fulfilling their malicious intent. It is a fact that most external security breaches succeed mainly with the help of insiders. An example for an insider is the Enron scandal of 2001 which led to bankruptcy of Enron Corporation. The firm was widely regarded as one of the most innovative, fastest growing and best managed business in the United States. When Enron filed for bankruptcy its share prices fall from US$90 to $1 causing a loss of nearly $11 billion dollar to its stakeholders. Financial officers and executives misled outside investors, auditors and Enron’s board of directors about corporation’s net income and liabilities. These insiders kept reported income and reported cash flow up, asset value inflated and liabilities off the book to meet Wall Street expectations. Enron's $63.4 billion in assets made it the largest corporate bankruptcy in American history at that time.

Existing security policies are inadequate to prevent the attacks from insiders. Current database protections mechanisms do not fully protect occurrence of these malicious transactions. These requires human intervention in some form or other to detect malicious transactions. In a database, a transaction can affect the execution of the subsequesnt transactions thereby spreading the damage and hence making the attack recovery more complex. The problem of malicious attack becomes more pronounced when we are dealing with mobile database systems.

This thesis proposes a solution to mitigate insider attack by identifying such malicious transactions. It develops a formal framework for characterizing mobile transaction by identifying essential components like order of data access, order of operations and user profile.

APPROVAL PAGE

The faculty listed below, appointed by the Dean of School of Computing and Engineering, have examined a thesis titled “Managing Malicious Transactions in Mobile Database Systems,” presented by Abhiruchi Singh, candidate for the Master of Science degree, and certify that in their opinion it is worthy of acceptance.

Supervisory Committee

Vijay Kumar, Ph.D., Committee Chair
Department of Computer Science and Electrical Engineering

Praveen Rao, Ph.D.
Department of Computer Science and Electrical Engineering

Ghulam Chaudhry, Ph.D.
Department of Computer Science and Electrical Engineering

CONTENTS

ILLUSTRATIONS

.

LIST OF ABBREVIATIONS

DBMS	Database Management System
MDS	Mobile Database Systems
MT	Malicious Transaction
BS	Base Station
AP	Access Point
MU	Mobile Unit

ACKNOWLEDGEMENTS

I would like to express my most sincere gratitude to my advisor Dr. Vijay Kumar for his support of my thesis, motivation and knowledge. Without Dr. Kumar, this dissertation would not have been possible. I sincerely want to thank him for his constant guidance and invaluable assistance during my thesis and sparing time for my work whenever required. He supported me when I was exploring various possibilities and topics and later guided me at each and every step.

I would like to thank Dr. Praveen Rao for his support in my thesis. Without his vast knowledge and support this thesis would not have been successful.

I also owe my deepest gratitude to Dr. Ghulam Chaudhry for sparing his valuable time. It has been a privilege and a very good learning experience to work under all my committee members.

I would like to thank my husband, family members and friends without whose support and encouragement I would not have been able to pursue my Masters, and completed it successfully.

CHAPTER 1

INTRODUCTION

1.1 Database Security and Attacks

All organizations (academic, business, etc.) use some database system to manage their information processing needs. Database management system is a key component in handling information infrastructure and forms the ultimate layer of data access. Organizations like banks have special data processing needs and they demand strict security and privacy in data management. A major difficulty faced by such organizations is the protection of data against corruption and attack. Database security is conventionally defined as the protection of database from unauthorized users also known as outsiders but security can also be threatened by Insiders i.e. authorized person accessing the database with malicious intent. Recent database systems are more or less well protected from outsider attacks. This is not the case for insider attack. A malicious intent can be authenticated misuse or inadvertent mistakes made by authorized individuals or processes.

Insiders and Outsiders with respect to a database management system are defined as follows:

Outsiders: Unauthorized users that may gain access to the database by exploring system vulnerabilities and can then execute unauthorized transactions.

Insiders: Authorized users that can execute malicious transactions which cannot be stopped by typical security mechanisms.

1.2 Database Security Mechanisms

The main goal of database management system dealing with security from outsider attack is to protect data stored in database from unauthorized access. These attacks take place when:

- Security measures are not properly activated and configured which allows intruders to get access to the database.
- Existing flaws in the database which can be discovered by the outsiders.
- Outsiders can get the credentials of authorized users for fulfilling their intent.

Several mechanisms have been developed to prevent the outsider attack. Following are some examples:

- Access Control and privileges: This includes provisions for restricting access to the database and is handled by creating user accounts and passwords to control login process by the DBMS. This way some users may be permitted only to retrieve data where as others are permitted to update data as well. A discretionary access control in DBMS is enforced based on granting and revoking privileges on an authorization level.
- Data Encryption: This protects sensitive data that is being transmitted via some communication network by encoding the data using encoding algorithm. This

way an unauthorized user will not be able to decipher the information but authorized user will be supplied with decoding algorithm so that they can decipher the data.

- Audit: Database system keeps a log of all the updates performed on the database in a system log. If any tampering with the database is suspected, a database audit is performed which consists of reviewing the log to examine all access and operations applied to the database.

These security measures against outsiders are mainly designed based on the vulnerabilities of the system which is related to the set of security mechanisms available, the correct configuration and the hidden flaws of the system implementation. But these typical database security mechanisms are not able to detect and handle malicious attack from authorized users which takes advantage of their privileges to maliciously access or destroy data. These malicious attacks mainly depend on the intentionality and capability of insider breaching the system. Even a truly secure system is vulnerable to abuse by insiders who abuse their privileges.

The objective of data security is approached in two distinct ways: Prevention and Detection, which means preventing the database from any security breach and in case a security breach do happen then taking necessary steps to detect that intrusion. The mechanisms incorporated for outsider attack are primarily designed for preventing intrusion but no measures are incorporated for the detection of any intrusion in the database. Since insider take place from inside hence prevention mechanisms do not hold valid for such attacks.

A system faces more danger from insider attack and also it is more difficult to protect information from insider attacks than outsiders. Statistically roughly a third of the computer security losses are because of insiders and also these attacks are more damaging than the outsider attack.

1.3 An Insider

The term insider is by no means exhaustive and it can be a current or former employee or it can be a person masquerading as a legitimate insider or someone to whom an insider has given access.

A malicious insider can be categorized into two classes; traitor and masquerades.

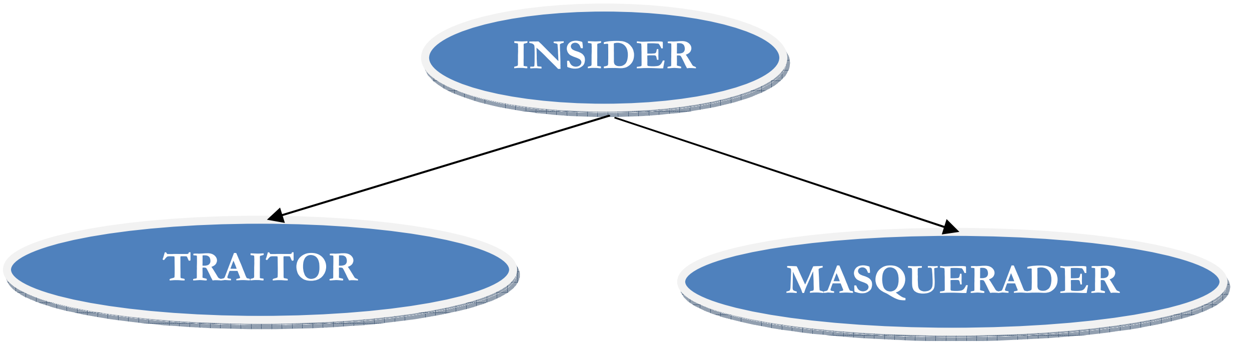


Figure 1. Types of Insider

- Traitor: A legitimate user within an organization who has been granted access to information resources, but whose actions are counter to policy and, and whose goal is to negatively affect confidentiality, integrity, or availability of information asset. The traitor uses his/her legitimate credentials when perpetrating their malicious actions.
- Masquerader: An attacker who succeeds in stealing a legitimate user's identity and impersonates another user for malicious purposes.

A traitor and a masquerader can be distinguished based upon the following two features:

- Amount of system knowledge each has: A traitor has full knowledge of the systems as they use it routinely and they are also aware of all the security policies in place. A masquerader on the other hand may have far less knowledge than the traitor.
- Intent: An insider attack may be due to an innocent mistake by a legitimate user but a masquerader always knows about his malicious intent.

1.4 Insider Malicious Intent and its Effect

Insiders can pose a significant risk to an organization because of the legitimate access privilege they have. An insider can cause security breach because of a number of reasons ranging from an unintentional mistake to getting some financial advantage. These actions can be grouped as:

- Unintentional intent: Testing system limits and vulnerabilities as a form of innovation, challenge or for killing boredom.
- Greed: Employees with financial problem or just out of greed for earning easy money use their access privilege to transfer money from a customer’s account to their accounts.
- Disgruntlement: Technical employees can use their technical knowledge or skills to pose a significant risk to network security by acting out of revenge for negative work related events like transfer, demotion or dispute with employer. Disgruntlement can also be a result of certain unmet expectations like timely or insufficient promotion or raise or diminished responsibilities or due to limitations on use of company resources.
- Predisposition: A number of employees exhibit malicious behavior owing to certain predispositions ranging from inability to conform to rules to serious mental disorders.

