SECURITY ANALYSIS AND IMPROVEMENT MODEL FOR WEB-BASED APPLICATIONS

A Dissertation

by

YONG WANG

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

December 2008

Major Subject: Computer Science

SECURITY ANALYSIS AND IMPROVEMENT MODEL FOR WEB-BASED APPLICATIONS

A Dissertation

by

YONG WANG

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

Approved by:

Co-Chairs of Committee,	William M. Lively
	Dick B. Simmons
Committee Members,	Yoonsuck Choe
	Suojin Wang
Head of Department,	Valerie Taylor

December 2008

Major Subject: Computer Science

ABSTRACT

Security Analysis and Improvement Model for Web-based Applications.

(December 2008)

Yong Wang, B.S.; M.S., Anhui Agricultural University, China;

M.S., Texas A&M University

Co-Chairs of Advisory Committee: Dr. William M. Lively
Dr. Dick B. Simmons

Today the web has become a major conduit for information. As the World Wide Web's popularity continues to increase, information security on the web has become an increasing concern. Web information security is related to availability, confidentiality, and data integrity. According to the reports from http://www.securityfocus.com in May 2006, operating systems account for 9% vulnerability, web-based software systems account for 61% vulnerability, and other applications account for 30% vulnerability.

In this dissertation, I present a security analysis model using the Markov Process Model. Risk analysis is conducted using fuzzy logic method and information entropy theory. In a web-based application system, security risk is most related to the current states in software systems and hardware systems, and independent of web application system states in the past. Therefore, the web-based applications can be approximately modeled by the Markov Process Model. The web-based applications can be conceptually expressed in the discrete states of (web_client_good; web_server_good, web_server_vulnerable, web_server_attacked, web_server_security_failed;

database_server_good, database_server_vulnerable, database_server_attacked, database_server_security_failed) as state space in the Markov Chain. The vulnerable behavior and system response in the web-based applications are analyzed in this dissertation. The analyses focus on functional availability-related aspects: the probability of reaching a particular security failed state and the mean time to the security failure of a system. Vulnerability risk index is classified in three levels as an indicator of the level of security (low level, high level, and failed level). An illustrative application example is provided. As the second objective of this dissertation, I propose a security improvement model for the web-based applications using the GeoIP services in the formal methods. In the security improvement model, web access is authenticated in role-based access control using user logins, remote IP addresses, and physical locations as subject credentials to combine with the requested objects and privilege modes. Access control algorithms are developed for subjects, objects, and access privileges. A secure implementation architecture is presented. In summary, the dissertation has developed security analysis and improvement model for the web-based application. Future work will address Markov Process Model validation when security data collection becomes easy. Security improvement model will be evaluated in performance aspect.

DEDICATION

To my parents, sisters, brothers;

my wife;

and my children

ACKNOWLEDGEMENTS

I want to express my sincere appreciation to my co-advisor, Dr. William Lively, who believed in me. I thank him for his invaluable guidance during my studies. His guidance and encouragement helped me to finish my studies and obtain a Ph.D. degree in Computer Science at Texas A&M University. I have learned so much by conducting and presenting my research. I also want to give thanks to my other co-advisor, Dr. Dick Simmons, for his advice regarding my research. His comments have helped to greatly improve my dissertation.

Special thanks go to Dr. Yoonsuck Choe for his discussions and insight about modeling in my dissertation and to Dr. Suojin Wang for his advice on statistical methods. I would like to thank Dr. Richard Feldman for being a substitute committee member for my preliminary exam and advising me in Markov Process Model.

I also extend my thanks to Mr. Jeremiah Grossman, CTO and founder of Whitehat Security; Mr. Ryan Barnett, Chief Security Officer at EDS; and Dr. Anton Chuvakin, Director of Product Management at LogLogic, for providing their time and help.

I wish to thank my parents, brothers, sisters, and many friends in Aggieland for their continued support and encouragement. To my wife, Weihong, my sons, Nicky and Phillip, and my daughter, Janelle and Evelyn, I express my sincere thanks for helping me to reach my goals by continued support and sacrifice over the years.

TABLE OF CONTENTS

Page

CHAPTER Page

LIST OF TABLES

LIST OF FIGURES

Page

CHAPTER I

INTRODUCTION

1.1 Scope of the Work

In the mid 1990s, the Web appeared to be a new medium for information dispersal. Universities use the Web to post university news. Faculties use the Web to deliver their lecture materials and homework assignments. Communities use the Web to introduce their community initiatives, development, safety, and rules [96]. Commercial companies use the Web to advertise their products, provide technical support, and sell their products. Soon, banks, hospitals, real-estates, and other areas were developing their web-based applications. The web-based applications have open access characteristics through the Internet connections.

Today, the Web has been a major channel through which information is delivered. As the World Wide Web becomes ubiquitous, web software quality and information security receives more attention. Web quality is related to user satisfaction, while web information security is related to privacy, confidentiality, and data integrity. The web adapts the basic client/server model using HTTP (Hyper-text Transport Protocol) and TCP (Transmission Control Protocol) protocols to request and deliver information across computer networks [14].

For software quality, different failure prediction models have been proposed [88].

This dissertation comforms to the style and format of *IEEE Transactions on Dependable and Secure Computing*.

The models include experienced models and regression models [15], [18], [100]. By 2001, more than 75 reliability growth models had been developed to predict software failures [5]. To some degree, software failure prediction models are already well studied.

In a web application system, the typical client-server communication is adopted in a distributed manner, but in distributed software systems, the systems determine the security measurements. Thus, the information technology has changed its approach from a network-security to a software application-oriented security [4], [23]. Once the application port is open to the outside, the firewall for traffic control loses its function. Based on the website http://www.securityfocus.com in May 2006, operating systems, web-based applications, and others contribute 9%, 61% and 30% vulnerability respectively.

For the web-based applications, two key aspects control web system security. The first method is access control. Through access to the Internet, attackers can post a high potential threat to the system. The second approach is the secure exchange of information. Access control preserves web information for proper usage. Secure exchange of information can keep information as original and confidential [42], [96]. Electronic entrance examination can play an important role in access control.

Dowd et al. (2007) assigned software vulnerabilities into three categories [27], [97]: design vulnerabilities, implementation vulnerabilities, and operational vulnerabilities. The design vulnerability primarily comes from an aspect of software design. Implementation vulnerability comes from software code phase. Operational vulnerability originates from the operational procedures and configurations of software in specific environments.

According to Shah (2006) [84], the web vulnerabilities primarily come from program source codes. Previous investigation indicates that the vulnerability from source code errors accounts for 64%, while configurations account for 36% of the vulnerabilities. Shah further reported that, according to findings in IBM research labs, every 1500 lines of web program carry more than one vulnerability.

Many trials have attempted to reduce vulnerabilities. A source code review can reduce the application vulnerability. During the code review, both external and internal reviews need to be conducted. Several security issues need to be addressed during the review: dependency, entry point exploration, vulnerability detection, and mitigations.

Dependency: Before an examination, we are required to know the structures of the programs. The goal of this phase is to analyze the programming dependencies. In general, the web-based applications have a database server in the backend (MS SQL, Oracle, or MySQL). When conducting a code review, we need to check the system interface. We also examine the computational platform and the type of web server in the software system. This can help us to establish a secure system. Analyses of the web-based application systems also help us to identify design patterns in the program. During the code reviews, we also need to check any integrated components from another party.

Entry point exploration: The entry points give input to applications. These inputs are delivered to the database server, web server, and other components. When these parameters are not correct, they create possible system vulnerabilities. The entry points include HTTP variables, SOAP (simple object access protocol) messages, XML files, and mail systems. HTTP variables from the web browser and end users are delivered to applications. The entry points may appear to be forms and retrieval parameters. SOAP

messages are also one kind of the important entry points in the web application systems. XML files used in the web-based applications may come from other resources. The web-based application systems may receive mails carrying vulnerable entry points in the programs.

Vulnerability detection: Detecting vulnerable entry points makes the threat analysis easy from its corresponding vulnerability. From the entry point, we can track the parameters' path in the program and how the results are attributed to the parameters. From the vulnerability detection process, we also can assess the severity of the vulnerability.

Mitigation: When having detected security weak points in a program, we can reduce the threat impact from the vulnerability. Thus, we can mitigate the frequency of vulnerability occurrence and the severity of potential damage from the vulnerability. For instance, we can add a rule to remove SQL server input with quote markers. In this way, we can reduce the chance of attacks in SQL injection.

To measure information security and define information security systems, several methods have been proposed. One of the popular methods in information security analysis is Markov Process Model. The Markov Process Model has been applied in process decisions in various domains. It has been used to analyze computer intrusion and detection data. Formal methods have been used to specify security-critical software systems [50]. Formal methods appear to be effective in the security system specifications. To effectively specify information security systems, we need to incorporate new technology into the application domain.

As the Internet technology keeps progressing, computer access can be easily traced to their physical locations using the GeoIP services [64]. The GeoIP services easily map the Internet IP address and its corresponding geographical location. The GeoIP services have been used in identifying attacker locations in network forensics [19], [24]. The GeoIP services can be used as a new way to authenticate a remote access computer in addition to user login.

In this dissertation, I plan to present a security analysis model using the Markov Process Model. Risk analysis for vulnerabilities is conducted using a fuzzy logic approach and information entropy theory. The Markov Process Model has "memoryless" characteristics. In particular, the Markov Process Model has the property that the future does not rely on the past if we know the current [26], [80]. In a web-based application system, security is most related to the current software systems and hardware systems. Security is independent of the web-based application system states in the past if we know the current system states; in other words, the system security is directly decided by current software systems and hardware systems.

Both system security and system risk for web software systems are the focus of this dissertation. For instance, if a computer virus gets into the web-based application systems, the virus must find some vulnerable points in the current software, hardware systems, or network configurations to spread it. Thus, current system state has a major impact on the system security in the future. System security has a kind of "memoryless" nature which the Markov Process Model owns. On the other hand, previous researchers have also validated that web access traffic has the Markov Chain Model property and can be modeled accordingly [51], [101]. Since web security is highly correlated with web

access traffic, security parameter in the systems appears to have a similar pattern to access traffic over time. The more access to web applications, the more security risks to the web-based application systems. Therefore, security risk in the web-based applications can be approximately modeled by the Markov Process Model.

Maden et. al. (2002) developed a generic state transition model for intrusion tolerant systems in the Markov Chain Model [55]. I will extend the generic state model to the web-based applications. The extended model can describe multiple subsystems and the interactions among the subsystems in the web-based applications. In this dissertation, security risk in the web-based applications can be expressed in the states of (web_client_good, web_client_vulnerable, web_client_attacked, web_client_security_failed; web_server_good, web_server_vulnerable, web_server_attacked, web_server_security_failed; database_server_good, database_server_vulnerable, database_server_attacked, and database_server_security_failed) as possible state space in the Markov Chain Model. The vulnerable behavior and system response will be analyzed, and the analyses will focus on availability-related aspects: the probability of the modeled system moving to a particular security failed state and the mean time to a security failure of a system (when we do these analyses, we need to set security failure state as an absorbing state). Vulnerability risk index will be classified in three levels as an indicator of the level of security (low risk level, high risk level, and failed risk level). Empirical application examples in security analysis will be provided using web attack data from http://www.honeynet.org/scans; vulnerability risk analysis will be conducted using the vulnerability report data from https://cirdb.cerias.purdue.edu/coopvdb/public.

As the second objective of this dissertation, I plan to propose a security improvement model for the web-based applications using the GeoIP services in the formal methods. In the security improvement model, web access will be authorized in role-based access control using user login, remote IP address, and physical location as subject credentials. If the subject credential is satisfied, a further authentication check is conducted using the requested objects and privilege modes. Access control algorithms are developed late in the dissertation. A secure implementation architecture will be presented. Ren and Jin (2005) used IP address and its location to combine with TTL, MAC, and TCP/IP stack fingerprint to prevent IP spoofing attacks in network security [77]. American Express uses the GeoIP services for online fraud detection for credit card access [65]. More recently, British Columbia Lottery Corporation chose the GeoIP services for territory-based web access control. My approach is to develop a general model for the web-based application access using flexible role-based access control and the GeoIP services. This dissertation is intended to contribute to the literatures regarding these popularly discussed topics: the security analysis, risk analysis, and security improvement models for the web-based applications.

1.2 Organization of the Dissertation

The rest of the dissertation is organized by topic. In Chapter II, a literature review for security analysis and improvement is presented. This chapter summarizes the work done on security analysis, risk analysis for the web-based software systems, and security improvement model. Chapter III overviews the research and describes the security analysis, risk analysis, and security improvement model for the web-based applications.

