CHAPTER 1

Introduction

1.1 Introduction

Software and software systems have become indispensable tools used in most of the business organizations, educational institutions and in our personal lives as well. However, the software systems always fear danger or risks from the malicious elements due to industrial growth and rapid evolution of technology. The growing Internet connectivity has made the sensitive information more vulnerable to unintentional and unauthorized use. It has indicated security as one of the major issues to be dealt with.

Security may be defined in a number of ways that focuses on protection of the software system, data or information from unauthorized usage. According to the dictionary of computing, security is the prevention of, or protection against access to information by unauthorized recipients, and intentional but unauthorized destruction or alteration of that information [DoC96]. Alternately, security can be described as the ability of a system to protect information and system resources such as CPUs, disks, programs, information, data etc. with respect to confidentiality and integrity. It may also be stated as “ensuring that data stored in a computer and transmitted between computers cannot be read or compromised by unauthorized users” [GS05]. Security can be explained as “the mechanisms that protect computer-based equipment, information, data and services from unintended or unauthorized access, change or destruction”. It can be expressed as “the protection of data from accidental or intentional disclosure to unauthorized persons and from unauthorized modification” [IBM68]. According to ISO 9126, security is defined as “attributes of software that bear on its ability to prevent unauthorized access, whether accidental or deliberate, to programs or data” [LINK01]. From the above definitions, it is evident that security mainly deals with the protection of assets including communication networks, computers,
programs etc. It also includes protection of systems from disasters, mistakes and manipulation so that the likelihood of security incidences are minimized [BOR].

Security can be considered as an issue of overall control, which involves the development of design and methods to ensure security and enforce privacy decisions [HIR71, CMM72]. The other terms that are closely related to security are namely; reliability and privacy. As defined by ANSI, software reliability is the probability of failure-free software operation for a specified period of time in a specified environment. IEEE states reliability as “The ability of a system or component to perform its required functions under stated conditions for a specified period of time.” The number of defects found and defects fixed are used to measure reliability [GSE90]. Privacy relates to personal, confidential, or personally identifiable information leaked to others. It is normally considered as a legal issue to be handled by organization’s legal council while security is considered to be technical issue taken care by the security officers [HER02]. Thus, privacy can be interpreted as a way of complying with policy and security as a way of enforcing policy [HL06].

Malicious intruders obtain unauthorized access to systems thereby taking advantage of the defects of software systems. To protect the software systems, the organizations use firewalls and proxy servers to restrict sites, to avoid unnecessary downloads, and protect the systems from viruses using anti-virus software. Software systems can also be protected through proper authentication by use of logins, passwords, biometric identification etc. and by providing authorization through access rights. In today’s scenario, such methods are not effective as the hackers intend to exploit the unknown vulnerabilities.

The softwares may be easy target because of virtually guaranteed presence of vulnerabilities due to lack of software security properties. The vulnerabilities when exploited intentionally/unintentionally by malicious/non-malicious user result into security breaches. These are the result of non-patched software, insecure configurations and insecure design and code [GOE09]. Unfortunately, it
has been observed that common development practices do not consider security, leaving end-product vulnerable [D+04].

A software system must fulfill security issues such as confidentiality, integrity and availability (CIA). These issues may further be comprised of many other security attributes such as authenticity, accountability, non-repudiation, and access control [HOR04]. Confidentiality refers to the disclosure of information only to authorized users. It can be achieved by encrypting the data during transmission as well as at rest. Integrity allows only the authorized users to modify the data and other resources. It can be achieved through data integrity and origin integrity (authentication) [BIS02]. Data integrity implies that the data should not be changed accidentally or maliciously without the permission of the owner. Availability indicates that the data and resources must be available to authorized users when needed. It is affected by many technical issues such as malfunctioning of the hardware, and natural or deliberate human intentions etc. Denying access to information is one of the most common forms of attack [CHI12]. In non-repudiation, the sender of data is provided with proof of delivery and the recipient is provided with sender’s identity so that neither can deny processing of data. The user can be authenticated by providing a proof of individual’s identity using password, card, biometric identity, public and private keys etc. The access control grants rights to an authenticated user regarding the operations to be performed on file or data. Accountability of a system allows tracing the actions being performed on a system to specific user entity so that malicious user or attacker may be traced properly.