The impact of these malicious attacks no matter whether originating from a Traitor or a Masquerader and regardless of the intent of the insider can be identified as:

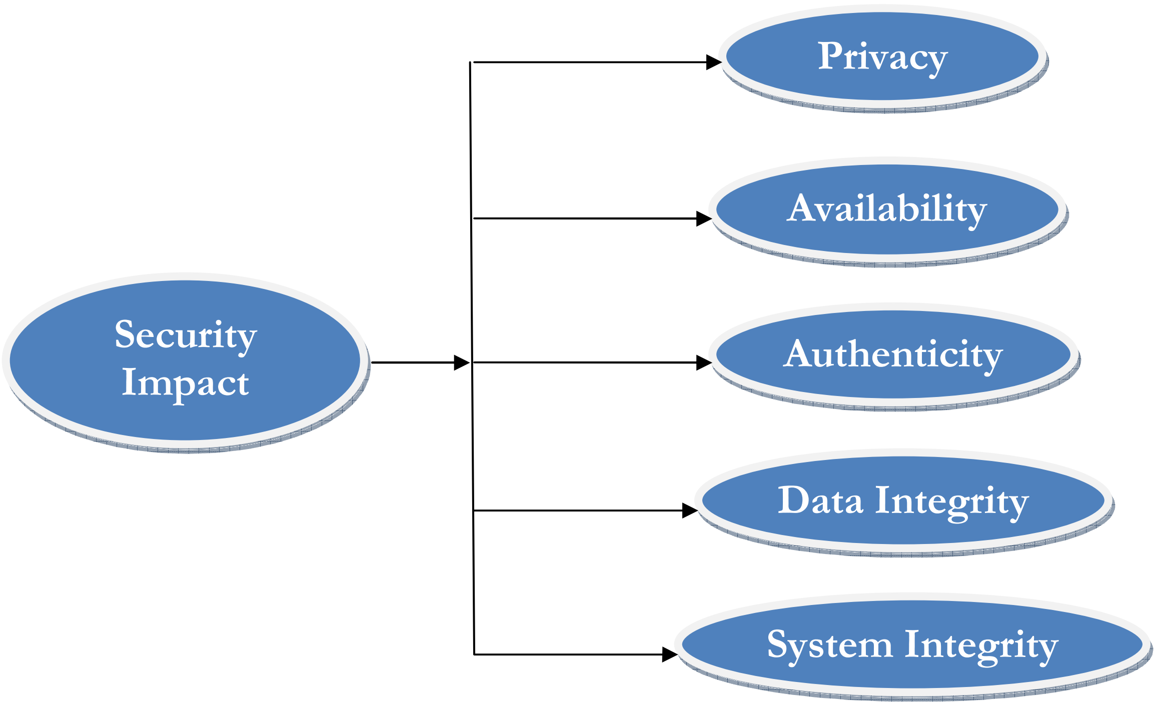


Figure 2. Security Impact

- ➢ Privacy Issue: In this case, individual’s data is divulged inappropriately and sensitive data is exposed.
- ➢ Authenticity Issue: This guarantees that a service or information is authentic. A malicious user can destroy or corrupt private data thereby destroying the authenticity of the data.

- Data Integrity Issue: Data Integrity is maintained by protection of a service or information against illicit and/or undetected modification. When an insider add/modify/delete data inappropriately data integrity is lost and data becomes corrupted. An example would be if a bank officer maliciously performs some credit activity to show that money has been added to an account without actually crediting the money.
- System Integrity Issues: System Integrity would be hampered when an insider with high security privileges grants privileges to user with lower privilege level and providing them access and disclosure of sensitive data.
- Availability Issue: Protection of service or information against possible denial of service. Interferences from a malicious user can cause undue delays in accessing or using data or even denial of service.

The threat analysis of the impact of a security attack revels that the greater the harm a threat poses the higher the security a system should poses against that attack. An insider causing harm with accidental blundering could be bad but an insider with malicious intent is even more dangerous because he means to cause harm and he can select the areas where the impact would be considerable. Moreover he can even work towards concealing his actions.

Major updates to a client's bank by a malicious insider can cause havoc to data integrity but these transactions are also likely to get caught but subtle data alterations like few transactions of very small amount from a rich account can pass unnoticed. For this reason threats caused by malicious insiders are of more risk than an external security attack.

1.5 Real World Examples

The effect of malicious transactions from insiders can be better understood with the following facts:

- Up to 80% of system breaches are caused by internal users, including privileged administrators and power users, who accidentally or deliberately damage IT systems or release confidential data assets, according to a recent Cyber-Ark survey.
- Average cost per computer security incident of financial fraud is close to $500,000
- The famous incident of wiki-leaks is attributed to be an insider job.
- A Flextronics employee was charged of insider trading. The executive was paid high sums for passing on information pertaining to iPhone 4 development plans.
- An Ofcom IT director was charged of defrauding the organization by creating false invoices sent to him.
- A Netflix call-center employee stole credit card numbers of customer he had spoken with.

The above mentioned points supports the fact that insiders threats are on the rise and poses more risk compared to an outsider but they are not getting anywhere the same attention. The reason behind this imbalance can be that insider threats represent a hard problem to quantify, especially since the biggest internal threat is often the employee itself.

1.6 Introduction to Chapter 2

Many businesses applications are now going mobile and hence use of Mobile Database Management System is on rise and with this increasing popularity of mobile applications there has to be a further increase in securing the data in this mobile platform. Mobile devices operating on wireless networks do not provide a secure storage environment for the protection of data; therefore layers of security must be implemented at the database level for the better protection of data. This problem becomes more pronounced when we are dealing with insider attack on mobile database systems.

CHAPTER 2

MOBILE DATABASE SYSTEMS

2.1 Architecture of Mobile Database Systems

Mobile Database system is a database management system connected to a mobile device which can be a laptop, PDA, mobile or any small device that can be connected to the wireless network. This works as a Client-Server model where mobile devices as client have light databases loaded up on them to exchange data on the fly without worrying about time and distance. Information can be later synchronized with the server database.

Mobile computing network typically consists of fixed hosts, Mobile Units (MU) and base stations. Base Stations (BS) or Access Points (AP) are equipped with wireless interfaces and communicate with mobile units to support data access over the wireless network. The BS serves as transmitter/receiver device connecting the wired network from a fixed location and receives buffers and transmits data between wireless network and wired network infrastructure. A base station covers a geographical area called cell. A MU can directly communicate with one BS which covers the area where the mobile a moves. MU's are equipped with antenna for capturing the signal, transceiver for receiving and sending signals and an interface for interacting with the user.

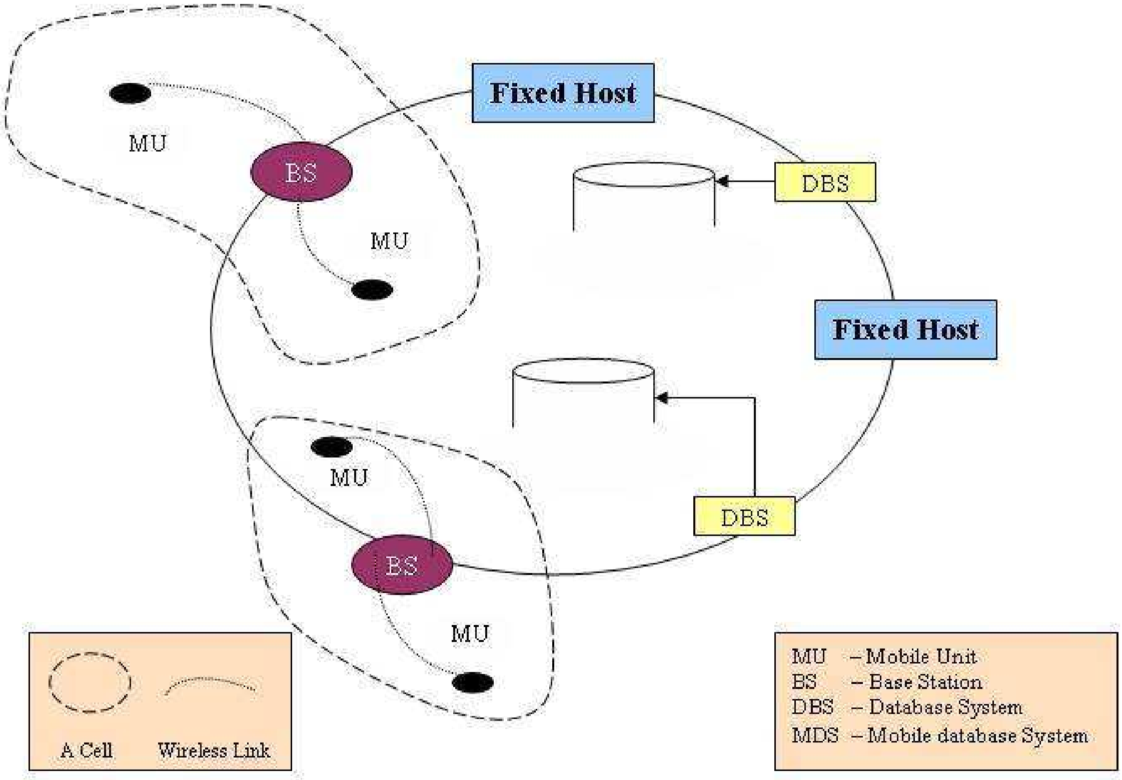


Figure 3. Mobile System Architecture

2.2 Vulnerabilities of Mobile Wireless Networks

The wireless medium used by mobile devices is 802.11which specifies an over the air interface between a wireless client and a BS. A wireless technology however induces certain more issues like:

- Disconnections: Due to issues of noise and interference, wireless medium is prone to frequent disconnections.

- Bandwidth: Limited capacity of the device together with interference and noise and the number of mobile users on an AS at one time can lead to variability of bandwidth available to transmit data.
- Security risks: Wireless network are more prone to security attacks as opposed to a wired medium.

Compared with wired network, mobile networks are much unreliable. Disconnection between a mobile device and network can be a frequent issue. The communication delay is also unpredictable and takes time. The data access in mobile environment is characterized by dominant issues of communication and mobility. Communication over wireless networks is prone to disconnections, low bandwidth and noise. Mobility causes data change at a frequent rate thereby making the data volatile. Wireless users typically pass through a number of cells while executing a transaction through their mobile device. This process of leaving a cell and entering a new cell is called hand-off. In order to have a smooth hand-off cells usually overlap, this process usually complicates the database management at a significant level.

It is much more difficult to execute transactions in a mobile environment owing to the fact of potential risk of network disconnection. Network disconnection not only affects the processing of the disconnected transaction but it can also affect the other transactions which want to access the data items locked by the disconnected transaction. Owing to the limitations of a mobile database techniques different from those developed for conventional distributed database systems are required.