Chapter IV describes the major methods used in the security analysis, risk analysis, and security improvement models. Chapter V analyzes the security analysis and risk assessment for the web-based applications. A case study example is provided. Chapter VI presents a security improvement model. Chapter VII concludes the dissertation by summarizing the research and its results. Cited references, two appendices, and a letter of permissions follow the conclusion.

CHAPTER II

SURVEY OF RELATED WORK

Computer security has become an important research topic in computer science. In this research area, computer security has developed its own assumptions and languages, attracting many researchers from a variety of disciplines. [67]. Previous work in computer security concentrated on complex protocols, whereas recent research in security is extended to a system-level. This research studies how secure systems are designed in response to possible attacks, and current work focuses on intrusion tolerance so that designed systems can continually implement the designated functions when encountering successful attacks. Previous popular technologies in the Internet in gopher, FTP, and telnet are out of date and have been replaced by web technologies such as RSS, Ajax, and Soap. The RSS is a XML-based word to describe news and other web contents for publications. It describes one of three standards: RDF (resource description framework) Site Summary, Rich Site Summary, and Really Simple Syndication [4], [92]. AJAX is an Asynchronous JavaScript and XML tool to build different applications for the Web to handle user requests. SOAP stands for Simple Object Access Protocol, a protocol that enables a program running in one operating system (i.e. windows) to talk with other programs running in the same or different environments (i.e. Linux) using HTTP and XML. As concurrently shifting in technology, the most attacks on computer networks ten years ago are now targeting the different applications and software systems on the computer. Software application-oriented security has become the focus of the research previously interested in network-based security [4], [23], [97].

A system-level model does not exist to measure security by a specific approach. Most trials are attempting to develop a security system process. To validate system security, the formal methods and red team approach are widely applied to different application systems [48], [53]. Both are very helpful for detecting and classifying system vulnerabilities. The main limitation may appear when applied to a large system [67].

Recently, many researchers try to describe the system security numerically. Modeling is a good tool applied in evaluating systems that are either secure or not. Computer system failures from attacks display some similar characteristics to random system failures. Therefore, the evaluation method in system dependability can be applied to the system security. In general dependability analysis, failures are assumed to come from random events in the hardware or software. In security analysis, security failures depend on the system state and the time in which attackers are on the system. These security failure processes appear to be in a stochastic process [97].

Reliability describes a system in service rate during a specified time period. Reliability is determined by the frequency and severity of faults in computer systems. For security measurement, using probability of attacks can help impact analysis on system reliability [39], [87]. Developing and validating stochastic models in cyber attacks is still an open question in the research arena of computer science [68].

Availability can be defined as the portion of time that a system can conduct its intended function in the routine operation. When we analyze the impact of security on availability, system availability in the web-based applications after cyber attacks can be changed in different ways. It can be changed by an attack behavior on the system and by the time needed for security analysts to investigate an attack. It can be changed by the

effort to bring the system back to a good state after an attack. Availability in security specifically means that a system can provide its intended service after a successful attack [97].

Safety indicates that a system can survive after serious damage. Since system safety relies on the impact of a system failure, we can define system safety similar to the dependability after attacks. When conducting security analysis, we can count the sensitive data as a part of safety. For instance, sensitive data open to the public can lead to severe damage for a security-sensitive system. The release of sensitive data will cause the severe safety concerns for a reliable system [68].

Performability describes system characteristics with an existing failure. In the analysis, a group of states for a system are specified. Each state matches to a system configuration, thus representing a specific system behavior. Each state also describes how the analyzed system transits. Performance is associated with the costs and rewards in the system design [16], [59], [68]. One of the popular measurements for the performability is using mathematical model approaches to analyze the system behavior.

Security covers the attributes we described before. Security also includes data integrity and confidentiality. Data integrity means that systems prevent the data from being revised in unauthorized ways while data confidentiality means that data are only accessed by the authorized users. Models for security analysis may include how and when the system security fails, the security failure consequences on the existing systems, expenses of system recovery, and costs of improving system defense [68].

As e-commerce becomes pervasive, security and privacy play more important roles than ever before. Computer security has spread beyond the technical arena; it affects

our daily life. Computer security appears to be a popular topic. The discussed topics mostly cover basic concepts: why we need a firewall to improve the security, what the encryption can do, and which security product is more useful [95]. Lake (2000) addressed the security problems [47]. He reported that there were about 20 new vulnerabilities reported every week from early 1998 to late 2000. Some vulnerabilities were found in open source software systems, while others were found in home-grown programs. Further analysis indicated the vulnerabilities in both Unix and Windows programs are well distributed in the collected data.

The results of security failures are various. Security failures can lead to all the system being unavailable. Attack on a service system is a direct cause for a security failure. Attacks may come from either remote or local machines to their targets. In a remote attack, an attacker can get into a computer that is on the network, making good use of software flaws. When the software is authorized through a firewall in an open port, the firewall becomes functionless. The remote attacker can then exploit vulnerable computers unlimitedly. In a local attack, the intruders try to obtain a higher privilege, i.e. root right, on other computers. Most of the security weak points are discovered in operating systems and different applications [95]. It is always good practice for organizations to keep patching and updating the software when security vulnerabilities are reported. Several popular websites contain a lot of vulnerability information. Of them, Bugtraq, CERT advisories, and RISKS Digest are good resources to provide helpful information.

Bugtraq: The Bugtraq uses an email discussion format to post security issues. The website is managed by www.securityfocus.com. A lot of security vulnerabilities were

first reported in the mailing list. The topics in the mailing list cover new security issues and host technical discussions for the solutions. Information from Bugtraq is relatively reliable and cited by many professional people. The Bugtraq often fully uncovers security vulnerabilities to the public, thus gives some pressures for vendors to fix the problems more quickly. The Bugtraq plays an important role in the computer security community.

CERT Advisories (www.cert.org): The CERT is an information center for the Internet security. It is hosted by the Software Engineering Institute at Carnegie Mellon University, and the research is funded by US federal government. Research concentrates on Internet vulnerabilities, support incident responses, and deliver security warnings. The CERT tends to provide services for important security problems and does not pay too much attention to low-level malicious activity. Some people are not satisfied with the slow response from the CERT, however. The delay in the past was attributed to the policy that attack incidents were not allowed to report without the patches being ready to deliver. If the CERT handles the incident, it provides the important information for risk management and continues to play an important role in handling major vulnerabilities.

Risk Digest: The Risk Digest uses a mailing list to deliver security related information. Risk Digest is managed by Peter Neumann. The mailing list discusses all topics regarding security, safety, and reliability. Risk Digest has a good connection with research community, and it is one of the first resources to report complicated attacks found by that community. A lot of Java security problems are first reported there. Interested users can subscribe the mailing list by sending an email to risks-request@csl.sri.com.

Technical trends and software security: Complex and large systems, compared to smaller systems, can carry flawed codes more easily. Their inherited complexity easily hides malicious codes in subsystems, due to detection and debugging difficulties. The flawed codes are a root cause for most of the vulnerabilities, and configuration errors account for other vulnerabilities. Extensible systems and programs are easy to introduce new vulnerabilities when the systems are extended. One of the biggest challenges nowadays is that the Internet is everywhere.

The quickly expanded Internet has increased the number of attack paths. The Internet not only makes local attacks easier, but also makes remote attacks possible. For example, if a computer runs Windows NT systems, the Windows NT has 35 million lines of source codes. The Windows NT systems can carry flawed codes and functional modules. The good function of a computer depends on proper kernel functions and its application functions. With multiple flawed codes, it is possible for an attacker to use them to get into the computer and control the computer operations.

The third challenge for software systems is that current software systems are easily extendable. For instance, a web browser plug-in is easy to install a new extension. Normally, we believe that the browser can manage security for the system. Practically, it is difficult to determine how a browser can have security operations, and it is hard to prevent malicious codes from getting into the extension codes. Analyzing an extensible system is much more challenging than analysis of a regular software system. Therefore, three challenges - pervasive Internet connections, increasing system complexity, and software extendability - make software security more important than ever [83], [95], [97].

How then can we define security? Different people have different concepts for security, or it can have different meanings for the same person. In my dissertation, security means that a policy can manage computer resource access and allocation. I define policy as the idea that different people have different access privileges. The computer systems have specified responses to each access request. For instance, if some users conduct a denial-of-service for the web-based applications, then, they are against the security policy of our computer system resources. Without the explicitly defined policy, it is difficult to define any kind of the security in computer systems [27], [71], [95].

Security goals: Software security has several goals. These include prevention, traceability, monitoring, privacy and confidentiality, multilevel security, anonymity, authentication, and integrity [97].

Prevention: The Internet not only improves the software development productivity because of communication efficiency, but it also helps to spread attacks. When a vulnerability is found by attackers, the attackers can disperse their attacks over the Internet. The attackers may use one script to hack multiple websites. Internet-based attacks on software systems are one of the most serious attacks. These attacks must be counted in the risk management matrix of the software project. In the software lifecycle, rigorous program code reviews are strongly encouraged to reduce potential vulnerabilities in different phases.

Traceability: A computer is difficult to operate at 100% security. The keys to restoring systems after attacks are to figure out when and how the attack happens. Auditing is not closely related to prevention approach; however, auditing may reduce

potential attacks to some degree. Auditing has been well applied to accounting, banking, and other financial areas. From the learning in these areas, we know auditing can help computer security professionals to detect and reduce potential attacks in the information systems. Ye et. al. (2001) developed a probability model for intrusion detection using audit data [103]. The results indicated that the audit data can identify the critical property for the intrusion detection. Multiple audit events are required to provide sufficient statistics for the intrusion detection.

Monitoring: It is a basic form of intrusion and detection technique. An easy way of monitoring is to check the known signatures, traffic patterns, and low-level system calls to detect the attacks in a real-time manner. Real-time intrusion detection can reduce potential damage. For example, tripwires can detect an attack in a real-time manner and may mitigate the severe attacks in a computer system.

Privacy and confidentiality: Privacy and confidentiality are very important for all the applications in different domain areas (i.e. business, individuals). Business companies must keep their commercial secrets from their competitors. Individual web users want to protect digital activities from exposing to others. Software is designed to run on computers to implement some specific functions. A running program on the computer may have access to other sensitive data in the same computer, and thus may cause potential security breaches.

Multilevel security: Government agencies normally have different levels of information security, specifying from open access to only for official use through unclassified to top secret level. Some companies also have different information security levels for different kinds of their employees. Multilevel security is one of the most

important approaches for information access control. For example, multilevel information security in a military battle field is extremely important.

Anonymity: Anonymity is an important aspect of software security. Anonymous actions are very useful in preventing information exposure, but sometimes, anonymous actions are not allowed. All the users are required to authenticate themselves before they can access resources. Anonymous advantage depends on different application domains. The FBI's Carnivore system is an email tracking system by traffic monitoring. Web cookies are often analyzed by different companies to detect the customer behaviors commercially.

Authentication: Authentication is another important security objective. Authentication is an important way to allow honest users to get into the system and prevent illegal users from being in the system. Specifically, as the Internet become pervasive, authentication becomes more important than ever before. On the Internet, users normally trust that a page is linked by hypertext page, but it is hard to indicate trust through a hypertext link. No one can tell whether www.goodbank.com is a good bank or not. Secure socket layer (SSL) provides secure information communication between the browser and the web server. However, other users can go to the end of information communication to get the transited information. From an authentication view, you may need to consider where your connection goes.

Integrity: Integrity means that the material is not changed from the original. Different from the authentication controlling user access, the integrity is about whether the material has been revised since its creation. For example, stock price in a company may be changed by a dishonest employee. Electronic information is relatively easy to

modify. Information integrity is very important in many applications (i.e. banks, other financial organizations).

Other common software security attacks: Although there are many different ways to attack, I have only listed the most common ones. Eavesdropping means that hackers read data while the data travels over a network. Tampering means that hackers revise data while the data is in travel. Spoofing means that hackers use bogus data to give the false impression about the valid data. Hijacking means that hackers alter data on the Internet with their own data while delivering the data. For example, when a user launches a FTP connection, hackers can control the connection by submitting bogus packets [95]. This category in hijacking has some relationship with the spoofing.

2.1 Security Analysis Model

As we discussed before, the software security goals are addressing preventions, traceability, auditing, monitoring, privacy and confidentiality, multilevel security, anonymity, authentication, and integrity. The Internet has fundamentally changed the role that software plays in business [95], [97]. Software project goals are emphasizing functionality, usability, efficiency, time-to-market, and simplicity. Generally, two sets of goals can occasionally conflict with each other. For example, aggressive software code reviews to reduce vulnerabilities may postpone time-to-market, or good usability in software systems may not maintain the system simplicity in web-based applications. Multilevel security may reduce the efficiency in productivity and communications. It is hard to determine which is more important. The decisions are very different and are greatly affected by business objectives and other concerns [97].