Earlier software engineering focused only on quality aspect such as maintainability, reliability, flexibility, testability, portability, reusability etc. that did not consider security [MRW77, B+78]. Later, with increasing security incidences, security has been included as one of the quality aspect of functionality as in FURPS model and ISO 9127 standardized quality model [GC87, LINK01]. Security has been signified as a Non Functional Requirement (NFR) since it defines constraints on the behavior of the software system. Hence,
it has not been considered as an integral part of secure analysis and design [CN95, CIN03, MOU04].

Implementing security is analogous to imposing countermeasures to reduce risks in the software and data. Security can be implemented at data level, system level, and network level by restricting the user’s access to the system resources. However, the main reason for security breaches is lack of secured software that fails the external security mechanisms such as firewall, anti-virus etc. The software can face security threats during its development, deployment, operation (by inside and outside users depending on the software), and sustainment [GOE09]. The development team may introduce vulnerability in the software intentionally or unintentionally while incorrect software configuration during deployment may also lead to security breaches. During operations, when the flaws in software are discovered and publicized, the patches may be applied and new versions may be released. The networked system may also become vulnerable during its operations. At the same time, the software may become vulnerable when the discovered vulnerabilities are not addressed properly during the release of new version.

Development of secured software is mostly concerned with guidelines, best practices and a number of ad hoc tools available to address security. A number of secured development processes and tools are available in the literature and industry such as misuse cases, attack trees, threat modeling, security testing etc. These can address security at technological context when considered during development process and can ensure security in the software product. Microsoft offers well defined Secure Development Lifecycle (SDL) describing twelve stages for secure development [HL06]. These stages include education and awareness, project inception, define and follow design best practices, product risk assessment, risk analysis, creating security documents, tools and best practices for the customers, secure coding policies, secure testing policies, security review, security response planning, product release, and security response execution. Figure 1.1 illustrates the reduction in vulnerabilities using SDL in
Fig. 1.1: Reduction in Vulnerabilities using SDL in Microsoft Products

(Source: Microsoft Security Blog)
Microsoft products in Windows and SQL Server.

Secured development of a software normally implies a reliable, bug free, stable and vulnerability free software [GC04]. It focuses on incorporating security during pre-development, development and post-development activities [MCG04]. Mostly, organizations devote time to patch security as a post-development issue, whereas security is an emergent property of software system [SHB13]. Hence, security concerns must be global in nature and should be applied at possible stages to build up secured software. Some security domains may include hardware security, system and application software security, network security, etc.

The development processes such as waterfall model, incremental model, spiral model etc. do not consider security as part of development process [SHB13]. It is important for an organization to include security explicitly during Software Development Process (SDP). It is helpful to use security frameworks that guide development process against common model and evaluate security activities being carried out [DAV05]. It has been observed that there exist some security frameworks to combat security in the software development process [MC08, BUE09, STE06].

Insecure software accounts to design related flaws, insecure coding and insecure configuration [MOR08]. It owes to lack of knowledge about security among the software developers. Even with the knowledge of application risks and vulnerabilities, the developers may not have idea to deal with these problems [RM09]. Understanding security aspects is a prerequisite for secure development and is evident in SDL that deals with security training as step 0 [HL06]. Gathering security requirements focuses on security risks and it needs extensive involvement of the stakeholders [FCA05]. Secure designing may involve the use of threat modeling, security patterns, etc. to identify as well as to validate the design related security threats [HL06, MJ08]. Moreover, none of the developing platforms can secure the software. Hence, deep knowledge about the insecure code and functions is mandatory for the developers to ensure the security of software [HLV05].
Security metrics and measurements are powerful techniques to SDP [CK09]. The metrics focus on identifying trust worthiness of security considerations during the development process. Several metric models and security metrics are available to deal with security concerns [NP07, CIS10, NEL10, SEH08]. Most of the product metrics may be validated while process development stage metrics are still in the nascent phase.

