1. Introduction

With rapid advancement of communication technology and computing devices, today’s era is the era of mobile computing. All service oriented applications are now ‘all time, everywhere’ services driven by user demands. The systems that will support such services demand fault free performance. However, in all practicality to design fault free environment is impossible and that too particularly in mobile computing. The pressing demand that rises here, is how to ensure a specified performance, albeit, in the presence of faults. The answer seems to lie in fault tolerance. Fault-tolerance can be defined as “how to provide, by redundancy, service complying with the specification in spite of faults having occurred or occurring” [1]. “Everywhere” services include location independent services like, banking, financial transactions, e-commerce etc. In the domain of academics, collaborative research work among a number of researchers located in different geographical areas but working on same domain is a continuous process. Needless to say, availability of services through systems is the core demand of the day. Hence non-availability due to some failure is going to create inconvenience to users. In all such systems, fault tolerance is much needed.

The computing environment considered here is mobile computing system. It is widely accepted that checkpointing and recovery is one of the most effective techniques for providing fault tolerance in distributed system, and thus, mobile computing system. However, with characteristics specific to mobile computing systems, security threat becomes an issue to be dealt with. Hence the checkpointing and recovery technique should also be secured for mobile computing systems. Along with these, security of checkpoints is an important issue that must be dealt with while assuring successful recovery of failed mobile hosts (MH). This can be implemented using security and trust evaluation methods. The unique characteristics of mobile computing system, the need of
fault tolerance using checkpointing and recovery, overhead issues of such techniques, security of checkpoints and basis of trust of mobile hosts are illustrated in this chapter.

1.1 Mobile Computing

Rapid advancement in communication and networking technology has given birth to a new method of computing – mobile computing. Mobile computing is computation in portable computing devices capable of wireless networking. The current era is the era of communication and computation while users are mobile. Mobile computing can be characterized by three properties: wireless communication, mobility and portability. Mobile computing enables users to compute and transmit data from remote locations to other remote or fixed locations. This facility enables different business transactions, collaborative research works in group, aircraft control, automatic health monitoring using electronic devices, etc, irrespective of physical location of the computing devices. Mobile computing enables users to access services and resources of fixed infrastructure also. Wireless network is able to link to static infrastructure using stationary transceiver. In comparison to wired network, wireless communication suffers from noise, low bandwidth, physical obstructions, and frequent disconnections. If a mobile user moves out of coverage area of static transceiver or enters into an already overloaded coverage area, then there is possibility of disconnection from network. Wireless networking is more challenging than traditional wired networking as mobile devices may traverse through different heterogeneous networks where transmission speeds and protocols etc. vary. Security attacks, threats, hazards are perceived more in wireless networks than wired networks. The portable computing devices, called mobile hosts (MH), (used as nodes interchangeably), do not work in physically protected area, hence get vulnerable to security attacks. Though mobile computing suffers from various limitations and challenges, the technology offers facility to access and use fixed resources through fixed access points (considered to be the point of attachment to a network) at all time, from every locations. Hence, tasks like accessing resources, communication, computing can be done irrespective of time and location of the computing devices, that is, mobile users. Computing is bounded by location of user/device in wired networks. In addition to this, wireless networking and portable devices facilitate computing while user location
changes. One of the examples of such infrastructure based wireless network is wireless cellular network in which there are large number of mobile hosts and a few mobile support stations (MSSs) that work as fixed access points through which mobile hosts can access fixed resources like stable storage. Advancement of infrastructure based wireless network towards infrastructure less wireless network has changed further the definition of mobile computing. Wireless network still needs connection to stationary transceiver in cellular technology. Portable devices are not capable of wireless computing and communication in the places where stationary resources are not present. The wireless network in which fixed infrastructure is absent is known as ad hoc network. This type of wireless ad hoc network coupled with the mobile devices gives rise to mobile ad hoc network (MANET). In wireless adhoc network, computing devices may have fixed locations but without any fixed or static infrastructure whereas in MANET, computing devices themselves are mobile, hence may neither have any fixed locations nor any fixed infrastructure. In different areas where it is difficult to establish fixed infrastructure, ad hoc network can be established with multiple portable devices having transceivers that can cover a limited range. MANET can be considered to be either flat network or cluster based hierarchical network. Total coverage area of MANET and its size are generally smaller compared to wireless cellular network. Time and cost of network establishment are lesser for MANET than that for wireless cellular network. A couple of disadvantages also come along with these advantages: no centralized control, no fixed access point or stable storage support, dynamic topology due to movement of mobile nodes, no separate routers, limited resources of portable devices, vulnerability of mobile hosts and wireless links towards security threats and attacks. Portability and scalability along with other advantages make MANET suitable for communication in different difficult-to-reach areas e.g. disaster affected areas (like earthquake), flood affected areas, etc., battle field, mining areas, to name a few.