In the Internet era, software does not only run on a local computer machine anymore, but instead provides direct access to information everywhere. The most challenging problem is that many computer professionals do not know exactly what the security problem is. Even if you have the best firewall, when the firewall opens a port to allow a remote user access, the software systems are remotely exploited. The firewall does not prevent attacks from the open ports. Strong encryption is not an effective way to protect data from attackers also. Attackers can go to the end of communications and steal data [95].

Three-tier architecture for the web-based applications has also become pervasive quickly. There is a web client in the front end. In the back end, there is an information manager system (database server). Shortly, hospitals develop their web-based health care information systems. Banks develop their online bank systems [96], [108]. As the web-based applications are developed, the performance and availability of the Internet receives more attention. Quality-of-service (QoS) metrics in response time, throughput, and availability are widely discussed [58].

The major challenge in the Internet nowadays is response time, throughput, and availability. For instance, the Internet service must provide the sufficient process speed and average response time to meet their customer expectation. To define the relationship between the throughput and availability, a performance model is proposed by Menasce (2004) [58]. We can assume that the Internet data centers have M similar machines to process user requests. If each machine has a processing speed in d requests/sec, the highest processing speed from the working machines can be expressed as (d* M) requests/sec. When a machine has a failure rate at α failures/sec, a failed machine goes to

a waiting queue and will be repaired. It will be in operation again when the machine is fixed. However, there is a tradeoff between performance and availability. The cost of operation moves up as the throughput and availability go up. However, the failed machines will go back to service faster. The expected throughput TH in the Internet service is expressed by g machines in operation and the probability of the g machines in services. So, mathematically,

$$TH = \sum_{i=1}^{M} (gd) * p_g = d \sum_{j=1}^{M} g p_g = d * \overline{M}, \qquad (1)$$

where $\overline{M}$ is defined as the anticipated number of working machines. The p_g is the probability of g machines in services. It is related to the machine failure rate, number of repairmen, and expected time to bring a failed machine to work. The availability is the proportion of time that the service is available. From Equation (1), we can easily see the optimal operation strategy in the Internet data center.

As research on the Internet performance progresses, Internet system reliability and security in the web-based applications become prominent challenges for information technology. Attackers compromise software. Software is a root cause for common computer security. When your computer system does not behave properly, reliability, safety, or a security problem may be one of the major causes. Attackers do not make security vulnerabilities. They detect them and make good use of them. Poor software design and coding are the root causes of security vulnerabilities [95].

Fault is a noticeably different behavior of system characteristics from the accepted and normal conditions [38]. A fault is a state in the software or hardware system. The unusual behavior appears to have two forms: fault value and the violated

limit for usual value. A fault tolerant software system can keep working properly after software system faults occur [73], [88]. A fault-tolerant software system has the properties that software system faults do not result in entire system failure. Four general ways can be used to recognize fault-tolerant functions:

(1) Fault detection: The system explores a fault that could lead to a system failure state. This may be implemented by checking whether the system state is consistent or not.

(2) Damage evaluation: The components affected by the fault must be found out. The system damage will be assessed according to the severity of a fault. The potential loss will be estimated.

(3) Fault recovery: The fault system comes back to a safe state. This may be realized by putting the damaged state back to a correct state in forward recovery or by moving the system back to a previous state in backward recovery. Which recovery algorithm is adapted depends on the specific system implementations.

(4) Fault repair: The systems in software or hardware will be changed so that the fault does not happen again. Many software faults appear to be in transient states. They are caused by an irregular system input. The software systems do not require repairing, and normal operation can come back immediately after recovery. For example, in an airplan system, we need to estimate the severity and potential damage to the system if a specific part fails to function properly. We need to consider fault recovery for different parts if the occurring fault is critical for the system function. Furthermore, we need to make assessment whether we need to repair the fault if the fault occurs in the airplane.

In the web-based application systems, many researchers have looked into the fault-tolerant approaches and proposed fault-tolerant web-based application systems [2],

[34], [39], [61], [108]. Aghdaie and Tamir (2002) developed a fault-tolerant web service by kernel support [2]. Their implementation applied a client-transparent mechanism to fault-tolerant services. The revised kernel required to multicast the messages to a web server and a backup server. The backup server delivered a reply to the requested client when a primary web server failed. An example for modifying the Linux kernel and the Apache web server was provided. The analysis for the performance in throughput, latency, and consumed processing cycle time was conducted. Janakiraman et. al. (2005) proposed a Cruz system which used system checkpoints to restore application state at user and OS levels [39]. The fault-tolerant mechanism was realized by recovering applications from a failure using backward recovering algorithm. Hong et. al. (2005) proposed a replication of information by multiple servers for the Internet banking systems [34]. A dispatcher was used to coordinate web client requests among web servers. Multiple dispatchers were established to improve the web services with fault-tolerant capability. Their approaches can be summarized in four aspects: 1). use replicated servers to improve system reliability; 2). use multiple phase commit protocol to deliver requested services; 3). kernel-level and web server modifications to support logs of requests and delivery information; and 4). multiple dispatchers or coordinators are deployed to increase system fault-tolerance. These approaches provide a good solution to the web-based applications with fault-tolerant functions.

As the Internet become essential to many applications, the computer systems become more vulnerable than ever before. Attacks are observed very frequently in different software systems [25], and incidents are reported in many different application domains. To face current situations, studies have been conducted in intrusion-detection

and intrusion-tolerant systems [25], [29], [55]. This work centers on security-tolerant system development by measuring the security attributes of an intrusion-tolerant systems. Traditional security techniques are not sufficient to handle these challenges; therefore, the fault-tolerance approach to security becomes a cost-effective option [25]. Intrusion can be treated as a special fault in a fault-tolerant system. In this approach, internal errors are separated from the external failures. All the fault-tolerant methods rely on detection and recovery from the internal approaches. The distinction between internal error and root cause is important. One noticed deviation in a system could be caused by different factors: a usage profile, an accidental fault, and a disaster fault. An intentional fault is defined as an intrusion [25]. An intrusion happens when an attack successfully detects a weak point in software systems.

Attacks are a special kind of human activity. When paralleling the security to a fault, there are three types of fault-prevention methods. First, there is attack prevention in a human sense. For example, in the web-based applications, we need to prevent human attacks to our systems since the web-based applications have open access nature. Second, attack prevention can be in a technical sense. In the web-based applications, we need to prevent technical attacks to our systems as they are directly related to the reliability of the web-based applications. Third, fault prevention can also be vulnerability prevention. The vulnerability prevention may take several approaches. The approaches contain formal specification, good design, code reviews, and user education. Vulnerability prevention can reduce the vulnerability from the root cause.

Fault removal may take place during software development in verification and validation process or after the product is in operation. Fault removal can be realized in:

(1) Attack removal in human sense. In the web-based applications, we need to mitigate the rate or level of human attacks to our systems below the allowed threshold. (2) Attack removal in technical sense. In a web-based application, we need to mitigate the rate or level of technical attacks to our systems below the allowed threshold. (3) Vulnerability removal. In the software development cycle, formal proof, model checking, and code reviews can help to identify code flaws which carry potential security problems in the future. When software systems are in operation, vulnerability removal can be implemented using security patching, removing a specific service, etc. Vulnerability removal can improve the system security tremendously. Intrusion tolerance means that systems are reliable enough to continually perform the planned services when an intrusion occurs.

Fault forecasting is about evaluating the fault prevention, removal, and tolerance methods. Fault forecasting has: (1) Attack forecasting in human sense. In our web-based applications, we need to predict future attacks and hazards from human activity. (2) Attack forecasting in technical sense. In our web-based applications, we need to predict future attacks and hazards from technical sides. (3) Vulnerability forecasting. We predict future vulnerabilities and potential hazards from our existing knowledge. Correct predictions can help us to assess potential vulnerability, estimate potential damage loss, and prepare for immediate response if the threat from a vulnerability becomes real. Sousa et. al. (2005) introduced a system model to represent an exhaustive-safe system [88]. Ye et. al. (2004) evaluated the Markov Chain Model for cyber-attack detection in Unix Solaris systems [104]. They found that the Markov Chain Model is better for cyber-attack detection than the chi-square distance test method for low noise level in the data.

The Markov Process Model has been applied to different applications. The Markov Chain Model has “memoryless” characteristics. The Markov model has the properties that state the future is only dependent on the present state [26]. For a web-based application, security risks are most correlated with the current software systems and hardware systems. So, current system state has a decisive impact on the system security in the future. System security risk has a kind of “memoryless” nature which the Markov Model also holds. On the other hand, the Markov Chain Models have been used in some studies [51], [101]. Since web vulnerable risk is highly related to web access traffic, security risk in the systems appears to have similar pattern as access traffic cross time. The more access traffic to web applications, the more security risks the web-based application systems may experience. Therefore, security in the web-based applications can be approximately modeled by the Markov Chain Model.

Jones et. al. (2006) modeled the security and vulnerabilities in facility infrastructures [41]. They used a network graph to express the system architecture. The model provided the break-in probability estimation for security breaches. Software risk management consists of security, reliability, and safety. However, software risk management is a relative new subject [95]. A common joke about the most secure computer: the computer has its disk removed and power off. As a result, the most secure computer is functionless. Good risk control needs professional knowledge in application domains. The security officer must be able to separate known attacks and possible vulnerabilities in the system. When risks have been explored, we can rank the risk in severity. Risk identification and severity ranking will decide resource allocations for further analysis and mitigation. Resource allocation is a business-decision process based

on good data. Mitigation for the vulnerability depends on good knowledge about the risk, proper assessment for the potential damages, and good strategy in response to the vulnerability. The goal of mitigation is to prevent the risk occurrence.

Network security analysts tend to approach security problems from a network perspective. They tend to address the security in firewalls, intrusion detections, and policy control, etc. When the security analysts and software developers meet together, the result is not as good as we expect. Security evaluations for software systems are normally conducted at the end of a project. This can easily lead to a disagreement because the software developers may think of the reviewers as nonprofessionals in programming. There are two popular approaches used by security analysts to address software security: black box testing and red teaming approach [97].

Black box testing: Black box testing is not as productive as white box testing (understand the architecture of codes and then develop test cases). In the views of Viega and McGraw (2002), a risk analysis directs the testing activity [95]. We need to make a trade-off with security. The black box testing for security functions is inefficient; this method does not make good use of the system architecture. The black box testing can detect implementation errors, but misses errors regularly. Most of the black boxing tests look for surface errors.

Red Teaming Approach: The red teaming approach lets a team attack the software systems as hackers. The test allows a group of people with different experience to break into the systems without giving any instructions. If the testers cannot detect any problems, the program may be good. If they do find some problems, one analysis between root cause and potential risk is conducted. We can fix the problems. However,

the claims about the systems that have some problems could be misleading. Previous case studies indicate that it takes a lot of time and effort to discover security problems. People in the red team approach may have different experiences, but the efforts taken by the red team approach may not be sufficient. Real hackers may invest a lot of time and effort into attacking your software systems. The red team approach could only look into the surface errors because of the time limitation.

An efficient way to reduce vulnerabilities is to conduct a security code review (Howard 2006) [36], [97]. In the code review, we need to know what we are looking for, then rank our tasks in priority order, and finally examine the code. In the first phase, Microsoft often establishes a review group consisting of some senior and new reviewers. The reviewers discuss different review strategies, and are encouraged to learn from each other. If the reviewer is totally new, s/he may need to check some popular websites (i.e. www.securityfocus.com, Bugtrag, etc.) for references very often. In the second phase, ranking all the tasks in priority is very important. Howard prioritizes the code review to address old software systems, programs running by default, programs running at a higher privilege, anonymously accessible program, programs listening on the Internet, software systems written in low language levels (for example: assembly), software systems with a history of security problems, software systems that interact with a security-sensitive data, larger complex software programs, and frequently updated programs. In his view, an old software program is likely to carry more vulnerabilities because modern developers can address security issues better.

Attackers frequently make good use of the program running by default. This kind of running program provides more opportunities to attackers. Program running in a

higher access right has a potentially serious problem because the program has more access to the software systems. A program that listens on the Internet is open to attackers remotely and locally, while a program written in lower level languages (i.e. assembly) provides easy access to hardware (i.e. memory). Buffer-operation can result in buffer overflows. A program that has had vulnerabilities in the past is more likely to have vulnerabilities again in the future. Programs that deal with sensitive data are important because we do want to preserve the data confidentiality and integrity for all software systems. Larger software programs often have more vulnerabilities inside the system because it is difficult to detect and debug them. It is also easy to add new vulnerabilities to frequently revised programs. To review a program, we need to use all code analysis tools, look for common security problem patterns, and investigate deeply into the programs. For a specific system, we need to analyze the system characteristics and compare these characteristics with the points discussed above. We pay particular attention to the system with the potential problems listed in Howard's paper.