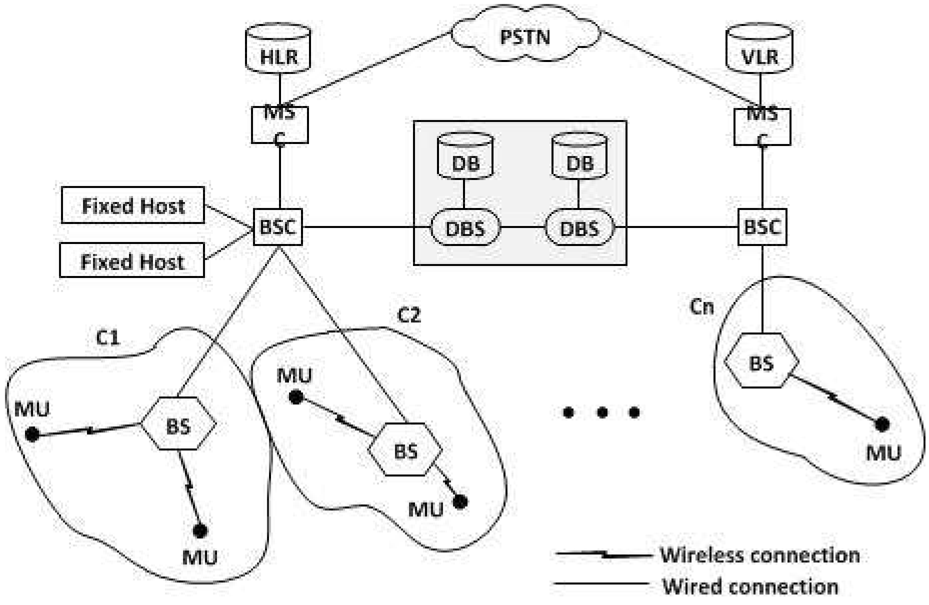


Figure 4. A Detailed Diagram showing connection of BS with BSC and database

2.3 Security Issues in Mobile Database Systems

With the growing popularity of smartphones there's a rise in business application going mobile and available to user anytime and anywhere. Initially mobile applications were used as electronic planners with calendars and schedulers but with the advent of wireless technologies user are now able to perform real-time data processing and transmission. User can even work on their financial data from any location and any time; this is one of the major achievements of including mobility in database. But with this ease of data access comes security of critical data. Mobile devices are originally designed to make the device as usable as possible, with little thought to security. Moreover the dubious record of wireless technology towards security calls for a granular and multilevel security setup. Security arrangements are hence needed on the data level itself. Security requirements which were once limited only to banks central database has to be now replicated on mobile devices with added security measures. The existing database management capabilities have to be adapted in the mobile context plus the new ones that attack issues pertaining to mobile environment of database systems. Host-based detection should be an integral part of an overall intrusion defense.

Database security is the process and procedure of protecting the database from unintended activities. This unintended activity also includes an authenticated misuse, malicious attacks or inadvertent mistakes made by authorized personnel. Traditional database systems are protected from external attacks by using firewalls which creates a demilitarized zone between internal and external network. On top of these existing security measures we need additional layer of security to detect and recover from malicious attacks i.e. malicious action detection protocol which is a host-based intrusion detection system should be built on top of network intrusion detection system which includes access control, auditing, authentication, encryption and integrity control.

But designing this layer in trying to secure the mobile data access is much harder in mobile database systems for the following reasons:

- Location dependent data: LDD’s value is coupled with location [1, 2, 3, 15] and its consistency is dependent on data location, origin of the transaction and time.

Example: Query from a moving vehicle to know the distance from a location say 'A' changes with location. Multiple base stations may need to be contacted for getting the correct version of the result this can lead to a long transaction.

- Mutual Incompatibility in transaction execution: A constraint or condition can be true at one location but can be false at some other location. Since in MDS nodes are mobile this can lead to mutual incompatibility. For example: constraint of using seat belt for the passengers in back seat of a car is mandatory in California but not in Missouri, hence a good transaction in Missouri is malicious in California.
- Management of transaction logs: MU can log by following any of the following approaches:
    - Logging at the processing nodes i.e. logging at MU itself.
    - Logging at centralized location i.e. at a designated DBS.
    - Logging at the place of registration i.e. BS
    - Saving logs on external drives like a zip drive.

For a MT discovered just before commit, entire transaction has to be roll-backed (as opposed to partial roll-back in case of failure of a good transaction) which is resource intensive and effects systems performance.

- Transaction Commitment: In MDS a transaction may be fragmented and may run at more than one nodes. For an MT, a compensation might not always be possible because of the non-availibility of node. Also, a server cannot monitor the disconnected processing in a node hence a local prevention scheme must be present in every MU.

- Database Recovery: Database recovery is much harder in MDS because of the mobility of processing nodes, limited wireless channels availability and disconnected processing capability. Recovery requires unique logging and checkpoint demands which if not managed correctly can even make the roll-back of MT's impossible.

2.4 Transaction Processing in Mobile Database Systems

A conventional database system uses the concept of an executing program called transaction that forms a logical unit of database processing. A transaction basically includes database access operations like reading a data item (reading a database item into a program variable) and writing a data item (writing the value of a program variable into database item).

A transaction must possess ACID properties which are enforced by the concurrency and recovery methods of the database management system. The ACID properties are:

- Atomicity: Transaction is either fully completed or not performed at all. Recovery technique of a DBMS takes care of undoing any effects caused by a failed transaction. This property ensures that there are no intermediate results.

- Consistency: A transaction takes the database from one consistent state to another i.e. if a database is consistent before the execution of a transaction it should be in consistent state after the completion of the transaction.
- Isolation: A transaction should be executed in isolation from another transaction executed concurrently. This is guaranteed by the concurrency control mechanism of the DBMS.
- Durability: Any final result in the database should persist and should not be lost due to any failure.

The above mentioned four properties forms a part of a flat transaction which performs one level operation and does not influence any other dependent transaction. But the ACID properties in their raw form are not enough to satisfy MDS requirements. For instance,

- Atomicity presents a problem because of the intermittent disconnections in mobile environment which leads to interruption of mobile transactions.
- Consistency presents a problem because mobile transactions before execution may have to refresh data that's is out-of-date due to local caching.
- Isolation presents a problem because ensuring data availability while trying to limit the traffic between server and mobile device can result in multiple users looking at the same data.
- Durability presents a problem because mobile transactions are long lived and can be error-prone.

Flat transaction model used in conventional database systems is not scalable for meeting needs of mobility. Growing requirements of mobility and management of continuous exchange and processing of real-time data requires that the concept of consistency and isolation should be extended. Also, in conventional database systems the processing units are not mobile which means the processing should take place at a fixed location. Even in a distributed system where data moves from one location to another, information is actually exchanged from one fixed node to another which does not necessarily means mobility.

The processing units in case of a mobile database system are mobile as opposed to stationary processing units of conventional database systems but at the same time the ACID properties of a transaction carried out in mobile database system should be maintained. Mobile database system adds concept of location based data access to the conventional database systems which makes malicious action detection quite hard [4] to catch. As a mobile host moves from a cell to another cell, its transactions migrate from one BS to another. We can say that transactions in mobile database require relaxed ACID properties.

As per Pitoura, E. and B. Bhargava [5] in Building information systems for mobile environments, "Mobile transactions are long-running, error-prone and heterogeneous. As a result, modeling mobile transactions as ACID transactions is very restrictive. ACID transactions have limited expressive power and offer no way of modeling computations with a complex control structure. Furthermore, ACID transactions do not support partial commitment or abortion of a transaction, or partial recovery. Finally, there is no way of “suspending” a transaction to survive a disconnection.

2.5 Introduction to Chapter 3

Since conventional database management systems are not suitable for mobile environment we need an alternate transaction model that can handle mobile environment requirements. A number of research works has been done on this topic and a number of solutions have been proposed for the same. But when we include the issue of insider in a Mobile Database System we note that very little or no work has been done on the issue. Database system currently is unable to detect a MT without any human intervention. It is believed that the issue of detection of an insider is purely based on the intent of the insider and is hence impossible to detect without any human help. So majority of the work done on the issue of Malicious Transaction is limited to controlling the damage caused by a malicious transaction rather than identifying it in its earlier phases.

CHAPTER 3

RESEARCH PROBLEM

3.1 Problem Specification

Mobile database use within banking and financial industry is increasing, at a time when pressure for data security is also increasing. Since the security provided by the device and the network itself is not proper and can be compromised this calls for a granular and multilevel approach of providing security on the data level itself. Research into secure access models of mobile database is emerging. There are number of mechanisms and design available in the market that secure the mobile database from the attack of an insider but no work has been yet performed to safeguard a mobile database from the malicious actions of an insider.

An insider has all the proper credentials and privileges to access the database and execute transactions. These actions can be malicious depending on the intent of the insider. Even when the transactions are malicious they are logically correct and hence system does not have any means for flagging or terminating these actions. When a malicious transaction is committed only then it can be identified as malicious and this process still requires human intervention. Human check required for detecting malicious actions on the data is resource extensive and hence very expensive.

3.2 Review of Past Work

In database security most of the mechanisms proposed are meant to place a barrier between the intruder and the data and in spite of the pertinence of the detection of the malicious database transactions no practical mechanism able to identify malicious transactions without involving manual interventions exists.

Also, to the best of our knowledge most of the work has been done assuming that MT's are already present and then concentrating on the removal of their effects from database [6, 7, 8, 9, 10, 11, 12, 13]. This means that all the work done on MT's is for damage control and no formal work have been done on actually identifying a MT before its effect is committed on the database.

Moreover, all the work on the MT's has been majorly done on conventional database systems and there's not been a single formal research on Mobile database system that handles the issue of insider in mobile environment.

Following are some of the earlier work done by researchers on MT issue.

- Scheme of Panda and Giordano [10]: This scheme provides two techniques of detection and recovery. Both the techniques take care of detection and recovery of the transactions. However the first technique performs them simultaneously and blocks new transactions until recovery is complete. The second technique takes care of these two issues but is very process-intensive.

- Scheme of Panda and Haque [11]: This scheme assesses damage by following a data dependency algorithm. In this approach only the affected operations are undone and redone. However this is makes the entire scheme a very time consuming process.

- Scheme of P.Ammann and Others [7]: This scheme uses a color technique for marking the severity of damage and heavily depends on exploiting the system log to find out the pattern of damage spreading and schedule repair transactions. It's a backward recovery process in which the system waits for the moment when all affected transactions are found and then undoes the affected transactions backwards from the last one. In this approach, for each write log entry, a corresponding inverse operation must be constructed, which brings performance degradation.

- Scheme of P.Ammann, S.Jajodia and P.Liu [6]: This scheme employs a two pass static repair algorithm. This algorithm is based on the concept of affected by relationships among committed transactions. A committed transaction T_i is said to be affected by another committed transaction T_j, if T_i reads one or more data items last modified by T_j. The affected by relationship is transitive, i.e., if T_i is affected by T_j and T_j is affected by $T_{k,}$ then T_i is affected by $T_{k.}$ However this algorithm cannot be applied to workflow systems having control flow and data flow dependencies.