Thus, the major contribution towards secured software may be achieved by incorporating security during the software development stages. It requires a thorough understanding of various types of security as well as its technological and management aspect. Systematic consideration of security can be supported by security framework to help the development of secured software product. Moreover, security requirements must be gathered as the first step towards secure development. Security must be incorporated systematically during the software development stages to ensure secured product. Additionally, secured development process must be controlled by help of security metrics.

1.2 State of Art

With the increasing number of security breaches, the interest of researchers and practitioners moved towards the need for security. Till date, security has been considered to be an add-on during post development phase. Security has been observed as an important aspect of technology. It is evident that the computer users suffer with varying threats such as virus, malware, financial and personal information loss, etc. [HRS07]. As an effect, the usage of external security mechanisms such as anti-viruses, firewalls, malware protection mechanisms etc. has been proven to be of extremely importance. The statistics pertaining to sale of anti-virus software shows the expected growth of 6.8% per year. On the other hand, report released by Norton (Symantic) highlights 42 million cyber crimes every year that owes to losses of 8 billion USD in our country [LINK05]. According to McCafe, lower rate of security measure adoption
and if attacked, the downtime due to major incident may lead to loss of Rs.30 crore a day [LINK06].

Now a days, cyber laws have been well established and enforced for IT security also. Privacy and information security laws provide provision for definition of sensitive personal data, privacy policies, restrictions on data collection and processing etc. [LINK07]. Similarly, IT Act 2000 and 2008 highlight the basic legal framework for electronic transactions. The laws provide fundamental data security solutions that are based on legal framework. Legal frameworks provide the way to secure data and information. But it does not address the root cause of insecurity i.e. the software itself.

The concept of secured software development was first introduced in 1991 by US National Academics of Computer Science and Telecommunications Board. Security is fundamental to the matter of prevention of breaches and hence must be considered during software development process. Some of the major secured development processes from practitioners include SDL from Microsoft, Touchpoint from Cigital and Comprehensive, Lightweight Application Security Process (CLASP) from Open Web Application Security Project (OWASP) [HL06, MCG06, LINK08]. These development processes have gained popularity in IT industry with some limitations. SDL is more suitable for heavyweight applications and is normally suitable for large organizations while CLASP is a lightweight process affordable for small organizations with less security demands. Touchpoint is based on industrial experience and is considered to be more practical [B+07]. Some other secure development processes are namely; SAFEcode, Coverity, Foundstone etc. [SIM11, COV12, LINK09]. There exist secured development model based on XP that focuses on the iterative development of secured software [DAU10]. Security can be well thought as a part of requirements gathering stage by considering the list for security requirements that may include choice of platform, need of training to gather security requirements, possibility of testability and validity etc. The development process can reduce vulnerabilities when these requirements can be included [PK13].
Many organizations focus on security considerations as an important aspect of software development. National Institute of Standards and Technology (NIST) publishes various standards in the field of information security such as computer security, information security, network security etc. CERT program ensures the use of appropriate security management practices to secure networked systems to limit the damage caused by attacks, accidents or failures. ISO publishes standards, technical reports, and related information for security implementation such as information security management, information security risk management etc.

OWASP mentions top 10 security threats based on data concerning risks from 8 organizations and 500,000 vulnerabilities. *OWASP Top 10 2013* reports indicated injection, broken authentication and session management, Cross Site Scripting (XSS), insecure direct object references, security mis-configurations etc. as some of the major security threats in web applications. Similarly, Common Weakness Enumeration (CWE) states the measurable set of security weaknesses. It helps in common discussions, security requirements, selection and use of appropriate security tools that can help avoid/ search these vulnerabilities. It aims at finding and dealing with causes of software security vulnerabilities [LINK10]. Weaknesses by CWE are also used to classify vulnerabilities to support secured software design [RM12].