The web-based applications are exposed to security threats by worms, viruses, spoofing, and many others [54]. In the past, Internet-based attacks have been widely waged against the web-based application systems. Lu. et. al. (2005) applied the technology acceptance model to the perceived risk analysis for online applications with security threats [54]. Their results indicated that the perceived risk highly influences continuous users. Conceptually, for a user who frequently uses online banking systems, he/she may pay greater attention to the perceived risk related to online transaction security. The perceived risk and related threats can restrict the user's desire to continue using online services. Risk metrics can measure assets, threats, and vulnerabilities in a

software system (Peterson 2006) [71]. Taking into consideration all of the related elements, we can produce a risk management model. Based on the risk model, we can then manage or mitigate various risks. In a web-based application, the risk is different from uncertainty because it is measurable. A lot of vulnerabilities are well-known to the public so that we can reduce their impact to some degree. However, the measurement method is not standardized yet. Threats to a system have a lot of uncertainty because the threat can be affected by many factors. Vulnerability management includes vulnerability evaluation, remediation, and redistribution. In a risk calculation equation, assets have a value. We can estimate the asset value from the expected loss. Schechter (2005) at MIT Lincoln Laboratory defines [81]:

Security risk = (chance_of_security_breach) x (cost_of_security_breach) (2)

When a system has multiple security problems, it is reasonable to define security risk by the frequency or expected rate of breaches. Therefore,

Security risk = (security_breach_rate) x (average_cost_per_breach)(3)

Most recently, Linstrom (2005) and Ravenel (2006) defined [52], [76]:

Risk = threat x vulnerability x expected_loss (4)

This is one of the most practical measurements for computer system security. In this equation (4), Linstrom (2005) and Ravenel (2006) introduced the expected loss [52], [76]. Lee and Shao (2006) developed a new method to estimate IT security loss [49]. Previously, IT security analysts used two major methods to estimate IT security benefits: annual loss expectancy (ALE) and cost-benefit analysis (CBA). The ALE is computed by the expected loss and the rate of loss for each attack in the attack rate over a specific time period. Lately, the risk ranking method and management efficiency model are developed

based on the previous work. However, these models are subjective and time-consuming. It is difficult to obtain an accurate estimation in a good time manner. The cost-benefit analysis (CBA) takes the risk-adjusted costs to approximate the internal return rate and total net value in order to calculate the ratio of the inputs to profits. To apply CBA, a cost metric is applied to appraise the damage, response, and operational costs. This approach is popularly adapted by the US National Institute of Health. The cost metric seems to be varied in to a large degree. In Lee and Shao's approach (2006), a stochastic model is developed for estimating the expected loss. Their model appears to be a better measurement. More recently, Whitehat security (www.whitehatsec.com) and Applied Research Associates Inc. propose that the security risk is measured by threat and vulnerability without the expected loss term [30], [78]. My understanding is that the expected loss can be a kind of subjective measurement. It may change with different application domains even if we have the same web-based software system. So, the security risk can be expressed as:

$$\text{Risk} = \text{threat} \times \text{vulnerability} \quad (5)$$

In this dissertation, the vulnerable behavior and system response will be analyzed. The analyses will focus on availability-related aspects: the probability of moving the web-based applications to a particular security-failed state and the mean time to the security failure of a system. The security analysis in the steady-state will be determined using the Markov chain analysis. The system security risk index will be classified in three levels as an indicator of the level of security risk at low risk level, high risk level, and failed risk level. The vulnerability risk analysis will be measured by a group of security experts in the web-based application systems. The measurement justification is taken

using information entropy weight coefficient to counter the subjective factors from the security expert assessments.

2.2 Security Improvement

When users get access to a computer system, the system will decide the resource for each user access [96]. Many access control models have been proposed to address this policy. Some complicated models have been developed in distributed systems. Access control in Unix and Windows is realized by using Access Control Lists (ACLs). In a Unix system, each user is identified by the user ID (UID). An individual group has a group identity (GID). Based on the resource allocation policy, we can set access privileges for each user and each group. In a similar way, Windows also have security IDs corresponding to different users and groups. Windows may use several types of tokens for access control implementations. The access token stores the information for the different authenticated users. When determining a particular access, the security modules check with the access token to make an access decision. For a more detailed discussion, please refer to the “Building secure software” book by Viega and McGraw (2004) [98]. Other systems, different from Windows and Unix, have their own control mechanisms similar to the ACLs.

The objective of information system security is to preserve systems and ensure that they have proper utilizations according to the specific policy [96]. The information security specifically addresses confidentiality, integrity, availability, and accountability in the resource access and allocation [42]. Access control has been widely used in

information security area. Access control can specify a user's access right and when the user has access to the allocated resources [45], [96].

The distributed and loosely coupled architecture of the web-based applications present a big challenge in verifying credentials in access control [22]. Though access control for web services has been studied, no specific architecture has been publicly accepted. For web services, four key elements are important in security. These elements are resources, policies, validation, and management [107]. Resources consist of the participants in organizations, developers, and customers in the project development life cycle. Policies are the guidelines that determine the factors in security breaches. Validation processes check software related security attributes and confirm the software system is working as designed. A combination of software attributes in dependability, interoperability, and fault tolerance is a major consideration for validation [66]. Management is observing web projects in development life cycle closely [95], [107].

The GeoIP services are database systems that convert an IP address to its physical location. The GeoIP services work easily with other software systems [64]. The GeoIP services can convert 4.3 billion used IP addresses to the corresponding locations. When integrating the GeoIP services with a web server, the web server can identify the location of a remote user without additional effort. The physical location information contains an IP address and its geographical information. Using the GeoIP services with the role-based access control will improve the security for the web-based applications [96]. As the second objective of this dissertation, I will present the specification and implementation architecture for a secure web-based application.

CHAPTER III

OVERVIEW OF THE RESEARCH

3.1 Goal of the Research

The purpose of this research has two aspects. First, system security and risk analysis will be explored. The goal of the research is to develop measurement methods for security and vulnerability risk for the web-based applications as these applications have become a major computing platform. Second, I develop a security improvement model to enhance security for the web-based applications.

3.2 Description of the Research for Web-based Applications

Though research on information security and risk analysis has been ongoing for some time, no specific security analysis model exists for the web-based applications. Maden et. al. (2002) developed a generic security model to describe the system security in {good, vulnerable, attack, triage, and failure} states in an intrusion-tolerant system [55]. They only covered a single system without any existing subsystem interactions. In a typical web-based application, the whole system consists of a web client, web server, and database server. The subsystem interactions obviously exist. For example, a web client may interact with a web server so that the web client can obtain the desired data access from the database server. Mustafa and Fai-Bahar (1991) performed research on project risk assessment model [62]. Peterson (2006) developed a risk measurement metrics for software systems [71]. However, these models are not well validated in the web-based

applications because the applications have their own characteristics [10], [93]. Their evaluation about the vulnerability risk is a subjective-oriented measurement. This research tries to formalize security analysis and risk measurement for the web-based applications. Hopefully, the proposed methods can be standardized in the near future. In recent years, different access control models have been discussed. Web computing encounters new challenges in security every day. To meet the incoming challenges, it is imperative that we develop new security improvement models.

CHAPTER IV

METHODS USED IN THE RESEARCH

In this dissertation, I will apply different research methods in security analysis, risk analysis, and security improvement model to the web-based applications. The applied methods are summarized as follows:

4.1 Security Analysis Modeling

In the security analysis, the Markov Process Model is applied to the web-based application systems. Here, I will review the key component for the Markov Process Model Analysis [26], [80]:

A Markov Process $Y = \{Y_t; t \geq 0\}$ has finite state space when $j \in E$ and $t, s \geq 0$

$\Pr \{Y_{t+s} = j \mid Y_u; u \leq t\} = \Pr \{Y_{t+s} = j \mid Y_t\}$

The Markov process has stationary transition probability if

$\Pr \{Y_{t+s} = j \mid Y_t = i\} = \Pr\{Y_s = j \mid Y_0 = i\}$

When a Markov Process has $Y = \{Y_t; t \geq 0\}$ has finite state E and jump times $T_0, T_1, \ldots$ and the embedded process at the jump time expressed by $X_0, X_1, \ldots.$, there is a set of scalars $\lambda(i)$ for $i \in E$, called the mean sojourn rates and a Markov matrix P (the embedded Markov Matrix) that meet the following conditions:

$\Pr \{T_{n+1} - T_n \leq t \mid X_n = i\} = 1 - e^{-\lambda(i)t}$

$\Pr \{X_{n+1} = j \mid X_n = j\} = p(i,j),$

where $\lambda(i) \geq 0$ and the diagonal elements of P matrix are zero.

For the embedded Markov Chain, the analysis is summarized as follows:

I. Identify irreducible sets in the Markov matrix P.

II. Reorder the matrix P so that irreducible and recurrent sets on the top, transient states at bottom of the matrix P'.

III. Steady-state analysis for irreducible sets using the following equations

$\sum_i \pi_i P_{ij} = \pi_j$ and $\sum_j \pi_j = 1$.

IV $N_T = (I-Q)^{-1}$ for transient states

I is identity matrix. Q is a submatrix associated with the transient states in the Markov matrix P. N_T is number of visits for Markov Chain to the fixed state.

V $F_T(i,j) = 1 - 1/N(j,j)$ if i = j or $F_T(i,j) = N(i,j)/N(j,j)$ if $i \neq j$, where $F_T(i,j)$ is the first passage probability that Markov chain eventually reaches state j at least once from initial state i.

VI The probability f_k from a transient state i to the k-th irreducible set with the sub-matrix b_k can be calculated by

$f_k = (I - Q)^{-1} b_k$.

The Markov process steady-state probability p_j has a relationship with Π_j (the steady-state probability for the embedded Markov chain) as:

$$p_j = \frac{\pi_j / \lambda_j}{\sum_{k \in E} \pi_k / \lambda_k} \qquad (6)$$

In the first part of this dissertation, security will be analyzed using the Markov Process Model. The state in the Markov model is defined as the software programming

running conditions and data transaction process in a computer. The database server and web server are assumed to have sufficient speed to process the data transaction requests. In the web server and database server, there are multiple processes running simultaneously because of multiple accesses from different web clients. The web-based application security can be expressed in the states that describe a single subsystem (i.e. web server or database server) or in multiple components combined. In the embedded Markov Chain, the initial transition probability is estimated from the scan data. Then, the probability in steady-state can be calculated in the Markov Chain and Markov Process. The availability will be computed. The mean time to reach a security failure also can be calculated. To estimate security risk, equation (5) will be used. At the end of the analyses, an illustrative application example will be provided using real data. According to Arora and Telang (2005), the average vulnerability fix time in patching was 242 days for the CERT data (n = 186) [6]. In our case study, 242 days are used as an estimate of time span for state transitions from good->vulnerable->attack-> security_failed->good in the Markov model for data collection. So, we can approximate each state transition to have 60 days on average. Security analysis will be conducted using scan data. Risk analysis will be conducted using the vulnerability data. For vulnerabilities, the National Vulnerability Database (NVD) describes vulnerability into three levels: severity, medium, and low [63]. High vulnerability severity allows a remote attacker to violate the security policy (i.e. use a higher access privilege) or allows a local attacker to obtain control of the system.

TABLE 1
Vulnerability Severity Levels by Grossman (2006) [30]

Vulnerability severity level	Descriptions
1	Low information can be obtained by hackers on configuration
2	Medium sensitive configuration information can be obtained by hackers
3	High limited exploit of read and directory browsing
4	Critical potential Trojan horses; file read exploit
5	Urgent Trojan horse; file read and writes exploit; remote command execution.

TABLE 2
Threat Evaluation Criteria by Hickman (2004) [32]

Evaluation Criteria Number	Description
1	Potential damage loss
2	Re-occurrence
3	Exploitability
4	Impacted users
5	Degree of discovery

Low vulnerability severity means that the vulnerability does not provide valuable information or the control of a system. The vulnerability provides attackers some information and helps attackers to exploit other vulnerabilities. The vulnerability severity also provides some assessments for organizations about the potential risks. In my dissertation, I will use industry vulnerability levels which can describe the vulnerabilities in details. According to Jeremiah Grossman at Whitehat security (2006), vulnerability can be classified into five levels [30]. The five levels are described in TABLE 1. Vulnerability risk will be estimated in equation (5) because vulnerability severity level determines the degree to which hackers control the web-based application systems. The threat can be evaluated based on the five categories of descriptions in TABLE 2.