- Scheme of Lala and Panda [14]: This is a damage assessment algorithm which locates damage data items based on read-write dependencies between the transactions. It uses an effective method of storing dependency graphs to decrease log access time. But this scheme does not identify MT's and might not work on MDS because of the way the log is processed.

All the above schemes only focus on the damage control and no attention is given to the identification of a MT. Some schemes imply that it is not possible to identify a Malicious Transaction and others just pronounce detection of malicious transaction as unnecessary.

3.3 Justification for Research

The rapid proliferation of wireless networks and mobile computing applications has changed the landscape of network security. This popularity demands for an even tougher design for security of the MDS. Thus we need to include malicious detection in the security architecture for mobile computing environment. We need to search for new architecture and mechanisms to protect wireless networks and mobile computing application.

It is very important that the security mechanisms of a system are designed so as to prevent both unauthorized access as well authorized but malicious access to system resources and data. However, completely preventing breaches of security appear, at present, unrealistic. We can, however, try to detect these malicious intrusion attempts so that action taken to repair the damage will be much more simplified.

If there are insider attacks on a system, we would like to detect them as soon as possible (preferably in real-time) and take appropriate action. This is essentially what this research is about. An attack from an Insider is seemingly impossible to unveil because the user has access to the system, his or her actions would not be flagged. Our approach is to detect the malicious transactions through examining history and user profile and then taking preventive measures when an attack is detected; it is a pro-active approach rather than the existing reactive approaches. It plays the role of a police officer rather than just an informant.

3.4 Thesis Approach

It is very important that the security mechanisms of a system are designed so as to prevent unauthorized access as well as malicious authorized attack to system resources and data. Completely preventing breaches of security appear, at present, unrealistic. We can, however, try to detect these intrusion attempts so that action may be taken to repair the damage.

In this thesis we propose that malicious intentions though seemingly outside of the execution profile of a transaction can be deduced by analyzing the execution history of transactions and user profile to identify malicious intention. This thesis proposes identification of the malicious transactions eventually leading to damage control. In this approach more emphasis is given to the identification of the malicious transaction which makes damage control less frequent and hence less costly.

The concept behind malicious detection schemes is that there are ways to represent attacks in the form of a pattern or a signature so that even variations of the same attack can be detected. Malicious transaction detection systems detect the malicious actions based on a signature that encompasses all possible variations of the pertinent attack. This signature is actually a set of constraints or rules integrated on top of the actual database model and act as an integral component of the database model. The core of this malicious detection system lies on the set of selected constraints which flags the malicious activities and passes the non-malicious activity.

CHAPTER 4

SOLUTION AND SCHEME

4.1 Identification of MT's

We propose that one of the most effective and reliable way of detecting the presence of MTs is by examining the following two aspects of a transaction.

- Execution history: Malicious actions detected by monitoring for specific patterns of activity.
- Profile of the user: Malicious actions detected by atypical behavior profiles, violations of security constraints, or use of special privileges.

Initially these transaction aspects were manually examined hence this was not a very cost effective solution. Our research approach follows the same line of examining the execution history and deducing the profile of the user but in a manner which is both cost and time efficient. Also, the process of identification and logging is an integral part of the transaction model rather than enforcing it as a middleware incorporated on the application level.

The framework of the proposed model will comprise a set of parameters and the interrelationship between the parameters. For the framework development we propose the following parameters:

- Location: This parameter represents the location from where user initiates the transaction. This location can be the starting point of malicious transaction.

- User Profile: It represents the profile of the user who initiated the transaction.
- Transaction Type: The type of the transaction under scrutiny for detection can be a read or a write transaction. Both these transaction types are mutually exclusive and require a unique identification approach.
- Data Access Mode: For a financial transaction a data can be accessed through a number of modes like debit, credit, payment, fund transfer etc.

The interrelationship among parameters is also an integral part in detecting and deciding the malicious intent of the insider. For example, a user performing major debt activities from outside his regular home location can flag concerns. Similarly, a bank official performing operations on a customer data after his office hours can also trigger a warning.

In this research we adopt the notion of employing constraints for identifying the MTs. We propose that a transaction is malicious if it violates a subset of predefined constraints. These constraints are based on a three vertex approach where location, user profile and data item are the three vertex of the proposed model. This model though targeted for MDS can also be used for conventional database systems with the exclusion of location involving constraints.

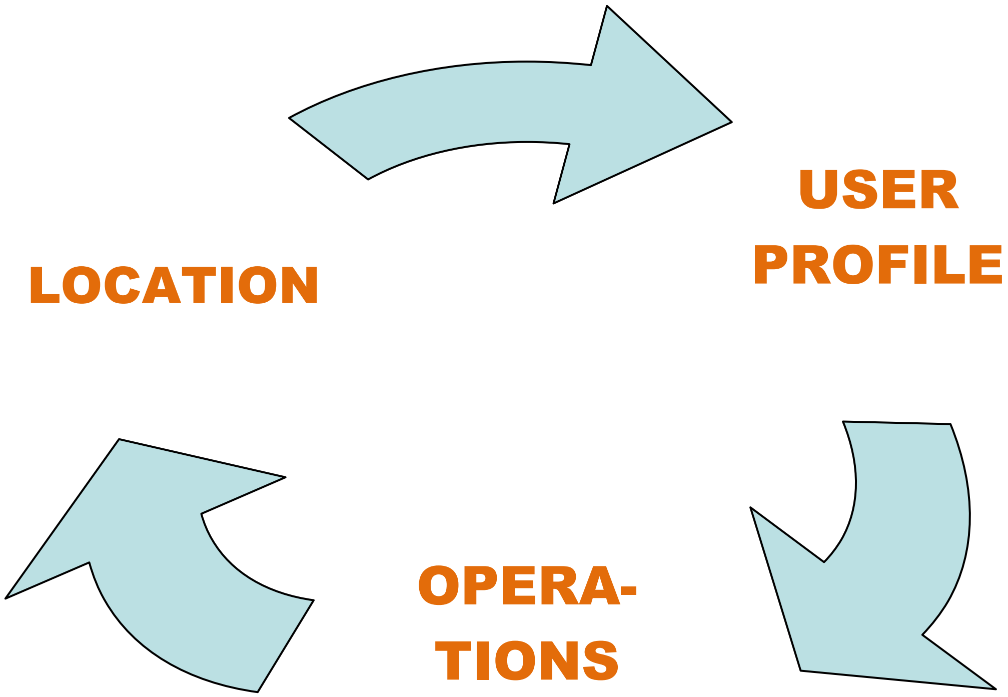


Figure 5. Constraint Model

The violation on any one or more of the vertex of the constraint model represents a constraint violation. For example, if a data item is accessed by a user from a location which is outside from the defined location then we say that a constraint violation has occurred. The violation of ‘k’ such constraints can result in the flagging of a malicious transaction.

In this research we also propose the execution dependency of the transactions. A transaction can either be user initiated or it can be forced dependent on other transaction. Since transaction are executed in a concurrent manner, the effect of MTs spread through transaction dependency. For example consider a user initiated malicious transaction $T_{malicious}$ and also consider a clean transaction T_{clean} which is dependent on $T_{malicious}$. T_{clean} can ultimately become “Forced malicious”. Our research takes care of both the situation of user initiated malicious transaction and forced malicious transactions. We create constraints that also take care of the forced malicious transactions.

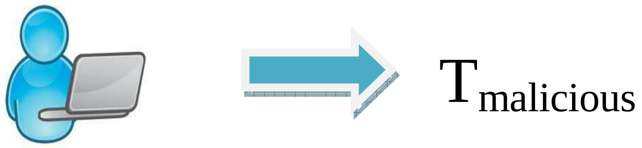


Figure 5. User initiated Malicious Transaction

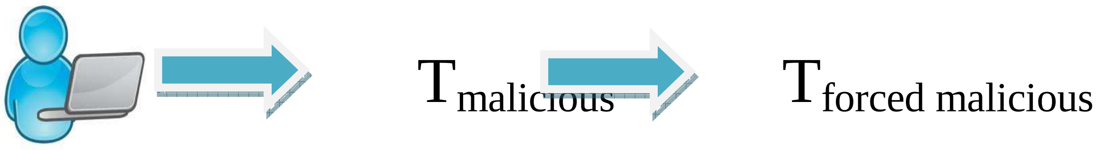


Figure 6. Forced Malicious Transaction

Forced dependency is managed either by never allowing the dependency to set in or dealing with dependency during execution or commit of the transaction. For this we maintain a reads-from dependency graph. A transaction identified as malicious is rolled-back along with any transaction which is dependent on this malicious transaction. In situations where a deadlock is present, we identify the Forced malicious transaction and roll them back as well which in some cases can result in rolling back of all the transactions present in the cycle.

4.2 Constraint Violation

As stated in previous section that the framework of the identification of MTs is based on the constraint model. This constraint model is categorized into three possible types of violation.

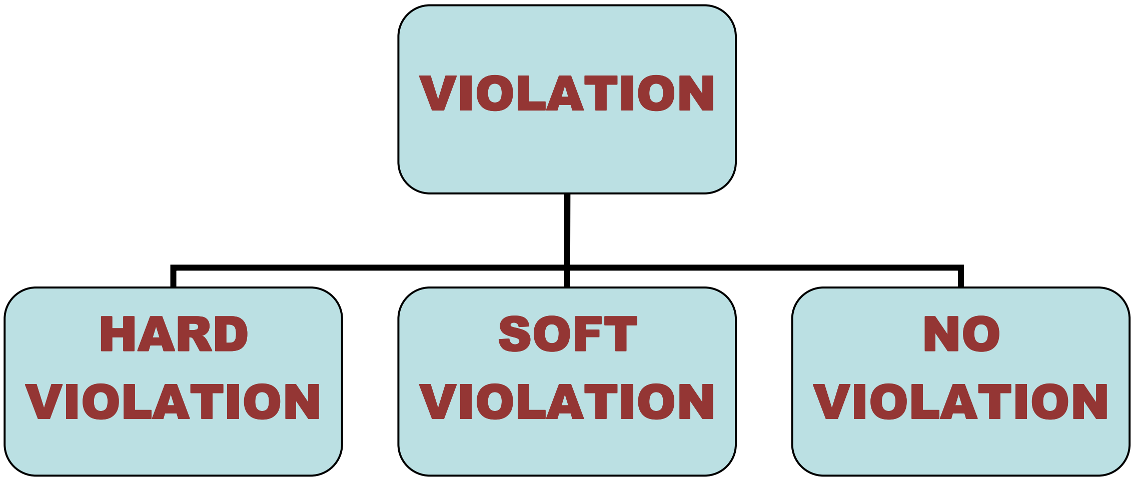


Figure 7. Types of Constraint Violation

For categorizing the transactions into one of the three forms of violations we define a parameter "k" which is the number of constraints employed. It depends on the transaction and the chosen constraint model.