1.2 Fault Tolerant Mobile Computing System

The property of fault tolerance of a system is conceptualized on the basis of assured expected performance in presence of faults. Simply put, a fault tolerant system is one which is able to perform normally while tolerating faults. Hence in a failure prone
system, performance is measured by accuracy in performance along with ability to tolerate faults. Now, how can fault in a system be defined? Wrong input parameters or incorrect design specifications, etc. may give rise to fault. Moreover in a software application, incorrect code or wrong algorithm or logic may lead to fault. Due to fault system performance may deviate from expected or desired output. This deviation in system output is termed as error. Depending on severity of error the system may fail. So, if fault, error and failure are categorized according to their level of abstraction, then fault is lowest level of abstraction, error is next higher level of abstraction, failure is the highest level of abstraction. Fault that causes error, error that in turn leads to system failure are part and parcel of any system, hardware or software. As counter measures there are different traditional techniques, e.g. fault removal, fault avoidance, fault detection and correction etc. Fault removal is an exhaustive process. Full fault removal is unrealistic and impossible to achieve in finite time. Fault detection and correction are also quite exhaustive. Every time an error occurs in a running application program, detection and correction of corresponding fault may cause following problems: disruption of normal execution flow of the program for fault detection, arbitrary time for fault correction. The assumptions that fault correction will improve performance of a system may not hold true always. One of the preferred options is fault avoidance. “Fault avoidance techniques are employed to avoid introducing faults into the system”[2]. However, fault avoidance may impose design restrictions that are not desirable. Hence fault tolerance[2] becomes a better option with respect to design and development of a system. Fault tolerance is acknowledging the presence of fault in the system, with a technique to bypass the fault and resuming normal execution. However, tolerating faults involves detecting a failure, gathering knowledge about the failure and recovering from that failure to continue execution.

Now-a-days the world around users is dominated by computers and networks. Service oriented software systems were generally distributed systems that converted to mobile distributed systems and finally to mobile infrastructure-less systems. Use of application programs has increased by volumes. These application programs are mostly deployed to avail real time services, for example e-commerce, financial transactions, banking etc. These services should be available all the time. Some of the systems are
safety critical systems in which system availability along with accuracy and consistency of data are equally needed. Examples of such systems include controlling unit of a power plant, flight control, specifically nuclear power plant, launching of a missile, etc. Other service systems like online reservation, online shopping etc. require availability, whereas safety critical systems are more focused on data accuracy and consistency along with availability. Traditional approach to achieve availability of a system and data accuracy is through implementation of fault tolerance. Todays service oriented systems are mostly based on client-server architecture, distributed, mobile networked environment so that users can avail services ‘all time, everywhere’. These systems can most suitably be defined by availability, reliability, data accuracy and consistency. Size and complexity inherent in a software system is increasing everyday [3]. Due to complexity, effort needed to detect and correct errors increases. Hence, fault tolerance appears to be the most suitable technique to be provided to such systems.

Mobile Computing systems are combination of computing and communication. These systems can grow without bound. Thus system complexity, uncertainty and failure probability increase. These systems are used to provide uninterrupted services to users in different sectors. Hence, along with other attributes, fault tolerance is must in these systems. These systems are also vulnerable to security attacks. Security attacks and threats can cause fault and hence failure of the system. To maintain correct services of the system inspite of occurrence of faults, fault tolerance is must. Thus security and fault tolerance are interrelated to each other. Another issue that crops up in such systems is cooperative trust or belief among each other. This concept is important in MANET where mobile host is itself router and one node has to forward data through intermediate nodes to communicate with destination node which is out of sender node’s transmission range. It is obvious that every node forwards its important data to its next node with a belief that the node is not a malicious or selfish [4] node and it will forward the data correctly towards the destination node. Now this belief should be verified based on specific parameters. One popular method of evaluation of belief of a node is through recommendations from other nodes that have previously interacted with that particular node. This concept is defined by trust [5] of a system: “trust in a system is a belief that it will resist malicious attacks”. Security in a system is implemented so that the system can
resist malicious attacks. So it can be established that if a system is believed to be trusted, it would be secure or vice versa.