Due to the uncertainty of the risk factor, fuzzy logic method has been popularly applied to the risk analysis. The analytic hierarchy process (AHP) was introduced to analyze the project risks by Mustafa et. al. (1991) [62]. The process uses multiple criteria in subjective and objective factors to assess the project risks. To overcome the subjectivity of a risk assessment, Zhao et. al. (2005) extended the AHP model using fuzzy logic method and entropy weight coefficient [109]. In the past, it was very difficult for the security community to collect detailed security data. Open proxy honeypots have been used to collect data for network security research [75]. Proxy is a software program which can work as a server and a client to make requests for other clients [8], [9]. Open proxy is a proxy server that does not have any access control and is open to access to other requests. The Internet connected to the proxy makes a request of the proxy server to access other Internet hosts. The proxy server has both a forward proxy and reverse proxy

server. The forward proxy is a server that is located between the client and the origin server. The reverse proxy adapts the client as an ordinary server [8]. Barnett described how to set up an open proxy honeypot. In his approach, the first layer of control is implemented by a router. Then, all the listen ports are closed. Finally, the Apache web server is configured as an open proxy. The purpose of open web proxy honeypots is to collect evidence of the actual threats in the web-based applications. The data collected can provide a real example for security analysis after attacks, identification of new threats, and statistical analysis (Dacier et. al. 2004) [24]. Security data in this dissertation was collected by Dr. Anton Chuvakin from February to March, 2005, and posted in http://www.honeynet.org/scans.The data for security analysis is processed in a similar way to the process described by references [8], [9], [69]. The vulnerability risk data is collected by Purdue University in the vulnerability report in https://cirdb.cerias.purdue.edu/coopvdb/public from July to September, 2006 [74].

4.2 Security Improvement Model

Formal methods are common mathematical approaches to software and hardware system development from requirements, specification, and design, as well as to system implementation [98]. The formal methods are used as theoretical tools in software engineering, particularly in the safe and secure systems, and these methods are also widely used in software testing due to their ability to reduce errors and provide a framework for testing.

In the security improvement model, the formal methods are used to apply security to system specification. Mclean (1999) reviewed the application of formal methods in

computer security [56]. The results indicated that applying the formal methods to computer security is a cost-effective approach. Because computer security covers numerous threat scenarios, the formal methods are suitable for most cases. One example where there are difficulties in applying formal methods to applications is an investigation into the attacks by terrorist groups. Computer security is involved multiple features for different programs. As the Internet becomes omnipresent, authentication has become a major problem for the security. In Mclean's words, the better we know what we are trying to do, the better we can make good use of formal methods. One of the big challenges in applying the formal methods to computer security is the difficulty to specify these features. Information flow is not functional property because it is difficult to decide whether a computer is operating with the desired function.

Mitchell (2002) at Stanford discussed the relationship between the formal methods and computer security [60]. In his view, the formal methods are popularly used in Java bytecode verification, protocol analysis, and security trust management in access control policy. Future challenges in computer science rely on the capability of software development process improvement (design, develop, and quality control). The formal methods can be used in analyzing a software system from its description and specification. Analysis depended on specifications between the system description and related properties. One good example of formal methods is the TCAS system [50]. The TCAS is specified by the formal methods to describe how the TCAS answers to sensor inputs. Analyses focus on module specifications and intended functions. When all the modules meet the specifications, an aircraft can fly as designed. In Mitchell's points, the formal methods can integrate a lot of past experience. A disadvantage of the formal

methods, however, is that some system features are hard to formalize. Nevertheless, the formal methods are still a popular approach in the software engineering areas of validation and verification. In the next part of my dissertation, I will apply the formal methods for a security system specification using role-based access control for the web-based applications. The specifications include subject credentials, object credentials, access privileges, and access control algorithms in access control.

CHAPTER V

SECURITY AND RISK ANALYSIS MODELS*

5.1 General Vulnerability Scenarios in Web-based Applications

In a typical web-based application system, the systems have a web client, web server, and database server subsystems. The web server and database server may be located in the same local area network. However, the software systems in the web server and database server are very different. In the web-based application systems, the database server normally stands alone. The computer with the database server typically does not have too many other network applications (i.e. a web browser) on the machine. In general, the web client and the web server do not belong to the same local area network. So, the software systems on the web clients and the web server tend to be divergent. The database server is hidden to public computer users in general (domain name and IP address). However, computer hackers have their ways to locate the database server. The web application vulnerability may come from software-based, hardware-based, and network-based aspects in general. The software-based vulnerability includes PHP scripts, programming buffer overflow, etc. The network-based vulnerability includes unauthorized access to an application system. The hardware based vulnerabilities are related to access to the hardware server or data. The vulnerability from the software systems and its related security and risk analysis are the focus of this dissertation.

---

* Part of the chapter is reprinted from "Software security analysis and assessment for the web-based applications" by Yong Wang, William Lively, Dick Simmons in the 17th SEDE conference with permission ©ISCA 2008.

For many web-based applications, users are required to identify themselves before being allowed to use some application features [27], [35], [90]. In some systems, applications may not only ask users to identify their identities, but also to confirm who the individuals are supposed to be. In a web client, the web applications use one of two standards to download executable programs over Internet: Java applets and ActiveX controls. The ActiveX is a modular of dynamic link library. It is developed by Microsoft using DCOM (distributed component object model) or COM technology. ActiveX can be used to download or run programs. For example, an ActiveX downloaded file from a web site can be run by a javascript in the other web browser. The early Java used a security mechanism: sandboxing. Sandboxing is a programming running environment to run other software. The sandboxing allows the program from others to run inside a security area and controls the applet to access outside areas.

In the web server, the first type of vulnerabilities is directly related to programming languages used in the web-based applications. One popular vulnerability in the web server is to provide application data to an unauthorized user. The programs that carry vulnerabilities often appear in CGI (common gateway interfaces) scripts, SSI which is the server side include for html markers, ASP pages (active server pages), and many other application programs. The second kind of vulnerabilities comes from invalid input data in the web-based applications. For example, most inputs for web servers do not have good mechanisms for checking if input parameters are correct or not.

In database servers, the most popular attack is the SQL injection attack [29]. The SQL injection is a group of unverified user input problems. The web applications run the

SQL code that was not planned using string operations. The SQL strings are produced on the fly and run using string replacement or other operations. The SQL injection can create some real problems for the web-based application systems [35].

For example, a form is popularly provided to input a username and password for data retrieval in a web-based application. Let us assume that we know a valid login name in the database server is JohnDoe. We run a query using JohnDoe'- - as a username to get into the database server. In the above statement, adding single quote and comment characters to the name allows attackers to login as JohnDoe without the valid password. The comment characters (- -) tell the SQL server to ignore all of the remaining statement without any authentication. If we don't know any valid login name on a database server, we can add 'JaneDoe' as a new user; INSERT users VALUES ('JaneDoe', 'password'); - - into a login name field. The single quote closes the login name string. The semicolon indicates there is a second statement. The comment characters after the INSERT statement instruct the SQL server to ignore the password. The first time the attacker submits this login name, he probably receives an "access denied" message from the SQL server. The comment characters in the string tell the server to ignore passwords without any authentication. After the first time, the attacker has created a valid user in "JaneDoe" as login name and "password" as password in the SQL server. The attackers can unlimitedly access the SQL server after his first attempt [28], [35].

5.2 Vulnerability and System Responses in Web-based Applications

Maden et al. (2002) developed a generic state transition diagram in discrete time Markov chain for intrusion- tolerant [55], [97]. In their approach, they defined that

intrusion tolerant software systems start running from a good state, then transfer to a vulnerable state when the system fails to stand. When attackers identify a vulnerability successfully, the systems move to an attack state. When the software systems discover the attack, the systems move to the triage state (the state in which the system looks for possible ways to respond to an attack to limit the damage) to reduce the potential loss. If the attack is successful, the systems move to a security-failure state. The system might be shut down to fix the security problems in confidentiality, privacy, and data integrity. This action may result in the service being unavailable. In the web-based applications, a transaction starts from a web client. The web client sends its request to the

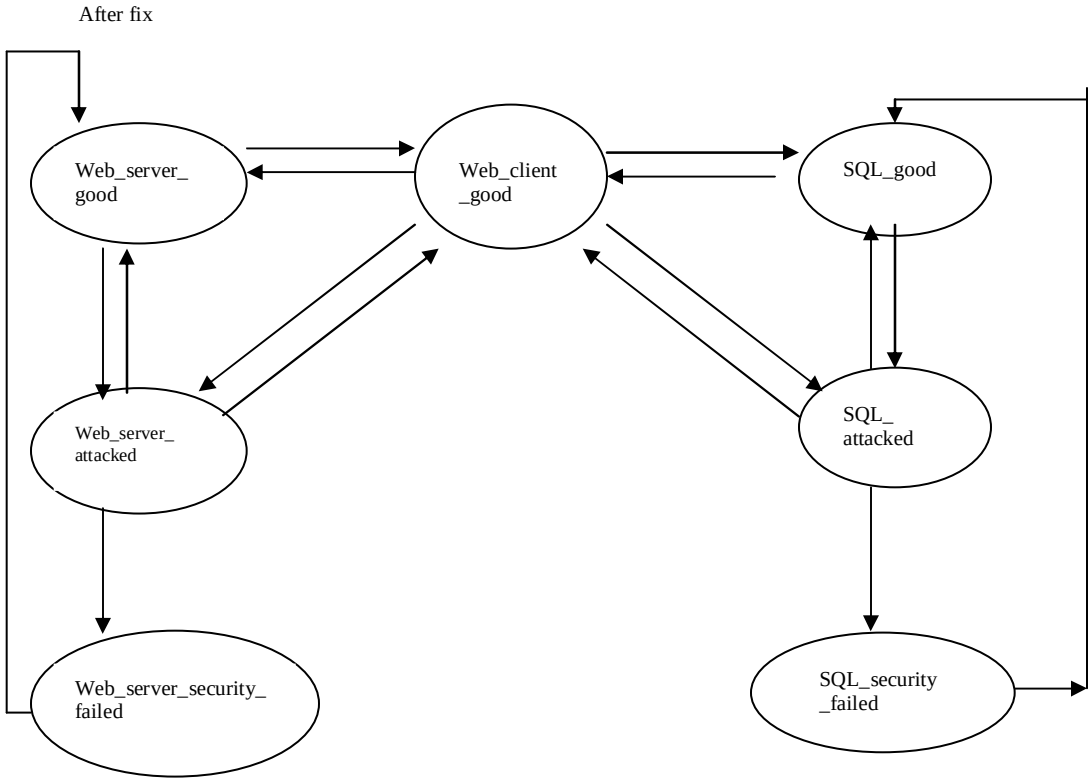

Fig. 1. System state transition diagram for web-based applications.

web server by sending TCP packets to ask for the desired data. When the web server receives the request from the web clients, the web server will authenticate the requested users by the user credentials. If the user can pass the web server authentications, the web server will retrieve the requested data from the database server. So, the web-based application systems consist of the web client, web server, and database server subsystem. In this analysis, we can conceptually assume that there are unlimited web clients possible to attack the web server and database server. So, the web clients are not a major target for attacks in security analysis in our web-based system. I assume that the web clients are secure in our modeled systems since the web clients are well distributed in the world, and it is very difficult to collect the data for all the potential web clients due to scattered distribution and data privacy concerns.

Although some researchers have modeled a computer systems in {good, vulnerable, attacked, and failure} states in the conceptual model [55], it is very difficult to separate vulnerable and attack states from the log data. So, here, I use an attacked state to represent the combination of vulnerable and attack states in Fig. 1. From the data collected, no data indicates that there are direct attacks from the web server to the database server. In the analyses, each subsystem is assumed to start from the good state. If the attacker detects a vulnerable point and wages attacks in the subsystem, the subsystem moves from a vulnerable state to an attack state. Each subsystem, after the attacks, either moves to a security-failure state or moves back to a good state if it has some intrusion-tolerant mechanisms. Each subsystem interaction also exists in the web-based applications. For example, the web client in the good state may directly launch

attacks on the database server without passing the web server in the good state. To outline our descriptions above, the state interaction diagram for the web-based applications is presented in Fig. 1. Fig. 1 is the case when a web client launches attacks on the web server and database server simultaneously. When having multiple process accesses to the web server or database server, the system interactions can be described using multiple diagrams in Fig. 1, multiple partial diagrams from Fig. 1, or a mixture of whole diagrams and partial diagrams in Fig. 1.

An attacker attempts to move the web-based applications to a security failure state. However, this attempt needs attackers to invest their efforts. These efforts can be described using time and modeled as a random variable. The random variable in time is assumed to have exponential distribution. In general, an attacker can only sometimes move the system to a security failure state. Each subsystem in the web server and database server can be modeled by {good, attacked, failed} states in the state transition diagram. So, we use exponential distribution to model multiple state transitions for the web-based applications. For example, the Nimda worm attacks web servers using file permissions, character decoding, directory traverse vulnerability. Finally, the Nimda worm moves the web server from good state to attacked state, then from attacked to a security failure state. Because there are multiple components in the applications, the system state may be described using more than one component. For example, one of the system states may be in (web_server_good, database_server_good) state in the Markov Process.