- Hard Violation: When a transaction violates at least k constraints then such violation is classified as Hard Violation.
- Soft Violation: When a transaction violates less than k but at least one constraint then such violation is classified as Soft Violation.
- No Violation: When a transaction does not violate any constraints then such violation is classified as No Violation and the transaction is identified as a good or clean transaction.

The constraints rule can be adjusted and changed for different databases. We can identify known patterns of attacks and can then use them to construct the set of constraints and then use them in the local mobile database.

4.3 ACIDS – Transaction Model for Intention Detection

As discussed in Chapter 2, ACID properties are not sufficient enough to identify malicious transactions in a Mobile Database System. This is because ACID properties do not identify the malicious intention of the user. For this reason we need to enhance the ACID model to ACIDS, where the fifth element 'S' stands for Safe. Thus any transaction issued by the user to MDS will have ACIDS properties. ACID properties will be handled through conventional database model approaches while S property will be satisfied through the Safe Analysis Protocol. The SAP protocol used for identifying the malicious intention of the user follows a twofold analysis approach of User Analysis and Transaction Analysis.

4.4 Safe Analysis Protocol

SAP is used for identifying the malicious intention of the user by doing User Analysis and Transaction Analysis. User Analysis is done through User Profile. A User Analysis triggers a transaction analysis where the transactions data requirements and their values are examined.

- User Analysis: User Analysis is basically the analysis of the User Profile of the user and is built on the basis of SAFE-credit score of the user. This score is a dynamic number and determines the risk profile of the user.
    - Safe Credit: A SAFE-credit value is used to provide indication of the intention of a user. A user earns a positive SAFE-credit for each good transaction and loses points for each MT. The threshold SAFE-credit limit is set to 70. When a new user enters the system a value of 65 SAFE-credit is assigned which is incremented or decremented depending on the intention of the user while performing the transaction.
- Transaction Analysis: Once the credibility of the user has been determined from the User Analysis, system performs the Transaction Analysis which is a multiple constraint check process. During this process SAP checks for type of transaction, amount of withdrawal, transfer or deposit. One of the most important constraints checked during this process is location from where the transaction is executed. Amount, Time of the transaction also helps in detecting the intention of the user.

4.4 Constraint Bank and SAP weight

SAP analysis involves verifying different constraints while analyzing a transaction. These constraints are stored in a Constraint bank and can be easily adjusted to suit the need of a particular organization.

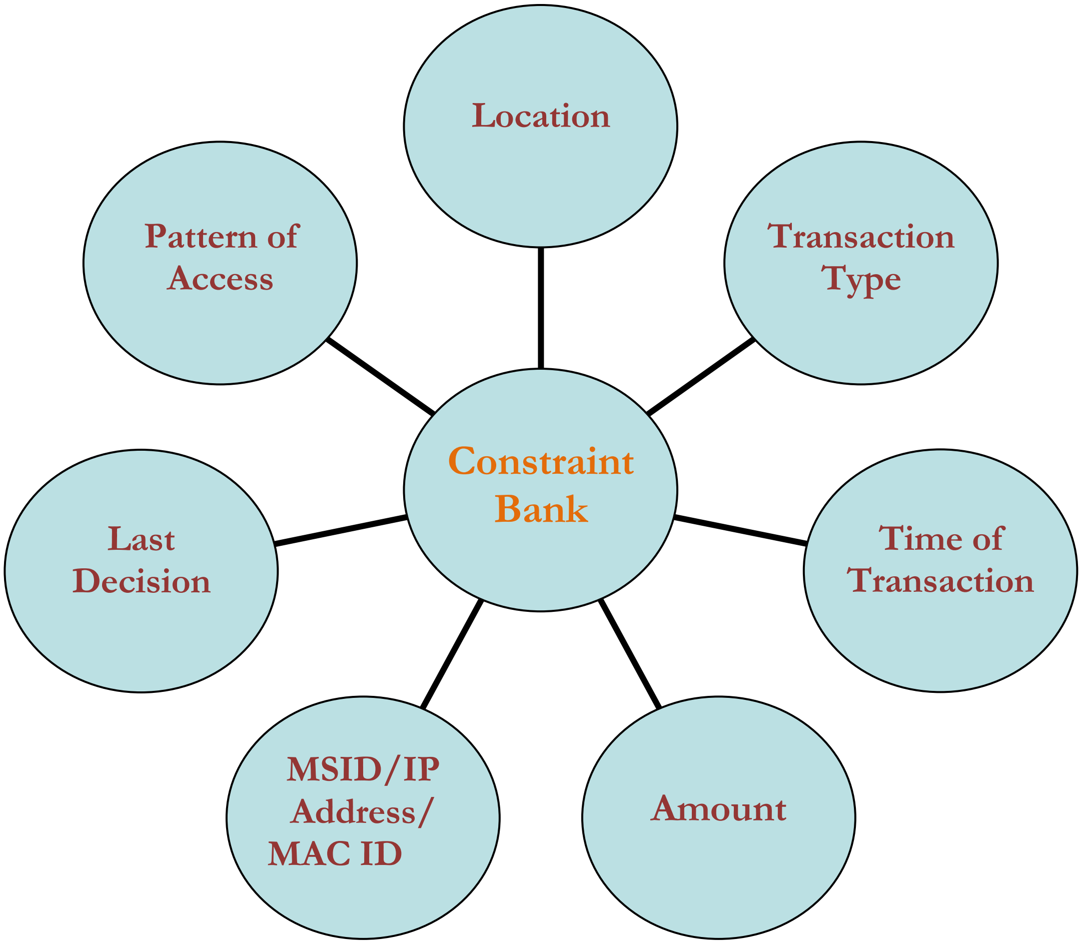


Figure 8. Constraint Bank

Every constraint carries a SAP weight which determines the weightage of that constraint for deciding the malicious nature of the transaction. The system checks the user SAFE-credit and depending on the score decides the threshold SAP-weight. Two threshold weights are assigned, one for determining Soft Violation and other for Hard Violation. Transaction above the threshold limit of Soft Violation are flagged and allowed to proceed but those above the limit of Hard Violation can be safely assumed as Malicious Transactions and appropriate actions can be taken for such transactions. For a user with a good/high value of SAFE-credit score the transactions initiated by this user can be assumed safer to start with, so the threshold SAP-weight assigned to this user will be higher. Hence the transactions initiated by this user have good chances of not crossing the SAP-weight threshold. The weightage assigned to these constraints can also be adjusted depending on the needs of organization.

CHAPTER 5

SIMULATION AND RESULTS

5.1 Simulation Components

The software used in this simulation is VB.Net. The simulator has been programmed in a modular fashion with the following components:

- Arrival Event: Schedules the next arrival event of the transaction based on Poisson distribution and places the arriving transaction in an Arrival queue.
- CPU Event: Transactions are picked in a FIFO fashion from the arrival queue. This event:
  - Decides the random number of data items for the transaction to be processed
  - Decides the data items calculated from random number generator for the transaction to be processed
  - Decides the read and write operation on the data item.
  - Locks the data item being processed and puts it in lock table
  - Handles the blocking mechanism
  - Puts the data item in I/O queue
- I/O Event: Clears the lock and block tables as per the status of the data items and puts the transaction back in Arrival queue

Other than these major components simulator is comprised of the following other components:

- Random number and Poisson distributed event generator.
- Data size and Data item computation: Calculates the data size and the data items of the transaction with random number generator.
- Constraint Bank: Stores various constraints used in the simulation along with their assigned weightage.
- Locking mechanism: Locks the data items of the transaction being processed by the CPU.
- Blocking mechanism: Blocks the transaction which require the same data items as is by the processing transaction.

5.2 Working of the Simulator

Following is the step by step working of the simulator.

- Simulation starts with all three events at 0 time and transaction T1 in arrival queue.
- Simulator scans the event table and picks up the event with the least time value. In case, two or more events have equal small value, the events are picked on the following priority

  Arrival event → CPU event → I/O event.

- In the start, since all the three events have same value of ‘0’ Arrival Event will be selected.
- When Arrival event is executed we get the value of the next schedule arrival event based on Poisson distribution.
- When CPU event is selected, it picks up the first transaction in the Arrival queue based on FIFO. It randomly selects the data items and the required operations on those data items. While it proves the transaction it places the data items required by the transaction in a Lock Table. It also performs a check to see if the data items required by the transaction are already getting processed and are in the Lock table. In this case, it places the transaction in the Block table. It checks the Block table in a regular interval to make sure if the required data item becomes available.
- Before CPU event actually process the data items it analysis the user profile to check for the SAFE-credit. Based on the SAFE-credit score it decides for the SOFT Violation and HARD Violation thresholds. Next the transaction is analyzed to verify other constraints. As soon as a constraint is violated the SAP-weight value is increased. After verifying all the constraints, SAP weight is checked against the threshold Violation and transaction is declared as “Good” or “Malicious” accordingly. For a “Good” Transaction, SAFE-credit of the transaction is incremented and decremented for the “Malicious” transaction.