Some of the popular tools for secured software development include misuse cases, threat modeling, attack trees, attack patterns, source code analyzers, etc. Misuse cases are the business threat modeling tool that describes the process of malicious act against software system. Threat modeling helps to apply structured approach to identify security threats, vulnerabilities and countermeasures [HL06]. Attack trees are conceptual diagrams to identify the attack methodology. It is an effective tool for threat analysis and risk assessment. Such tools help in identification of security requirements as well as design related security flaws.
1.3 Objectives

On the basis of extensive literature review on security aspects and secured software development processes, we designed objectives with some identified challenges. These challenges include understanding the need of security and its integration during the development process. To combat these challenges, we have designed framework, secured software development process and metrics and have been assessed using case studies.

Our first objective is affirmed to identify various security domains for software system, classify these domains as a part of pre-development phase and use the classification for development of different types of software. Classification involves the understanding of various security measures required for secured software systems. At first level, security is classified as hardware security, logical security and security management which is further classified in five more levels. Thus, varied security domains such as hardware security, application software security, system software security, single user system security, peer-to-peer network security, server-based system security, server-based platform security, server-based communication security, web-server security, web communications security, logical data security, physical data security and security management have been identified. Further, we recognized some of the software systems such as static website, dynamic website, web-based enterprise application, intranet and extranet based systems, client/ server based systems, e-commerce, kiosk, cloud, and desktop based systems to identify the security domains involved to secure such systems. Security can be observed as an issue of overall control with software security as one of the major aspects for secured systems. Even with many efforts on security management and network security, security has emerged as a problem of software security.

Our next objective is stated to identify the dominance of security in networked environment. It is evident that there exist many networks such as Internet, client/ server based, centralized system etc. with varied security issues. To understand the security issues, we proposed an umbrella of networks based on
various dimensions such as size, design, network architecture, organizational scope, computing models and topologies. The network dimensions may be further classified along with the security concerns. Additionally, we tried to identify the dominance of security on the stages of software development process based on varied types of networks. While most of the security breaches are result of insecure software, practitioners still rely more on security configurations and perimeter defense.

The subsequent objective is avowed to identify the technical and management security aspects to secured software development process. Secured software development is a means to software security. It requires systematic security consideration throughout SDP. We developed techno-management view of security that focus on technical and process management aspects of security to be incorporated during Secured Software Development Process (SSDP). The technono-management view of security may help in bridging the gap between process and the management aspects of security. We also introduce generalized Software Product Security (SPS) framework for systematic inclusion of security aspects. The framework is a three layered structure consisting of control, security aspects and development layers. These layers may help in systematic inclusion of security during the product development. Further, a mathematical model is formulated to estimate the security concern of some software system developed using SPS framework through Security Factor ($F_s$).

Later objective is established to design secured software development process. Secured development initiates with gathering specific security requirements. We established Software Security Requirements Gathering Instrument (SSRGI) that promotes gathering security requirements in detail with the help of the stakeholders. SSRGI focuses on gathering Secure Functional Requirements (SFR), Drivers, Functional Security Requirements (FSR), Non-Functional Security Requirements (NFSR), Secure Development Requirements (SDR), and Security Testing Requirements (STR). When SSRGI is integrated with Software Requirements Specification (SRS) document, it may promote the systematic gathering of security requirements along with the functional
requirements. Since software developers might unknowingly inject flaws during software development process, SSDP has been designed to address security issues during each development phase. It may support the development team in security considerations throughout the development stages. It shall ensure that the updates and patches do not add security weaknesses during SDP.