When attacks occur, the software systems try to move back to a secure state from a compromised state [97]. As Sommerville (2004) pointed out, many software faults are

transient [88]. No specific action is taken to correct the system. The faults may disappear in the system's next execution. When this is not the case, some actions are taken to repair. Each subsystem may move to a security failure state. Griffin et al. (2005) concluded that a system responding to an attack is similar to a system responding to accidental faults [31]. In the web-based applications, the system will be modeled by Markov Process with embedded Markov chain. The discrete state space E = {(web_server_good, sql_server_good), (web_server_attacked, sql_server_attacked), (web_server_good, sql_server _security_failed), (web_server_good, sql_server_good), (web_server_attacked, sql_ server_attacked),
(web_server_ attacked, sql_ server_security_failed),
(web_server_good, sql_ server_good),
(web_server_security_failed, sql_server_attacked)}.

To model the web-based applications in the Markov Process, we need the mean sojourn time in each state and initial transition probability p_{ij} from state i to state j in the Markov Model, where i, j Є E.

In the following context, we will use listed abbreviations to represent system states in the web-based applications:
(wsg, sqlg): (web_server_good, sql_server_good),
(wsg, sqla): (web_server_attacked, sql_server_attacked),
(wsg, sqlf): (web_server_good, sql_server _security_failed),
(wsa, sqlg): (web_server_good, sql_server_good),
(wsa, sqla): (web_server_attacked, sql_ server_attacked),
(wsa, sqlf): (web_server_ attacked, sql_ server_security_failed),

(wsf, sqlg): (web_server_good, sql_ server_good),

(wsf, sqla): (web_server_security_failed, sql_server_attacked).

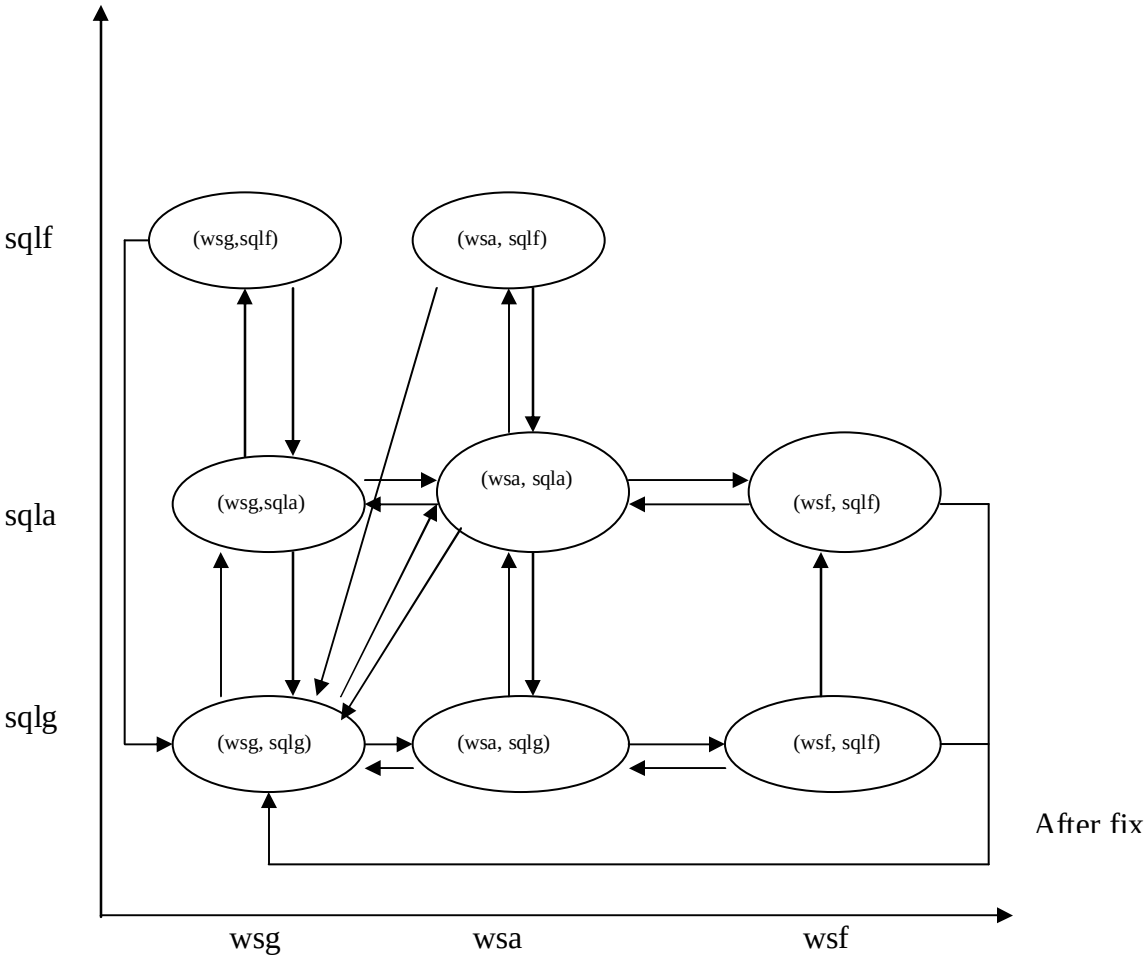

Fig. 2. System state transition in the web-based applications (Revised from Wang, Lively, and Simmons (2008) [97]).

The system state transition diagram in the web-based applications in the Markov Process is described in Fig. 2. In the embedded Markov Chain, a state's space can be classified into transient sets and irreducible sets. When a closed set of states that do not have a subset which is also closed, this type of subset is defined as an irreducible set. All states in an irreducible set belong to the same classification [94]. The steady-state probability can be calculated for different states in the embedded Markov Chain and the Markov Process.

In the web-based applications, we are specifically interested in system security in software. Hollar and Murphy (2006) summarized the security goals in web application triangle in integrity, confidentiality, and availability [33], [97]. In their notations, availability is the probability that a system can provide the intended service at a point in time when customers need them. For example, the availability can measure successful resource allocation rate to incoming requests in the web-based applications. Integrity is related to the correctness of the application data. Software security can prevent unauthorized people from changing the data. Confidentiality means that information must be kept private. The software system is expected to prevent information from unauthorized access. Which attribute is more important to the software system depends on different application domains. For example, patient information systems are more interested in confidentiality because the patient information is critically private. Bank information systems pay more attention to integrity. These systems always need to have correct numbers for their customers. When any of these attributes in the web-based applications are compromised, the action will lead to the application system security failure. The system security failure can be modeled by the Markov Process.

In a web-based application, any security failed states from the integrity, confidentiality, and availability will finally lead to system unavailability. In this case, the system needs to be fixed before it can operate properly again. To compute the mean time to security failure, we treat the security failure states as absorbing states so that we can conduct analyses for the Markov Process Model. In the next section, I will conduct analyses for the system functional availability and the mean time to a security failed state.

5.3 Availability Analyses

In the reliability area, Menasce (2004) and Trivedi (2001) defined the system availability using mean time to failure, and mean time to recovery [57] [94]. The web-based application systems will be shut down to fix the security problems if the system security fails. If any subsystem reaches a security failed state, the entire web-based applications will be considered as security failure because the system cannot support the requested services. Availability can provide a measurement for immediate access rate to the web-based applications to achieve desired services.

Let Av be the functional probability that the entire system can provide the intended services properly [97]. We want to derive the steady-state functional availability A as the time moves to the unlimited. In our security model, the entire system will not be able to provide the intended service when the web-based applications arrive at any state of {(wsg, sqlf), (wsa, sqlf), (wsf, sqlf), (wsf, sqla)}. When there are different vulnerability attacks existing both in the web server and database server, the system availability can be calculated as:

$$Av = 1-p_{(wsg,sqlf)} - p_{(wsa,\ sqlf)} - p_{(wsf,\ sqla)} - p_{(wsf,\ sqlg)}, \quad (7)$$

where (wsg, sqlf), (wsa, sqlf), (wsf, sqlf), (wsf, sqla), and (wsf, sqlg) stand for the different states described before, respectively. P_{state} indicate steady-state probability in a security failed state in the Markov Process. When systems only have database server attacks existing, we can treat the SQL server as in good state. Thus, the availability can be expressed as:

$$Av = 1-p_{(wsf,\ sqlg).}$$

Mean Time To Security Failure (MTTSF): In a web software system, it is common that the mean time to failure is used to describe software reliability (Tian et. al. 2004) [10], [93]. The mean time to failure (MTTF) measures the expected time for a system to reach the failure state from a good state. To parallel mean time to failure in software reliability to man time to security failure, mean time to security failure measures the expected time for a system to reach the security failure state from a good state. Mean time to security failure is an important system parameter to measure system reliability from a security aspect. The parameter can describe how long the system can provide trusted services. MTTSF is computed by making the states of the embedded Markov Chain in security failure into absorbing states. Classification of the Markov process states, into absorbing and transient states, relies on the nature of the analysis. For example, a vulnerability exists in the web server in a web-based application system and no other vulnerability exists. A web client launched a malicious attack to move the web server to a security failure state. The attack is modeled by different states in the Markov Process E_t = {(wsg,sqlg), (wsa, sqlg), (wsf, sqlg)} states. E_a={(wsf, sqlg)} is considered as an absorbing state. When the web application systems move to the security-failed state

and the active security breach in software systems exists, this security failure state actually becomes an irreducible set in the Markov Process Model [97].

For the Markov Process, we calculated the mean time to security failure as follows:

As discussed in the method [26], [80], [94], [97], we have Markov matrix P for one-step transition probability in the Markov chain. P can be reorganized as P'

P' =	1				
		1			
			1		
				…	
	b_1	b_2	b_3	…	Q

where b_k is a sub-matrix with the one-step probability of describing transient state i to irreducible set. Sub-matrix Q from the Markov matrix P represents the transition probabilities between the transient states in one-step transition. The mean time to security failure can be calculated using the following operation:

$$N(i,j) = (I - Q)^{-1}(i,j). \quad (9)$$

where N(1,j) is the average number of times the state j ($j \in E_t$) is visited in the Markov chain before the Markov chain arrives at one of the absorbing states from the beginning state.

When we obtain the mean sojourn time in state j (T_j), the MTTSF can be computed by:

$$MTTSF = \sum_{j \in E_t} N_{1j} T_j . \quad (10)$$

5.4 Vulnerability Risk Analyses

In the previous sections, I developed a security analysis of availability and mean time to security failure. However, the root cause of system security failure is software vulnerability. Using existing vulnerabilities, I will develop vulnerability based security risk assessment to quantify software systems.

In the security risk analysis, the security risk is positively related to the threat, vulnerability, and the expected loss from the vulnerability according to equation (4). In the security risk model, the expected loss is related to the protected asset. The protected assets at hardware level may include storage and communication devices. At software level, the protected assets may include data storage system, utility programs, operating systems, and other applications [105]. There are several common attack threats for software systems. These are spoofing, tampering, repudiation, information disclosure, denial of service, and elevated access right, etc [95]. Attackers in spoofing claim to be some other identities. Whereas attackers in tampering try to revise information while it is in a travel or stationary position. Furthermore attackers in repudiation conduct some actions that are difficult to identify. While attackers in information disclosure also try to obtain data by stealing. Meanwhile attackers in denial of service prevent systems from normal operations. Finally attackers in elevated access right conduct some unauthorized activities in systems [95].

The Web Application Security Consortium (2004) has classified web threat into several categories [9], [99]. The threats include authentication, authorization, client-side attacks, command execution, information disclosure, and logical attacks. In authentication, threat may come from insufficient authentication and weak password

validation. For insufficient authentications in the Apache web server, there is a directory /admin that is similar to the root directory. If hackers get into this directory, they can and will look through all the other directories. Weak passwords are passwords shorter than six characters. The passwords would not be changed for a long period of time, and sometimes, the passwords repeated the old password. In the Internet era, the users may come from local or remote websites from the host country to the other countries. Authentication in a distributed web-system presents a big challenge for user identity validation. In authorization, authorization threat may come from credential prediction in controlling web users. Authorization threat may come from insufficient session expiration in using previous session credentials for authorization in the web-based applications. These two categories of threats now pose a high risk for information security since these two categories decide the access control. In command execution, a threat may come from buffer overflow, format string attack in using string library to access the physical memory, operating system commands in controlling application input parameters, and SQL injections in the web-based applications. For example, you may define a char buffer [11] in your program written in C. Then, you may input some strings as char data [] = argv[0] from command line. Finally, you use strcpy (buffer, data) to copy the input data. If your argv[0] holds a long string, it may cause allocated space with buffer overflow. In information disclosure, threats may come from unauthorized information access from other directories and path traversal. Local attacks may come from misused functions in access control, denial of service in causing a website from normal operations, and limited process validation in the intended traffic. All these threats

are highly likely to happen in the web-based applications depending on the software system quality, system configurations, user behaviors, and many other related factors.