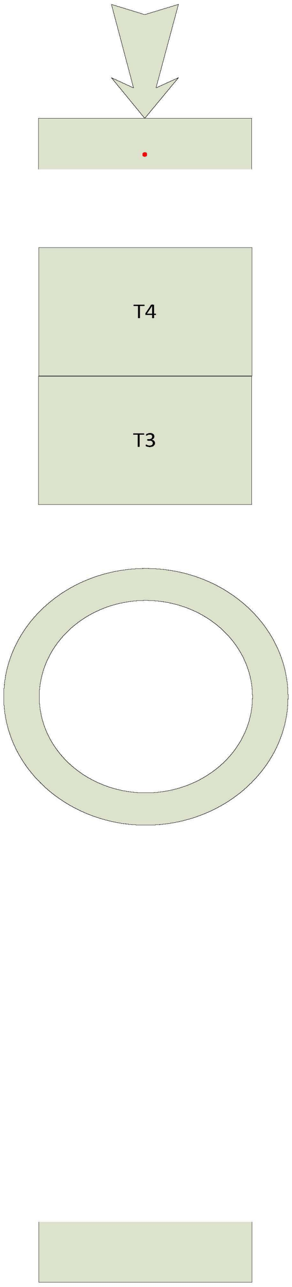


Figure 9. Arrival, CPU and I/O events

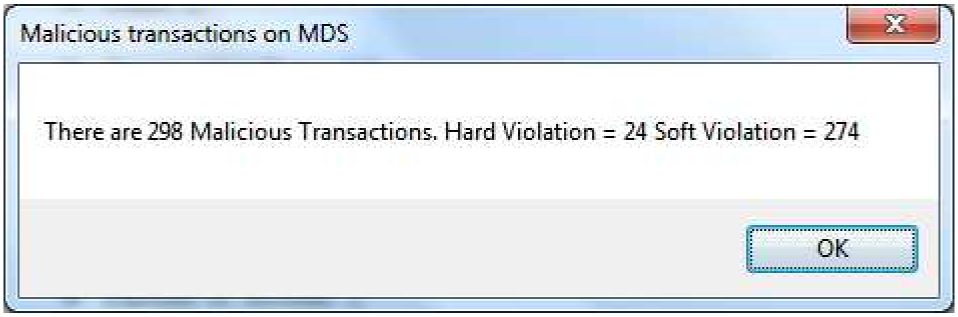
5.3 Test Data and Output

For simulation different users with different constraints and profile were selected. Following is the test data selected for the simulation:

- Total number of users: 8
- Total number of transactions executed: 1000
- Number of data items in repository: 5000
- Minimum data size: 3
- Maximum data size: 10
- Constraints and their SAP-weight:
    - Location: 2
    - Time: 2
    - Transaction Type: 1/3
    - Amount: 3
    - Last Decision: 3
    - Pattern of Access: 2
- SAFE-Credit Threshold: 70
- SAP-weight for users with SAFE-credit score more than 70
    - For Hard Violation: >7
    - For Soft Violation: = 6 or 7
    - For No Violation: <6

- SAP-weight for users with SAFE-credit score less than 70
  - For Hard Violation: >6
  - For Soft Violation: = 5 or 6
  - For No Violation: <5

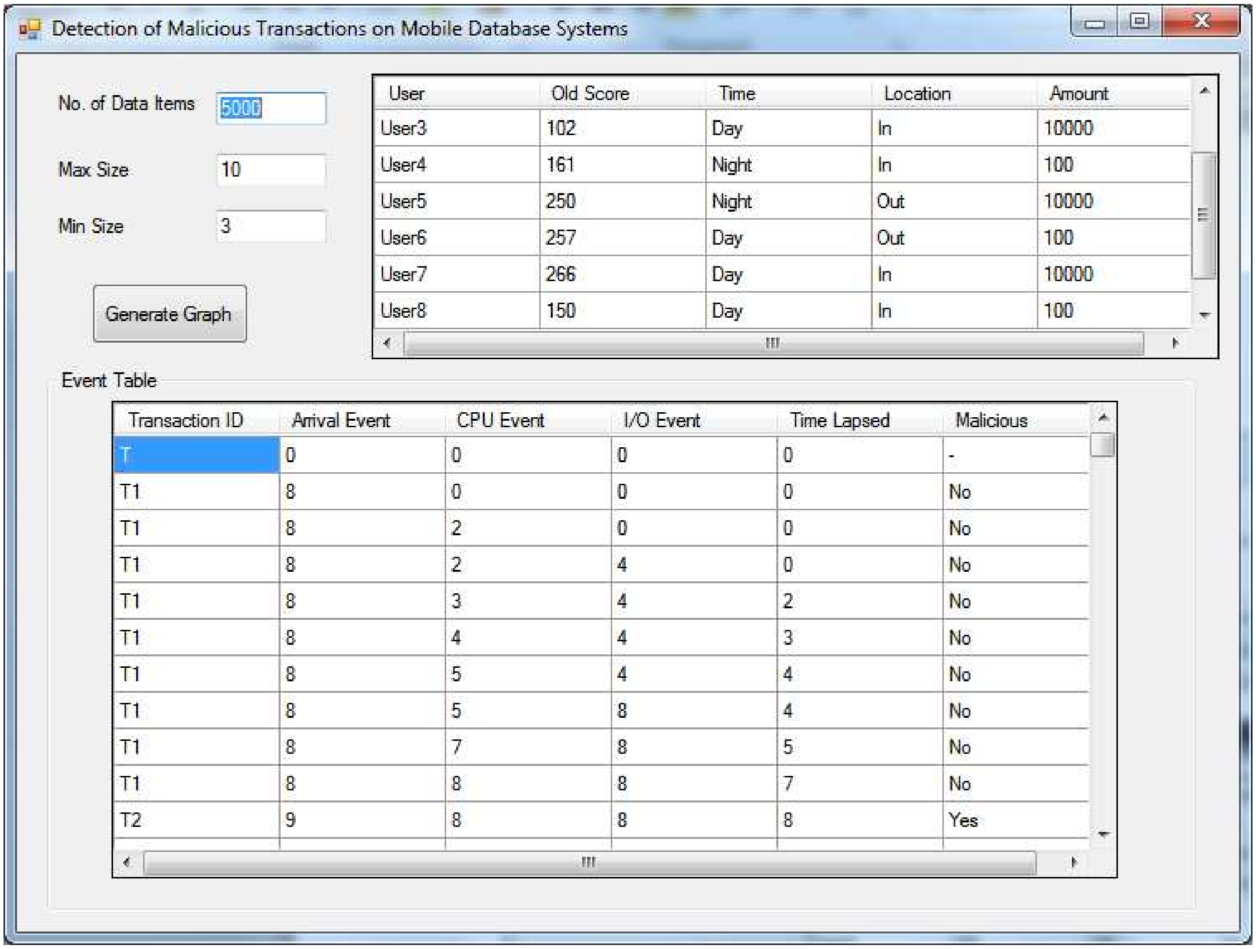
After the simulation completes processing of 1000 transactions it reports the number of Malicious Transactions found along with the number of Soft and Hard Violation. For example as per the following figure there are 274 Soft Violation and 24 Hard Violation i.e. total numbers of malicious transactions are 298 out of 1000 transactions



Malicious transactions on MDS

There are 298 Malicious Transactions. Hard Violation = 24 Soft Violation = 274

Figure 10. Simulator Output


Detection of Malicious Transactions on Mobile Database Systems
Generate Graph

No. of Data Items 5000

Max Size 10

Min Size 3

User	Old Score	Time	Location	Amount
User3	102	Day	In	10000
User4	161	Night	In	100
User5	250	Night	Out	10000
User6	257	Day	Out	100
User7	266	Day	In	10000
User8	150	Day	In	100

Event Table

Transaction ID	Arrival Event	CPU Event	I/O Event	Time Lapsed	Malicious
T	0	0	0	0	-
T1	8	0	0	0	No
T1	8	2	0	0	No
T1	8	2	4	0	No
T1	8	3	4	2	No
T1	8	4	4	3	No
T1	8	5	4	4	No
T1	8	5	8	4	No
T1	8	7	8	5	No
T1	8	8	8	7	No
T2	9	8	8	8	Yes

Figure 11. Malicious Transaction Detection Simulator – Detailed Output

5.4 Graphs

Following figures depict the output obtained from simulator in a graphical manner. The conclusion drawn from these graphs is that number of malicious transactions depends on the number of constraints put in place. These graphs are plotted on the assumption that user test data consists of 50% of users with SAFE-credit score below the threshold. The sample user initiating the transaction is randomly picked up from the given selection of test users.

Graph 1: Malicious Transactions vs. Non Malicious Transactions: Considering total number of transactions as 1000, with 6 constraints put in place we received an output of 298 malicious transactions

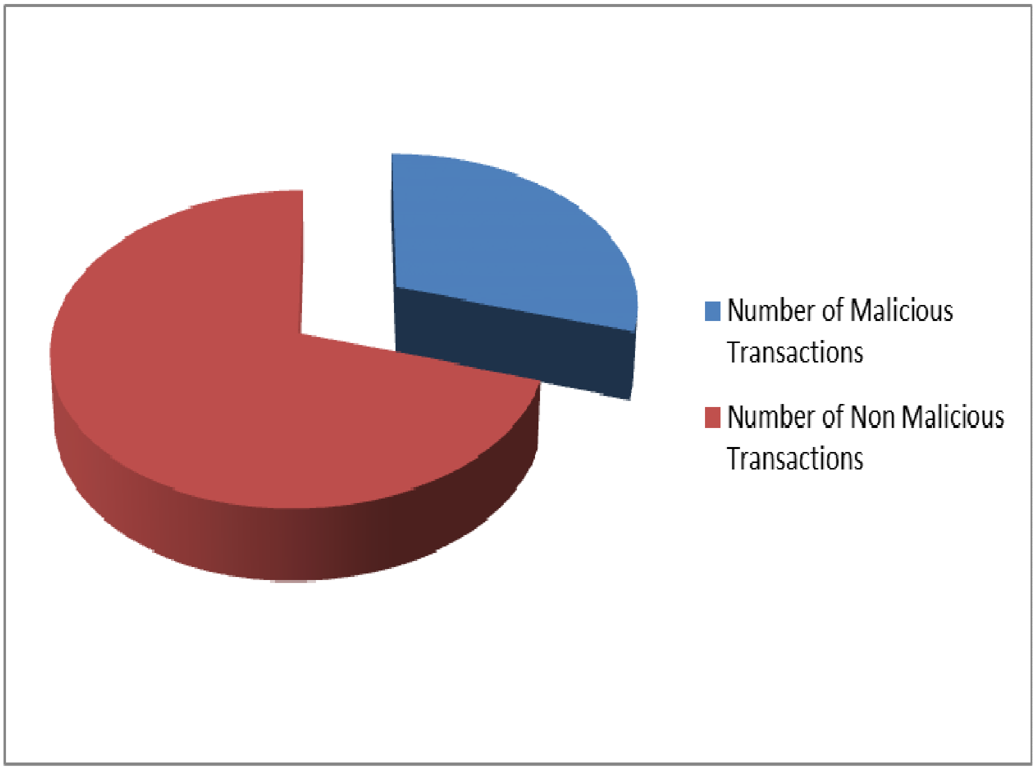


Figure 12. Graph 1 - Malicious Transactions vs. Non Malicious Transactions

Graph 2: Soft Violations vs. Hard Violations: Out of the total 298 malicious transactions, 274 were found as Soft Violations and 24 as Hard Violations.