Our last objective is concerned with identification of security metrics for each of the software development stages. Metrics serve as a basis for analyzing security improvements by investigating data regarding measurements and metrics such as number of security requirements gathered, number of design flaws related to security, mean time between security incidents etc. throughout the development. It also analyzes product implementation by measuring the security parameters such as failure in audit capturing mechanisms, number of known vulnerability incidences etc. for improving performance and accountability. In this respect, we analyzed currently available metrics for software development stages. It has been observed that late security assessment can not help in correcting the security issues of early stages. Moreover, the process metrics do not provide sufficient details to analyze security. We have developed security metrics on the basis of security issues of the development phases. Further, a methodology to identify the effectiveness factors to analyze and judge the security efforts have been built. The metrics may support the development team in evaluating and monitoring the security efforts throughout the development process.

1.4 Thesis Outline

The thesis is organized in 8 chapters to cover the research issues while integrating security in software development process. The research issues include understanding the various security related dimensions, security requirements gathering, integrating security in development process and security estimation through metrics. A general overview of said research field and the objectives of the research are covered in each chapter.
In Chapter 2, detailed literature review has been provided that focus on varied research going on in the field of securing the software systems. The research domains include security taxonomy, security models, security aspects for securing the software systems, network and communication security, security standards and compliances etc. for securing information. Other research areas include formal approaches to security, security technologies for secure design and implementation, secured software development process, security metrics etc.

Hierarchical classification of security is presented in Chapter 3. At Level 1, security is classified majorly as hardware security, logical security and security management. Additionally, in the chapter, we identified various security domains in next five levels in classification. Later, a range of software systems such as static and dynamic websites, web-based application, Internet, Intranet, Extranet, client/server, e-commerce, kiosk, cloud and desktop based system have been analyzed for security issues under the security domains. Lastly, the benefits of classification to management, development team and administrators during development of a software have been delineated.

In Chapter 4, we proceed with the identification of various network dimensions that are further classified along with the security issues. Dimensions identified for classification are size, design, network architecture, organizational scope, computing models and topologies. Further, we proceed with the discussion of security on software development phases in developing networked systems. Later, in the chapter, we discuss the dominance of security on software development phases while developing varied network systems such as client/server, Internet, Intranet, etc. based systems. It may be concluded that the security dominance may help in identification of security needs during SDP rather than depending on defense perimeter and secured configurations.

The concepts for SSDP are covered in Chapter 5. In this regard, we identified technological and process management aspects of security. We tried to establish links between the various aspects on the basis of feasibility of actions. The techno-management view is an effort to bridge the gap between developers
and managers. Next, generalized Software Product Security (SPS) framework is discussed with layers as control, security aspects layer and development layer. The framework is developed for systematic inclusion of security during software development. Further, a mathematical model to identify Security Factor ($F_s$) of the software product has been developed using SPS framework and $F_s$ is elaborated with the help of a case study.

In Chapter 6, the need for gathering security requirements is presented for the various kinds of stakeholders such as customers, managers, designers, coders and testers. The types of requirements include SFR, Drivers, FSR, NFSR, SDR, and STR. The generalized Software Security Requirements Gathering Instrument (SSRGI) is developed for systematic security gathering. It is also integrated in SRS document to help in gathering security requirements along with the functional requirements. The chapter also elaborates on security issues of the SDP stages, proposed actions for incorporating security and the tools to analyze security issues. Benefits of SSRGI and secured development process are highlighted with the help of case studies at the end of the chapter.

We depict the need of security metrics in Chapter 7 based on the review of varied security metrics available in the literature for secured development process and the software product. It is evident that the widely accepted metrics have not been covered yet. In this chapter, we discuss security metrics that may be used at different SDP stages on the basis of its security issues regarding security requirements, secured design, coding, testing etc. Later, the effectiveness factors to identity the security efforts of the development team during SDP stages have been illustrated by analyzing some software projects. Finally, we conducted a study for different project cases and established the fact that the security metrics and effectiveness factors may help to judge the secured development efforts.

In Chapter 8, we conclude by the remarks on the contributions made in the field of software security especially while integrating security during the software development process.