Ravenel (2006) proposed that annual loss expectancy can be computed according to the threat in different phases in the software lifecycle [76]. Vulnerability severity level can be used to determine the vulnerability risk value in equation (5). These numbers can be used to calculate security risks for different organizations. Hickman (2004) at SoftSource Consulting proposed a threat ranking scheme for application security [32]. In his approach, threats are evaluated according to several risk factors: potential damage loss, re-occurrence, exploitability, impacted users, and degree of discovery. Each category is graded from 1 to 10. Then, an average of all categories is the threat score. For example, if we have a threat from obtaining a root access right, the scores for each category are: 10 in potential damage loss, 10 in re-occurrence, 9 in exploitability, 9 in impacted users, and 10 in degree of discovery. So, the threat scores 9.6 on average. I will modify Hickman's approach for risk evaluation in fuzzy logical method and information entropy-weight coefficients to make a better measurement for the web-based applications because the proposed score systems are very subjective. It may only work in a limited environment.

5.5 Risk Analysis

Risk index will be calculated using threat and vulnerability severity level. Threat will be evaluated by Hickman's approach [32]. Web software vulnerability will be evaluated using five different levels proposed by Grossman [30]. The risk index will be

calculated using fuzzy logic method developed by Zhao et. al [109]. In the following, I will synthesize the risk evaluations in incorporating all different pieces of the works.

Using the approach described by Zhao et. al. (2005), the risk factors in the web-based applications can be described as $H = \{H_1, H_2, H_3, H_4, H_5\}$, where H_1, H_2, H_3, H_4, and H_5 stand for potential damage loss, re-occurrence, exploitability, impacted users, and degree of discovery respectively [109]. The security experts of web-based applications will give the evaluations of R matrix based on the five risk factors and evaluation rules in fuzzy map. The fuzzy map FZ: H-> FZ(V), where FZ(V) is the fuzzy set on V. H_i -> $FZ(H_i) = (r_{i1}, r_{i2}, \ldots r_{im})$. The risk evaluation rule set can be expressed as $V = \{v_1, v_2, v_3, \ldots v_m\}$. The R matrix indicates the contribution from the risk factor H_i to the criteria in the evaluation set V. The R matrix can be expressed as $\{r_{i1}, r_{i2}, r_{i3}, \ldots, r_{im}\}$, where i = 1, 2, 3, 4, 5; m=1, 2, 3, 4, 5, 6, 7.

$$R = \begin{matrix} r_{11}\ r_{12}\ \ldots r_{1m} \\ r_{21}\ r_{22} \ldots r_{2m} \\ \ldots\ldots\ldots\ldots \\ r_{51}\ r_{52} \ldots r_{5m}. \end{matrix} \qquad (11)$$

To compute the frequency of the risk factors, the weight vector in A will be assigned to each risk factor. The weight vector comes from the expert estimation in the web-based applications. Thus,

$$A = (a_1, a_2, \ldots, a_5) \qquad (12)$$

The weight set for the evaluation set V is defined in K vector. For different web-based applications, the weight for the evaluation set V varies. So,

$K = (k_1, k_2, \ldots, k_7)$ (13)

The risk from vulnerability i in the web-based applications can be calculated using the following equation (14):

$Risk_i = A*R*K^T$ (14)

where K^T is the transposition matrix of K. The weight vector A is produced by the security experts in the web-based applications. Matrix R is the frequency of each risk factor H_i for the evaluation set V in the web-based applications. To overcome subjective evaluation, the entropy-weight coefficient will be calculated in the following equations.

To overcome subjective judgment [109], the relative importance of a risk factor is measured by:

$$H_i = -\sum_{j=1}^{m} r_{ij} \mathrm{Ln}(r_{ij}) \quad (15)$$

where the bigger the H_i value is, the bigger the contribution from the risk factors to the web-based application system is. When r_{ij} values are 1/m (j = 1,2,…,m), the H_i becomes the maximum with the value of H_{max} in Ln(m). In the web-based applications, i = 5 and m = 7. The entropy of the risk factor in importance can be computed in equation (16).

$$e_i = -\frac{1}{\mathrm{Ln}(m)} \sum_{j=1}^{m} r_{ij} \mathrm{Ln}(r_{ij}) \quad (16)$$

when r_{ij} values are all the same, entropy e_i will be with the maximum value of 1. So, e_i has a property: $0 \leq e_i \leq 1$. When an entropy is at the highest, each risk factor contributes the least to the system risk assessment in the web-based applications.

Uniform 1- e_i, the adjusted weight of a risk factor can be measured:

$$\lambda_i = \frac{1}{n - E}(1 - e_i) \quad (17)$$

where $E = \sum_{i=1}^{n} e_i$ (18)

λ_i satisfies: $0 \le \lambda_i \le 1$, and $\sum_{i}^{n} \lambda_i = 1$.

In our web based applications, there are three components: web clients, web servers, and database servers. According to the risk theory developed by Koller (1999), the chance of failure (COF) is the rate from a risk category to cause a system to a failure [46]. When we treat multiple vulnerabilities in a subsystem as the multiple component risk factors in the subsystem, the total chance of success (TCOS) for a risk assessment with multiple vulnerabilities in one subsystem in the web-based applications can be calculated by:

$TCOS_m$ = (1 – COF_vuln*1*) x (1- COF_vuln*2*) x…x (1 – COF_vuln*i*),

(19)

where vuln*i* represents *i*th vulnerability in a subsystem.

In our web-based security term, total chance of successful security (TCOS) for web-based application can be defined as:

TCOS = (1- web_client_security_failed) x (1-web_server_security_failed) x (1-database_server_sceurity_failed).

(20)

I will use all the equations above in the following vulnerability based risk assessment. According to TCOS index, we will classify TCOS into three levels to indicate the web application system risk: low, high, and failed level. In statistical terms, we define a quartile of security index range as low and failed risk level. The high risk

security level accounts for two quartiles of the index range. Thus, the threshold index for low system security is defined as: 25 percent of TCOS index distribution (TCOS index from 0 to 0.25), physically it means that the web-based application system is at failed level in this index range; high risk level with 26-75 percent of security index range (TCOS index from 0.26 to 0.75), and top 25 percent of TCOS index is defined as high system security level (TCOS index from 0.76 to 1.0). This means that the web-based application system is at a very low risk level.

5.6 Case Study

5.6.1 Security Analysis

Several web client access errors were recorded in the logs during the data collection time. The web client access errors are outlined in TABLE 3. In the table, I only present the major error types; the most frequently observed error is “File does not exist.” The web-based applications move the published materials very often. The second most frequently observed error is “Directory index forbid by rule,” an error related to the user violation to security policy when the web clients access the materials in the web server. The next most frequently observed error is “Script not found or unable to start.” This error is related to security policy in access privilege or program execution privilege. In the following paragraphs, I will present some more security issues observed in open web proxy honeypot in details.

TABLE 3
Major Error Summaries from the Error Logs [97]

Major Error Type	Number of Errors (each type)
File does not exist	590
Directory index forbid by rule	214
Script not found or unable to start	70
Attempt to invoke directory as script	17
Premature end of script heads	21

From the analysis, the top ten attacker IP addresses are listed in TABLE 4. In TABLE 4, the top attacker IP address owners and locations are obtained using http://www.dnsstuff.com/tools/whois. The Internet that waved most attacks to the open honeypot was from IP address in 64.62.145.98. The IP address owner is Energy Group Inc. in Southeastern, PA. The next top attacker IP address is 210.118.169.20, located in Seoul, South Korea. From the IP address owners, we can see that 6 of the top 10 were from other countries (China, South Korea, and Romania). 4 of the top 10 attacker IP

TABLE 4
Top Ten Attacker IP Addresses

Attacker IP address	IP address owner (location)	Number of attacks
64.62.145.98	Energy Group, Inc. (Southeastern, PA)	414
210.118.169.20	Shinbiro-IDC (Seoul, Korea)	410
210.116.59.164	KRNIC (Seoul, Korea)	184
4.152.207.238	Level 3 Communications (Spartanburg, SC)	94
210.51.12.238	Tongtai IDC of China Netcom (Beijing, China)	92
64.122.238.114	Integra Telecom (Portland, OR)	91
220.170.88.36	Hunan Telecom (Hengyang, China)	63
81.181.146.13	SC Mediasat SA (Romania)	55
222.95.35.200	Jiangsu Province Network (China)	51
4.152.207.126	Level 3 Communication (Spartanburg, SC)	46

TABLE 5
Top Attack Targets Are Listed [97]

Requests	URL	Comments
587	/_vti_bin/_vti_aut/fp30reg.dll	Proxy authentication
180	/sumthin	Requested material not found
97	http://www.yahoo.com	Request forbidden materials
93	//cgi/awstats.pl?configdir=|%20id%20|	Request materials not found
73	/scripts/..%255c%255c../winnt/system32/cmd.exe?	Nimda Worm
71	//cgi-bin/awstats/awstats.pl?configdir=|%20id%20|	Request materials not found
68	//cgi-bin/awstats.pl?configdir=|%20id%20|	Request materials not found

addresses were from Pennsylvania, South Carolina, and Oregon in the United States. The top attacker targets are listed in TABLE 5.

From the data for the top attacked targets [97], I observed that there are many attacks recorded in the honeypot logs. The major security problems were computer worms (CODE RED and NIMDA), AWSTATS attack, unauthorized access request (HTTP 1.0/1.1 error code 403), unidentified request method (HTTP error code 501), not allowed http request method (error code 405), non-http compliant requests, denial of service attack from Internet Relay Chat (IRC port 6666 and 6678), MS-SQL Worm propagation, MS-SQL version overflow, etc.

In TABLE 4, the Nimda was observed as a major worm that spreads during the data collection time, for it is a self-spreading virus. It regularly attacks the Microsoft IIS server and outlook users. The Nimda attacks Microsoft II server using the IIS file permission, characters decoding execution, and unicode directory traversal vulnerability, but it can also attack Microsoft Outlook users using readme.exe file in an attached email. The Nimda can detect the vulnerability in Internet Explorer. The vulnerability fix for the Nimda worm is available in Microsoft website [72], [97].

The Code Red is another worm that attacks the IIS server resulting in buffer overflow. It is reported that the worm activity on an infected machine is time-related to the machine clock [95], [97]. In TABLE 4, proxy authentication is the most severe attack observed. AWSTATS attack is using remote command execution. Attackers make good use of configuration directory to execute arbitrary commands prefixed with "|" character. For example, //cgi-bin/awstats/awstats.pl?configdir=|%20id%20| was observed in the used data. Unauthorized access request (http error code: 403) means that the web server

knew the request. The request did not pass the audit process. The remote request should be forbidden. Unidentified request method (http error code 501) means that the web server does not implement the request. The web server cannot allocate any resource for the request since the web server can not identify the requests. Not allowed http request method (http error code 405) means that the requested materials do not use the standardized http methods as specified by the protocol. Non-http compliant requests mean that the http request from the web clients does not use http (hyper-text transport protocol) standard format. For example, get/www.utexas.edu http/1.0 is a standard request format in RFC2616 [8]. Attacks from IRC connections are observed from the open web proxy as a system operator in the channel. MS-SQL worm is a slammer worm that attacks database systems. There were a lot of attacks by MS-SQL slammer worm during the data collection period of time. All these attack activities impacted the web-based application operations, whether earlier or later.

As discussed before, the Code Red is a malicious worm that can self-propagate over the Internet. The buffer overflow in Microsoft IIS server on an infected machine is time-sensitive. The Code Red attack examples are presented in TABLE 6.