Checkpointing and rollback recovery algorithms are well established techniques to provide fault tolerance to applications running on static or mobile computing devices in both wireless cellular network [6][7] and MANET [8]. Mobile computing devices and hence the applications running on them are more prone to security attacks and threats than static computing devices. Process states of user applications running on devices are saved in checkpoint [9]. Checkpoint is also vulnerable to security attacks while it is saved or transferred through mobile devices. If checkpoint content gets modified then recovery would be unsuccessful leading to unsatisfactory performance [10]. So, checkpoints need to be protected too from security attacks.

Above observations help to identify that to achieve fault tolerance in true sense in security attack prone mobile computing system, checkpoints must be secure. So fault checkpointing-recovery must be combined with security techniques and trust evaluation methods as shown below:

![Figure 1-1 Relationship among Checkpoint-Recovery, Security, Trust and Fault Tolerance](image)

1.2.1 Checkpointing and Rollback Recovery in Mobile Computing

In mobile computing system, processes execute, communicate with each other through wireless network which is linked to a stationary wired network via access points. As mobile devices and wireless network are failure prone, only the stationary node (may be a server) has stable storage. State of a process [11] running on portable computing device is saved on stable storage. A failed process can restart its execution from latest
saved state. The saved process state is known as checkpoint. The resumption of the failed process from last checkpoint is the technique of rollback recovery. The processes running on different devices placed at different locations communicate through message passing. Receipts of computation message, mobile host movement, failure are nondeterministic events. If process A receives a computation message m from process B, then A becomes dependent on B. Now A saves a checkpoint in which receive event of computation message is saved. But B fails without saving checkpoint, so send-event of the computation message is not saved, then it implies that the computation message has been received without being sent. This message becomes orphan [12] and causes inconsistency. During recovery as A will rollback, B will also have to rollback which in turn can cause rollback of other processes dependent on B. This rollback propagation can continue till the initial state is reached. Thus all computations done so far may be lost. This is known as Domino Effect [12]. This is obviously undesirable. This situation has to be prevented at any cost.

1.2.2 Security and Trust Issues in Mobile Computing Systems

Two types of mobile computing systems are considered here: wireless cellular network and mobile ad hoc network. Characteristics and limitations of both the networks have thrown difficult security challenges for both mobile hosts and wireless links. Mobile hosts work in physically unprotected area and hence can be easily compromised by an attacker. Message eavesdropping and fault injection are possible in wireless channels [13]. Moreover, traditional security mechanisms based on certification or on-line servers applicable in infrastructure based wireless cellular network, may not be applicable in MANET as it lacks fixed infrastructure support. In wireless cellular network, mobile support stations provide required infrastructure support to implement cryptography to make checkpoints secure in both the mobile host and wireless links. MANET is more prone to both information and physical insecurity than infrastructure based wireless network because autonomous computing devices and their self-organizing characteristics make implementation of security protocols difficult. Particularly in MANET, the mobile hosts themselves also act as routers. Thus it is important for a sender mobile host to choose a secure route (for the data to be sent) such that the data reaches its destination.
without being compromised. This implies that the intermediate hosts must be reliable. The sender can rely on a host only if it has complete trust on that host. Thus there must be some method of evaluating trustworthiness for the mobile hosts. If the intermediate hosts on a route are evaluated to belong to the category of ‘trusted host’, then data may be routed through trusted hosts securely without encryption. Otherwise the overhead of encryption / decryption for data security has to be incurred.

1.3 Motivations

Literature survey on existing checkpointing-recovery techniques has been done. Almost all of them are suitable for distributed systems. A few of them that are targeted for mobile computing system do not always meet all essential requirements of the system. There is urgent requirement of secure checkpointing and recovery technique because of the following reasons:

1. In mobile computing system, mobile hosts move from one region to the other, many a time leaving behind checkpoints, logs, other data related to computation, dependency information etc. with the old regional head. After such multiple movements, if an application fails, then its recovery information dispersed in a number of different places need to be collected together. This will provide the last saved process state. The costs of searching and retrieval of information get added to the total recovery cost, in terms of time and space complexity.

2. Placing a checkpoint is also an important issue, since retrieval cost has to be minimized (in case of recovery).

3. Mobile hosts often disconnect from network. If checkpoint is not saved properly, recovery may result in undesirable domino effect.

4. Mobile hosts are portable, low weight, small scale computing devices having limited resources. Due to insufficient battery power or memory, applications running on nodes may fail. So failure rate of applications increases in mobile computing system.

5. Maintaining information consistency is a big challenge due to the characteristics (e.g. frequent disconnection) of mobile computing system.

6. Mobile computing environment is prone to security attack. This is because, MHs
are not physically protected and wireless links can easily be intercepted by an intruder. So data saved in memory of mobile host or transferred through wireless links is not secure. Hence data integrity may be compromised.