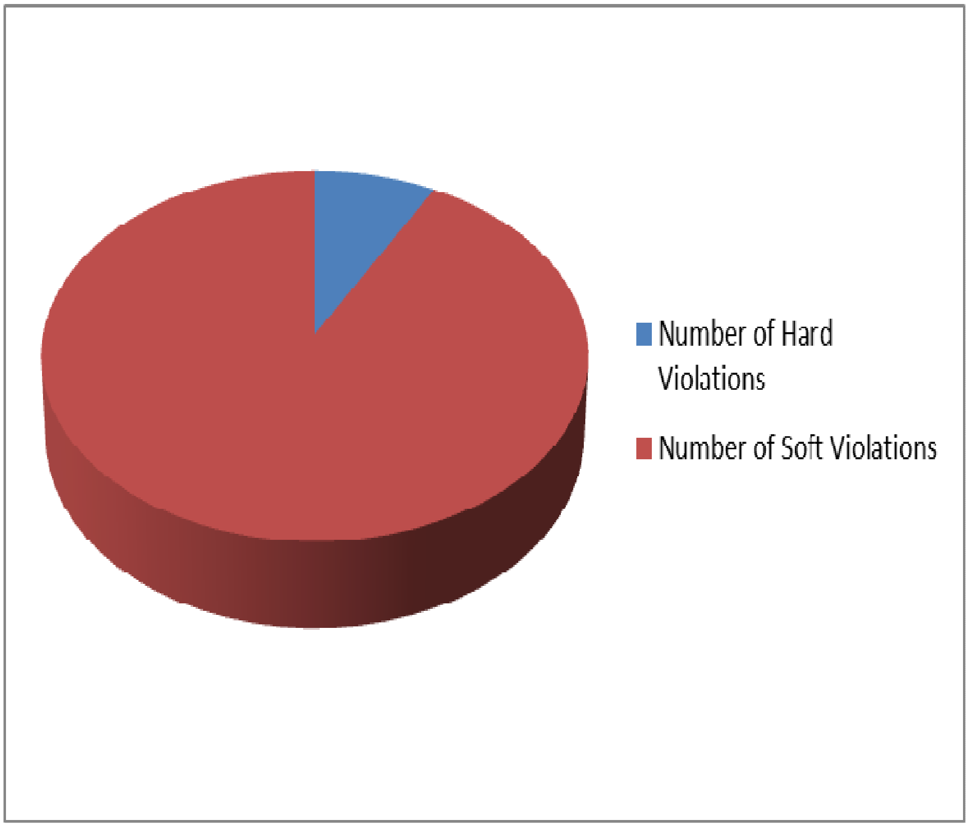


Figure 13. Graph 2 - Soft Violations vs. Hard Violations

Graph 3: Total Arrival event of malicious transactions: The following graph shows total malicious transactions at any point of the arrival event. As expected the graph shows an increment in the number of malicious transactions but does not follow a definite increment curve. The given graph is a segmented graph and the data is captured for transaction number 1 to 100.

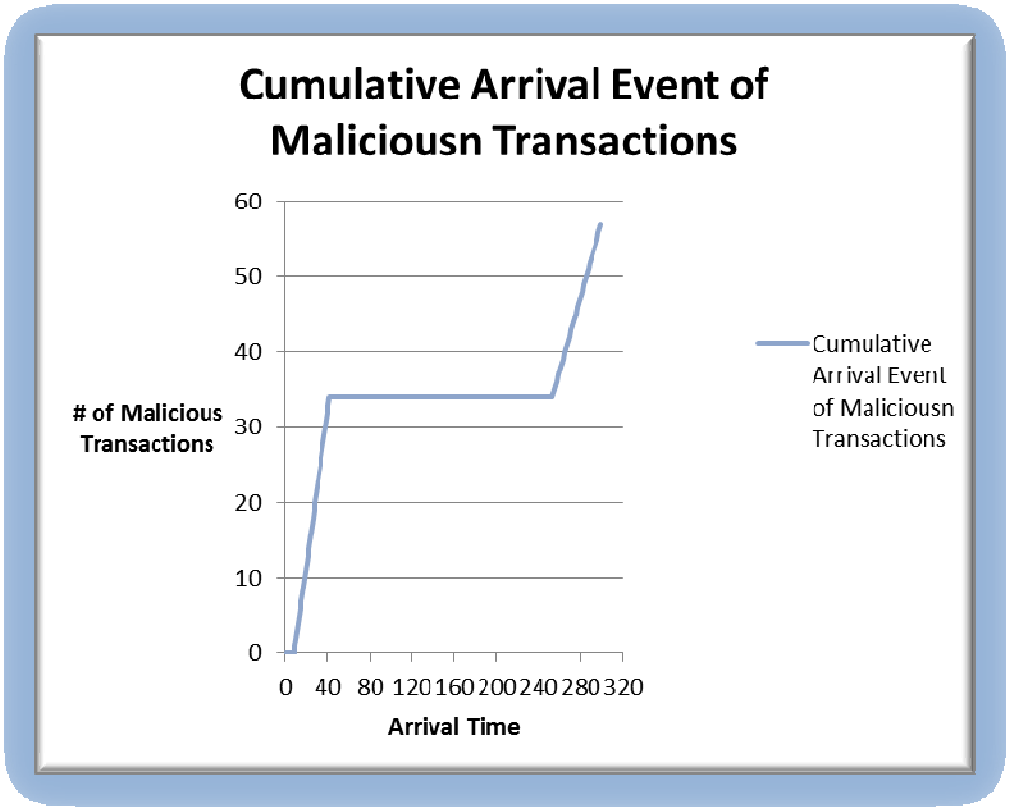


Figure 14. Graph 3 - Total Arrival event of malicious transactions

Graph 4: Number of malicious transactions vs. Arrival rate: The following graph shows the change in number of malicious transactions with the change in arrival rate. The graph shows a steep rise in the number of malicious transactions as the simulation begins. However the graph gradually declines after some time, this is because the user profile is build up by now and the sample users picked up by the transaction has a better low risk profile leading to less number of malicious transactions.

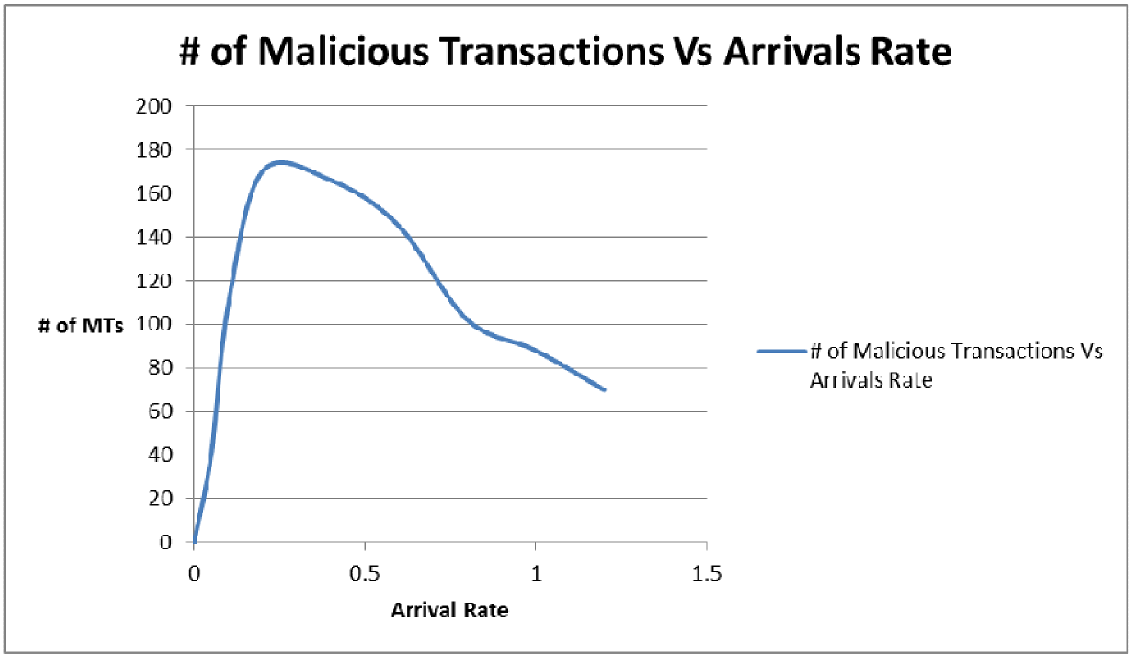


Figure 15. Graph 4 – Number of Malicious Transactions vs. Arrival rate

Graph 5: Type of malicious transactions vs. Arrival rate: The following graph shows the number of different types of malicious transactions with the change in arrival rate. The graph shows a steep rise in the number of malicious transactions as the simulation begins. However the graph gradually declines after some time, this is because the user profile is build up by now and the sample users picked up by the transaction has a better low risk profile leading to less number of malicious transactions.

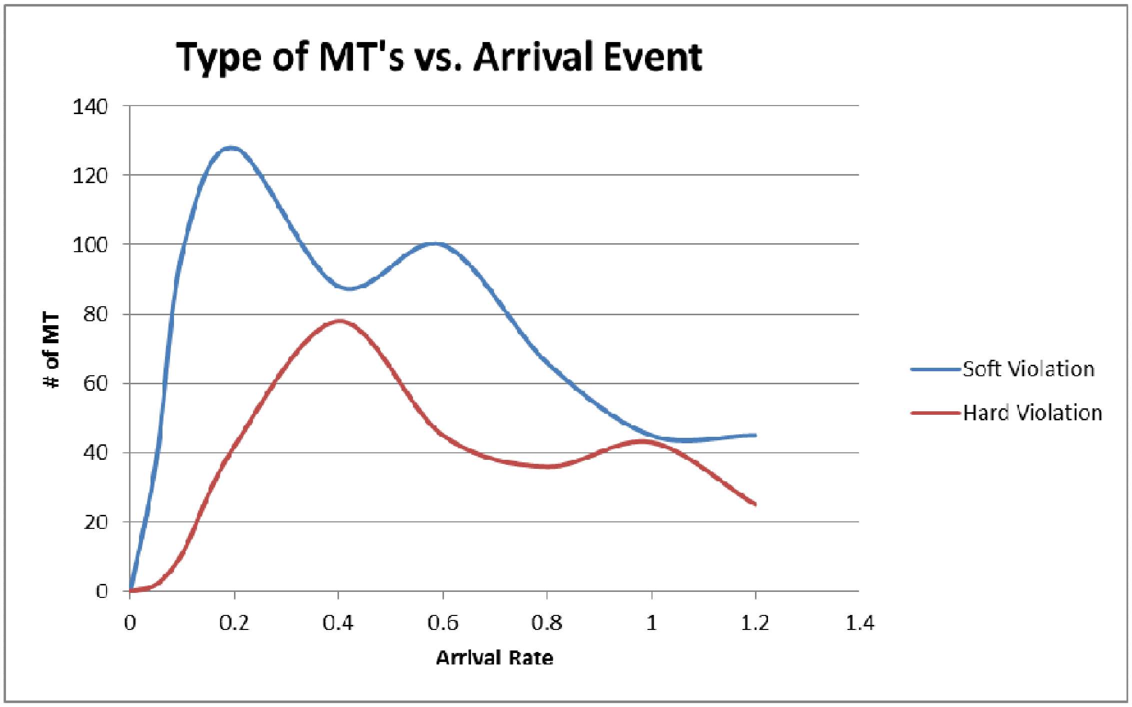


Figure 16. Graph 5 – Types of Malicious Transactions vs. Arrival rate

Graph 6: Throughput: The following graph shows the total number of non-malicious transactions with the arrival event. This graph gives the total number of processed transactions. Since malicious transactions are rolled back as soon as they are detected as malicious only non-malicious transactions are considered for throughput.