7. In mobile ad hoc network, mobile hosts themselves perform routing tasks also. So before forwarding data to a MH, the sender MH may want to be assured that the data would ultimately be properly forwarded to its destination.

With the evolution of computer network and computing devices, there are changes in the nature of failure, vulnerability to security attacks. To fit in with the changing environment, with changing characteristics and limitations of devices, checkpointing techniques are also evolving accordingly. Most of the existing works [7], [14] focus on the overhead reduction of checkpointing techniques. This is appropriate for resource constrained mobile computing environment.

Some of the works concentrate on checkpoint placement [14][15] so that at the time of recovery minimum time and effort are spent. Having understood the full implication of placing checkpoints, this has been considered to be an issue and dealt with it.

In mobile computing system, mobile computing devices are autonomous. They can move randomly, communicate and compute independently. Hence, over a period of time, amount of computation updates, mobility rate, inter-process dependence, amount of available resources etc. will be different for different mobile hosts over a period of time. So, checkpointing technique should be designed in such a way that each mobile host should be able to save checkpoints independently based on own requirements.

Independent checkpointing may be done based on specific parameters. However, the appropriate one in mobile computing system, according to our observations, is based on mobility of computing devices. A few of the existing works [16][17] have focused on movement based checkpointing. This also considered this in our work.

Another important aspect that has been explored very minimally in existing works is consideration of movement patterns of mobile hosts along with their mobility. Mobile hosts may move within a cell or cluster over a long period of time or can move across cells or clusters. For intra-cell movement, there will be no handoff [18] but for inter-cell movement, number of handoffs will be high. So an efficient checkpointing algorithm should be aware of this and take measures accordingly.
Authors in [10] have addressed the security issues of checkpoints in mobile computing environment. But they have used private key cryptography which suffers from sharing of secret key in an open environment like wireless network. This has been a great driving force behind proposing a secure checkpointing technique based on public key cryptography.

In MANET, no existing work has focused on security issue of checkpoint. This may be due to the reason that security of checkpoint using cryptographic techniques increases overhead of checkpointing and recovery procedure. If this increased overhead results in high recovery cost and delay, then applicability of such method for improving system performance cannot be justified. The issue of how efficiently checkpointing in MANET can be done so that checkpoints would be secure and that too without adding much overhead has prompted us to combine trust model with checkpointing and rollback recovery algorithm.

Based on above observations, measures needed to be taken to make checkpointing and rollback-recovery algorithms sustainable with ever-changing characteristics and limitations of mobile computing system are described in the following section.

1.4 Objectives

Our work is aimed to find an optimal solution for providing fault tolerance using checkpointing and rollback recovery technique that is aware of mobility and movement patterns of mobile hosts, security issues of checkpoints and trust issues of mobile hosts. The present work aims at developing fault tolerant computing environment where computing devices compute, communicate and move randomly. If a running application fails, then the application could be recovered from last saved process state or from last saved determinants of non-deterministic events ensuring minimum recovery time and cost. Thus the system would be able to maintain a satisfactory performance in failure prone mobile computing environment.

In present work following issues have been considered and suitable techniques are proposed to deal with them.
• **How to take checkpoints?**

As MHs have limited resources, one of the proposed algorithms is a low overhead method that minimizes number of coordination messages so that battery power consumption, bandwidth consumption can be reduced. One of the parameters used by the MH to save checkpoint is based on total number of computation messages sent or received during a particular time interval to keep an upper-bound to inter-process dependence so that checkpoint coordination time can be reduced. Another parameter selected to save checkpoint is based on number of handoffs in case of cellular network and inter-cluster movement count in case of mobile ad hoc network. A threshold of movement count is set previously per application, per user, per environment basis. A MH saves checkpoint if handoff or inter-cluster movement of a mobile host exceeds predefined threshold.

• **Where to save checkpoints?**

After taking checkpoint, mobile host in an infrastructure based network saves it in its memory and sends a copy to its current mobile support station (MSS) to save it in MSS’s stable storage. In case of transient failure, checkpoint can be retrieved from mobile node itself but in case of crash failure, checkpoint would be retrieved from stable storage. In MANET, as cluster head is like any other MH and may not have stable storage, a copy of the checkpoint may be saved in a suitable mobile host e.g. nearest neighbor or the MH that has highest available memory, battery power etc.

• **Checkpoint placement:**

A MH moving in inter-cell movement pattern may move away from the MSS that saves backup copy of its last checkpoint. The MH crashes later. After it is repaired, it will probably join in another MSS and send recovery request to it. New MSS will find and ask the previous MSS to transfer the mobile host’s last checkpoint. Now if distance between these two MSSs is large, then checkpoint transfer cost will be high. So, a better strategy is to keep a threshold of this distance and if a mobile host moves away beyond this threshold, then checkpoint copy will be transferred to the host’s current MSS.
• **Log based recovery:**

Each mobile host saves copy of sent computation message and its determinant in a file called log. The MH that receives a computation message will save determinant of this received computation message as log record. If the receiver mobile host fails before saving checkpoint of this receive event, it will rollback up to last checkpoint. As this mobile host will resume execution, at the point of computation message receive event, it will check log record and find the sender mobile host to which the receiver mobile host will send a request along with the determinant. The sender mobile host will identify the request by checking the determinant and will replay the saved computation message from its log. If sender mobile host itself saves two consecutive checkpoints then it will delete its last checkpoint and logs of that checkpoint interval. Before permanently deleting logs of any checkpoint interval, it will replay all its logged messages with a tag ‘these logs will be deleted shortly’. The mobile hosts that have already saved checkpoints of any of these replayed logs, will discard replayed logs. Otherwise the receiver mobile host will save these replayed logs in its current MSS. If any of the receiver mobile hosts is not connected to the network, the mobile support station in which the MH was connected before disconnection will save or discard logs as mentioned above on behalf of the MH. Log based recovery ensures consistent recovery without coordination overheads.

• **Secure checkpointing:**

To prevent information leakage due to confidentiality attack and modification of checkpoints due to integrity attack, security of checkpoint is important. Modification of checkpoint leads to unsuccessful recovery of failed mobile host. Without security measures, recovery probability of a failed mobile host will be much less in security attack prone mobile hosts. Thus, efficiency of checkpointing and rollback recovery algorithm will degrade without secure checkpoint. This issue has not been addressed in any existing checkpointing and rollback recovery algorithm except in [10]. Instead of using symmetric key cryptography as in [10], low overhead public key cryptography based secure checkpointing and recovery algorithm is proposed.
• **Trusted checkpointing:**

  Security attack to a checkpoint can happen when it is saved in mobile host’s memory or in stable storage. This can be prevented by encrypting the checkpoint. But this adds encryption and decryption cost. This additional overhead must be avoided in resource constrained mobile computing system like mobile ad hoc network. If by any means it can be ensured that there will be no security attack in the mobile host’s memory then the checkpoint can be saved in plain text. This can be done by ensuring the mobile host as a trusted entity. Trustworthiness of a mobile host is defined by its failure rate, available resources, rate of security attack, recommendation from other mobile hosts etc. Based on this evaluation, if a mobile host is trusted, then it is ensured that it can save or transfer checkpoint probably without any chance of failure or security attack.

### 1.5 Organization of the Thesis

The thesis is organized as follows. Chapter 2 presents the related works. This chapter presents a thorough study and investigation of existing works that address different issues like characteristics, challenges, limitations etc. in mobile computing. Chapter 3 describes a fault tolerant mobile computing environment in which failed applications (running on mobile hosts) recover consistently from latest checkpoint. This chapter also sets some assessment parameters by which efficiency of both checkpointing and recovery algorithms can be measured. All these algorithms consider mobility issues and activities of mobile hosts as decision parameters while saving checkpoints. Probable security attacks in wireless links and mobile hosts are continuous threats to information leakage of data and successful recovery. Two algorithms presented in Chapter 4 describe secure checkpointing using low overhead public key cryptography. In first algorithm, checkpoint is encrypted and sent through wireless channel or saved in mobile host memory. In addition to this, second algorithm illustrates that rate of failure of processes increases with time if checkpoints are not encrypted. Thus there may be integrity attack on data. So, at the cost of cryptographic overheads checkpoints can be made secure. Mobile computing has extended its domain to mobile ad hoc network from infrastructure based wireless cellular network. The Mobile Adhoc Network (MANET) considered is...
cluster based hierarchical network. Chapter 5 describes mobility based checkpointing and trust based rollback recovery algorithms. Secure checkpointing using cryptography is not practically feasible in MANET because of its different limitations. Two trust models are proposed to evaluate trustworthiness of mobile hosts. If a mobile host is evaluated to be trusted then the checkpoints are secure while saved on or transferred through them. Trust based recovery ensures secure routing of checkpoints through trusted mobile hosts only. In Chapter 6, a secure checkpointing-recovery algorithm using trusted mobile hosts is described. Chapter 7 concludes the present work and mentions limitations and scope of further extensions.