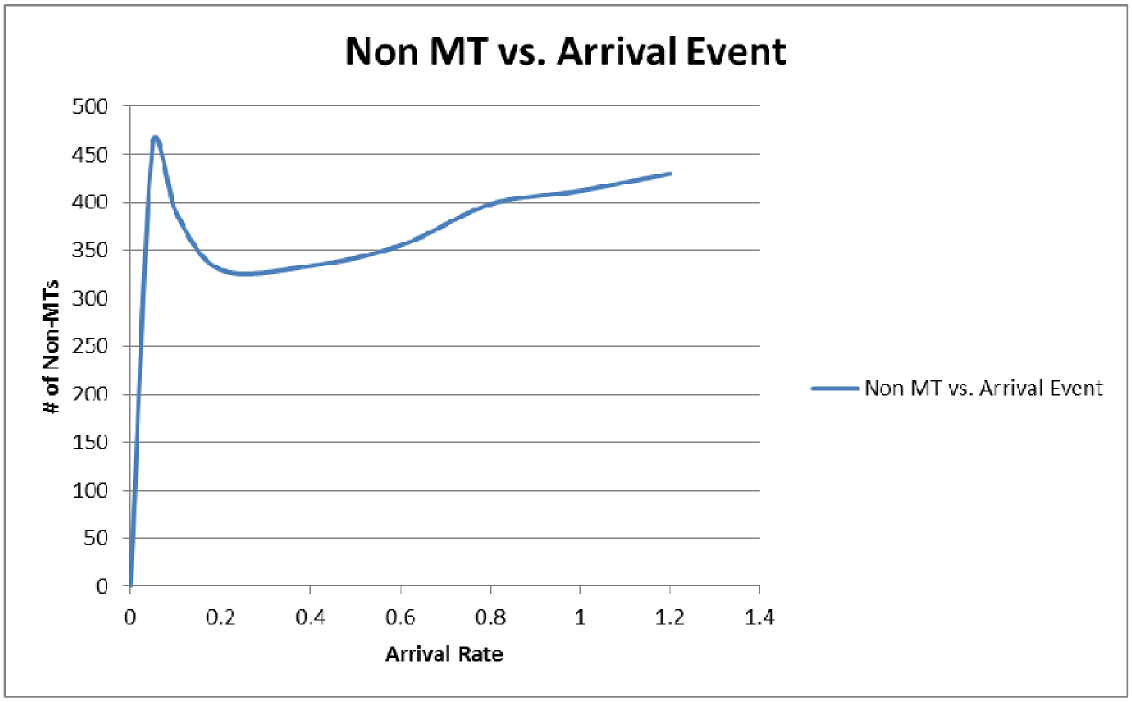


Figure 17. Graph 6 – Throughput

CHAPTER 6

FUTURE WORK

As the mobile network-computing environment increases in complexity, so does the functional requirements of Detection of Malicious Transactions in Mobile Environment. Common functional requirements of a Malicious Transaction detection system being deployed in operational environments can include the following:

- Detection system must continuously monitor and report intrusions.
- Malicious transaction detection systems deployed in a financial organization should have a very low false alarm rate. This will depend on wisely choosing the constraints and SAP weight which as per the need of organization can be adjusted as per the employee.
- The Malicious transaction detection systems should be able to learn from past experiences and improve its detection capabilities over time. Self-tuning system will be able to learning from false alarms with the guidance of database administrators and eventually on its own.
- This system should be able to be easily and frequently updated with attack signatures as new security advisories and security patches become available and new vulnerabilities and attacks are discovered.

- Data mining tools will be helpful in running statistical analysis tools on processed data in support of malicious detection techniques.
- Should be able to handle rapid changes in network conditions with limited network administration.
- The system can be designed with complete security in mind. For example, the system can perform the tasks of authentication, authorization and administrator and audit user's actions authenticate devices and perform various security tasks.
- The system can be tuned to provide optimum performance and can be designed in such a way as to not place undue burden on the network. A system that is functionally correct, but that detects attacks too slowly is of little use.
- The system can be made more scalable. As new mobile computing devices are added to the network, the system must be able to handle and process data in every type of system.

CHAPTER 6

CONCLUSION

Absolute security is an abstract concept – it does not exist anywhere. All networks are vulnerable to insider or outsider attacks, and eavesdropping. No one wants to risk having the data exposed to the casual observer or open malicious mischief. Regardless of whether the network is wired or wirelesses, steps can and should always be taken to preserve network security and integrity.

Any secure network will have vulnerabilities that an adversary could exploit. This is especially true for wireless ad-hoc networks. Malicious Intention Detection system can complement intrusion prevention techniques (such as encryption, authentication, secure MAC, secure routing, etc.) to improve the network security. However new techniques must be developed to make malicious detection work better for the wireless networks.

Malicious Intention detection system developed in this research is a involved process and modeling complexity increases with the number and types of constraints to be modeled. This software doesn’t standardize anything. The proposed system flags transactions as malicious if they are largely deviant from the expected behavior. To do this, it builds user profiles and these profiles are checked and updated every time a user performs an action. It also checks the transaction parameters against a set of defined constraints. The system proceeds this way, accumulating more and more evidence for a malicious attempt until a threshold is crossed; at this point, it signals an malicious attempt. The malicious detection component analysis the transaction and updates the user profile accordingly.

BIBLIOGRAPHY

[1] Acharya, D., Kumar, V. (June, 2005). Indexing Location Dependent Data in Broadcast Environment. *Journal of Digital Information Management (JDIM)*, Volume 3, Issue 2, 41-112.

[2] Acharya, D., Kumar, V. (June, 2005). Locations based Indexing Scheme for DAYS. *4th International ACM Workshop on Data Engineering for Wireless and Mobile Access.* Baltimore, Maryland, 17-24.

[3] Agrawal, R., Chrysanthis, Panos K. (June, 2001). Efficient Data Dissemination to Mobile Clients in E-Commerce Applications. *Proceedings of the Third International Workshop on Advanced Issues of E-Commerce and Web-Based Information Systems.* San Jose, CA, 58.

[4] Gadiraju, S., Kumar, S., Dunham, M. (April 2004). Recovery in the mobile wireless environment using mobile agents. *IEEE Transactions on Mobile Computing*, Volume 3, Issue 2, 180-191.

[5] Pitoura, E., Bhargava, B. (1994). Building information systems for mobile environments. *In Proceedings of the ACM Third International Conference on Information and Knowledge Management*. New York, NY, USA. 371-378

[6] Ammann, P., Jajodia, S., Liu, P. (Sep/Oct 2002). Recovery from malicious transactions. *IEEE Transactions on Knowledge and Data Engineering*, Volume 14, Issue:5, 1167-1185.

[7] Ammann, P., Jajodia, S., Mccollum, Catherine D., Blaustein, Barbara T. (1997). Surviving information warfare attacks on databases. In *Proceedings of 1997 IEEE Symposium on Security and Privacy*. 164 - 174

[8] Liu, Peng, Hao,Xu (2001). Efficient damage assessment and repair in resilient distributed database systems. *Proceedings of the Fifteenth Annual Working Conference on Database and Application Security*. 75-89.

[9] Liu, P., Jajodia, S., (June, 2001). Multi-Phase damage confinement in database systems for intrusion tolerance. In *Proceedings of 14th IEEE Computer Security Foundations Workshop*, Cape Brenton, Nova Scotia, 75-89.

[10] Giordano, J., Panda, B. (July 1998). Reconstructing the database after electronic attack. In *Proceedings of the Twelfth IFIP WG11.3 International Working Conference on Database Security*, Chalkidiki, Greece, 143-156

[11] Haque, A., Panda B. (March 2002). Extended data dependency approach: a robust way of rebuilding database. In *Proceedings of the 2002 ACM Symposium on Applied Computing*, Madrid, Spain, 446-452

[12] Yu, M., Liu, P., Zang, W. (December, 2003). Multi-version attack recovery for workflow systems. In *Proceedings of the 19th Annual Computer Security Applications Conference*, Pennsylvania State Univ., University Park, PA, 142-150.

[13] Zuo, Y., Panda, B. (July, 2004). Damage discovery in distributed database systems. In *Proceedings of the 18th IFIP WG11.3 Working Conference on Data and Applications Security*, Volume 144, 111-123.

[14] Lala, C., Panda, B. (Jul 2001). Evaluating damage from cyber-attacks: a model and analysis. *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans*. Vol:31, Issue:4, 300-310.

[15] Dunham, M., Kumar, V. (1998). Location dependent data and its management in mobile databases. In *DEXA '98 Proceedings of the 9th International Workshop on Database and Expert Systems Applications.* IEEE Computer Society Washington, DC, USA. 414-421

VITA

Abhiruchi Singh was born in Korba, India. She got her B.S. in Electrical Engineering with distinction from NIT, Raipur, India. She joined her M.S. program in the Department of Electrical Engineering at UMKC in the fall semester of 2010 and graduated in January 2013. Prior to that she worked in Tata Consultancy Services, Mumbai India as a Sr. Software Engineer. Presently she is working as a Software Engineer in Cerner Corporation in Kansas City, MO, USA.