TABLE 6
The Code Red Requests Are Exampled [97]

y0w4000@sun(~/SotM34/http)>grep default.ida access_log.* | less
access_log.1:63.102.226.241 - - [07/Mar/2005:02:28:35 -0500] "GET
/default.ida?XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXX%u9090%u6858%ucbd3%u7801%u909
0%u6858%ucbd3%u7801%u9090%u6858%ucbd3%u7801%u9090%u9090
%u8190%u00c3%u0003%u8b00%u531b%u53ff%u0078%u0000%u00=a
HTTP/1.0" 404 1061 "-" "-"access_log.1:63.226.106.228 - -
[07/Mar/2005:05:46:04 -0500] "GET
/default.ida?XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXX%u9090%u6858%ucbd3%u7801%u909
0%u6858%ucbd3%u7801%u9090%u6858%ucbd3%u7801%u9090%u9090
%u8190%u00c3%u0003%u8b00%u531b%u53ff%u0078%u0000%u00=a
HTTP/1.0" 404 1061 "-" "-" access_log.1:63.226.106.228 - -
[07/Mar/2005:14:08:12 -0500] "GET
(CUT remaining parts here)

In the above, I conducted a brief security analysis in open web proxy honeypot as security analysis background. From the analysis, it is obvious that some software

vulnerabilities did exist in the analyzed systems since we observed different computer attacks, Code Red, NIMDA worm spread during data collection time. Computer hackers did use these vulnerabilities to break into the open web proxy honeypot. All these security issues are caused by either source code defects or system configurations. Because software vulnerabilities exist, the web-based application system may not provide reliable services all the time as intended. In the following, I will estimate access availability and mean time to security failure using Markov Process Model.

From the data collected, the initial transition probabilities are estimated using access transaction rate between states in the Markov matrix P [97]. To compute mean time to security failure, the number of visits in each state and mean sojourn time in each state are required. The number of visits in each state can be calculated from state transition probability. Mean sojourn time in each state can be estimated as follows: Mean sojourn times: previous study (186 software systems) indicates software systems, from release in operation to a vulnerability fix, need about 242 days [6]. Let us have the Markov Process that spends 0.25 unit time in (wsg, sqlg), 0.50 time unit in (wsa, sqla) state (the attacked state is combination of web_server_vulnerable and web_server_attack), 0.25 unit time in (wsf, sqlg) state. For security failure from the attacks on the web server vulnerability only, the time to security failure experienced the state transitions in E_t = {(wsg, sqlg), (wsa, sqlg)}, the mean sojourn time in these states are expressed as $\{T_{(wsg,\ sqlg)},\ T_{(wsa,sqlg)}\}$.

Using the following equations,

$$\sum_i \pi_i P_{ij} = \pi_j \qquad (21)$$

$$\sum_{j} \pi_j = 1 \tag{22}$$

and

$$p_j = \frac{\pi_j T_j}{\sum_{k \in E} \pi_k T_k} \tag{23}$$

Using equation (21) and (22), the steady-state probabilities in the embedded Markov Chain can be obtained:

$\pi_{(wsg,sqlg)} = 0.3766$, $\pi_{(wsg,\ sqla)} = 0.2561$, $\pi_{(wsg,sqlf)} = 0.0866$,

$\pi_{(wsa,\ sqlg)} = 0.1648$, $\pi_{(wsa,\ sqla)} = 0.0844$, $\pi_{(wsa,\ sqlf)} = 0.003$,

$\pi_{(wsf,\ sqlg)} = 0.0245$, $\pi_{(wsf,\ sqla)} = 0.0126$.

Using the relationship equation (23), the steady-state probabilities in the Markov Process can be obtained:

$\pi_{(wsg,sqlg)} = 0.3487$, $\pi_{(wsg,\ sqla)} = 0.2454$, $\pi_{(wsg,sqlf)} = 0.062$,

$\pi_{(wsa,\ sqlg)} = 0.1530$, $\pi_{(wsa,\ sqla)} = 0.1557$, $\pi_{(wsa,\ sqlf)} = 0.0277$,

$\pi_{(wsf,\ sqlg)} = 0.0220$, $\pi_{(wsf,\ sqla)} = 0.0113$.

The Immediate Access Availability Analysis

From equation (10), we can modify the equation since we assume the web client is a reliable part in the system and vulnerability only exists in the web server, So,

$Av = 1 - p_{(wsf,\ sqlg)}$

Apply the number we got from the steady-state probabilities in the Markov Process Model, and the availability is:

Av = 1-0.022

= 0.978

So, based on the data collected, the web-based application system has 97.8 % time to provide the intended services reliably. Beyond this rate, the whole system cannot provide reliable service because of the web server security failure.

Mean Time to Security Failure (MTTSF):

$N(i,j) = (I-Q)^{-1}$ =		(wsg, sqlg)	(wsf, sqlg)
	(wsg, sqlg)	1.17	0.28
	(wsa, sqlg)	0.57	1.17

MTTSF = 0.4325 time units = 104.7 days, where time units = 242 days.
This equation means that the web-based application may experience security failure after 104.7 days in open web proxy honeypot because of multiple attacks on the web server.

Validating Mean Time to Security Failure by Simulation

In this sub-section, I will change the probability from attacked state to security failure state in the web server to validate the correctness of mean time to security failure using Markov Process Model. The security failure probability increase in the web-server means that there are more attacks on the vulnerabilities in the web-based applications. When security failure probability decreases, this decrease means that there are fewer

attacks on the vulnerabilities in the web-based applications. When the security failure probability changes, the mean time to security failure is as follows:

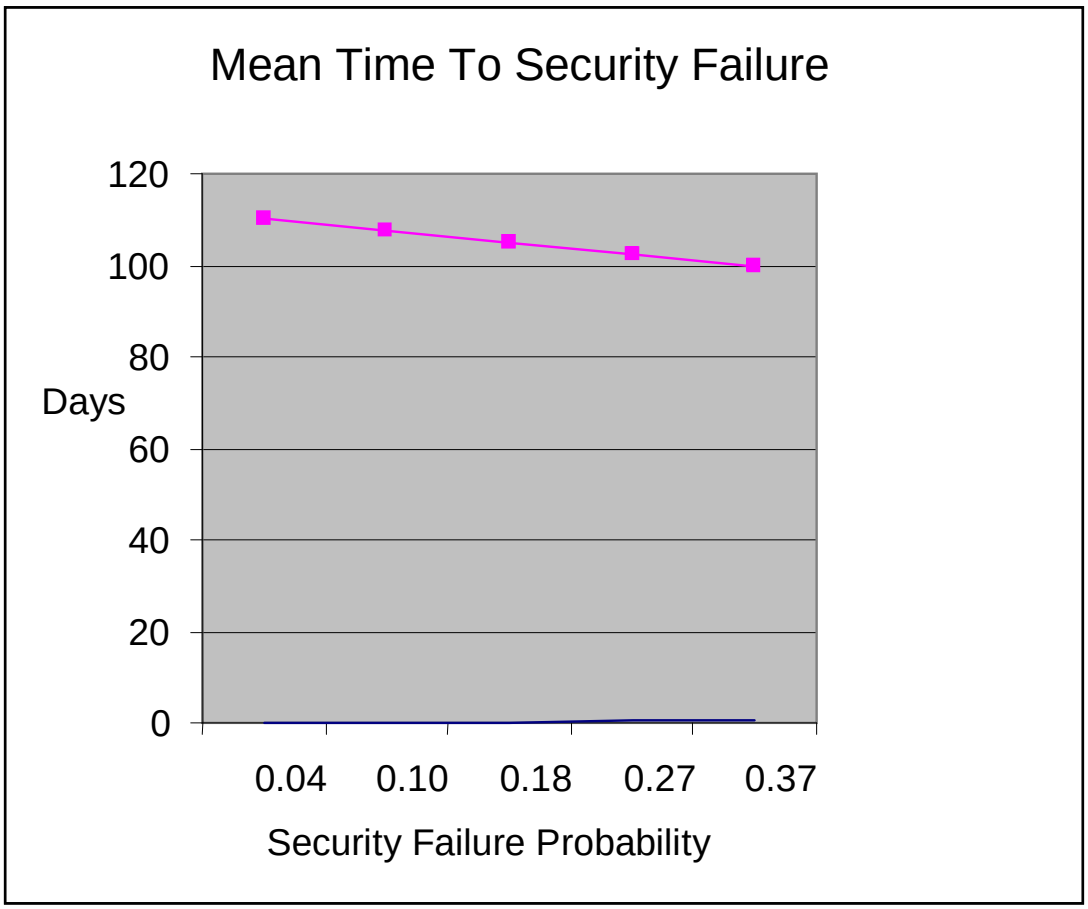

Fig. 3. Mean time to security failure decreases as the failure probability increases in the web server.

From the simulation, it is clear that the web server can provide trusted services for longer periods of time while the system has fewer attacks on the vulnerabilities and low security failure rate. When the web-based application systems have more attacks on the vulnerabilities and higher security failure rate, mean time to security failure is reduced also. Conceptually, the Markov process model can calculate the mean time to security failure correctly. Please see Fig. 3 for details.

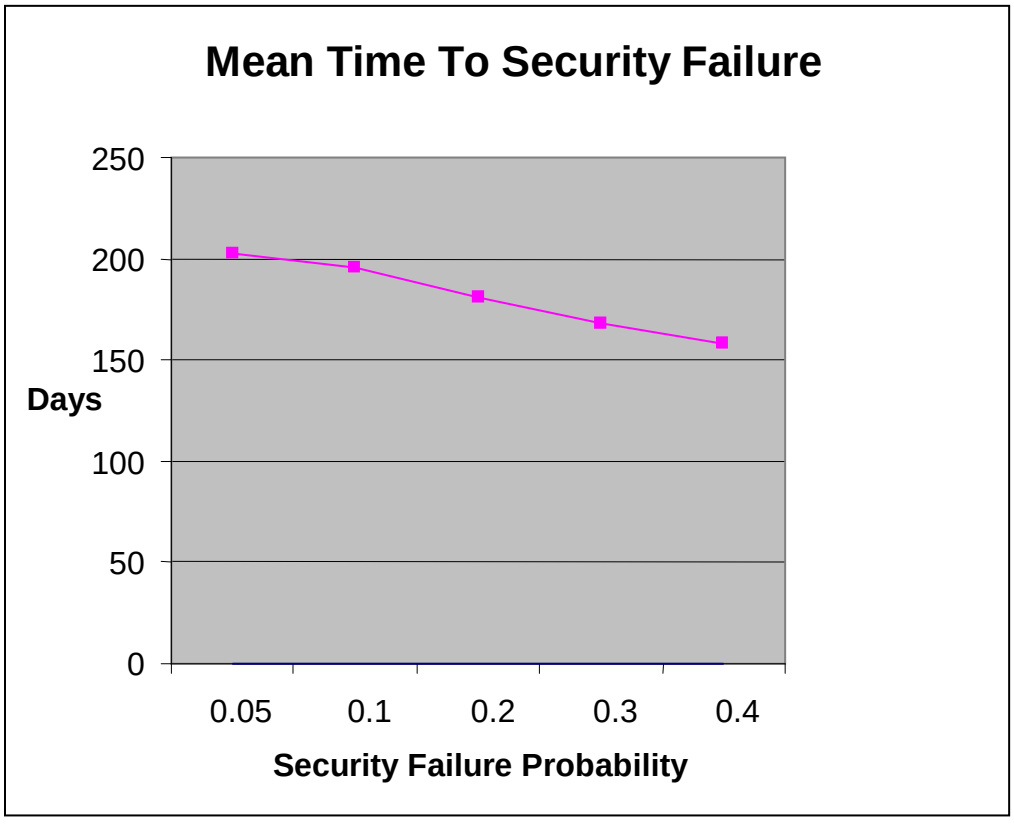

Fig. 4. Mean time to security failure is reduced as failure probability increases in the database server.

When the security failure probability is changed for the database server, mean time to security failure is changed, similar to Figure 4. From the simulation in Fig. 4, we can see the mean time security failure in the database server will decrease when there are more attacks on the vulnerabilities. When the systems have fewer attacks on the vulnerabilities and low security failure rate, the web-based applications can provide longer trusted service. Therefore, Markov Process Model can calculate mean time to security failure correctly. In a similar way, I can validate the immediate access availability correctly by simulation.

5.6.2 Security Risk Analysis

In the previous section, I present the case study for security measurement in access availability and mean time to security failure. However, the cause of security failure is due to the vulnerabilities. In this section, I will present a vulnerability based risk assessment for the web-based applications.

From https://cirdb.ceris.purdue.edu/ccopvdb/public, we found that six well-known vulnerabilities were reported from July 1, 2006 to September 1, 2006. We can assume that all the vulnerabilities appear in our web-based application system during that time period. One vulnerability was reported for the database server with CVE number in 2006-4041. According to http://www.securityfocus.com (2006), this is a SQL injection vulnerability when using a Postgres database server. Attackers can use this vulnerability remotely to run an arbitrary SQL command. As we discussed before, the SQL injection vulnerability can pose a high threat to the web-based application systems. Three vulnerabilities were reported for the web server with the CVE numbers in 2006-4089,

2006-3921, and 2005-3620 respectively. The CVE 2006-4089 is about multiple buffer overflows in AlsaPlayer 0.99.76. This vulnerability allows remote users to launch attacks in denial of service on a web server. AlsaPlayer is a MP3 and audio player. When the web server performs denial of services, basically the web server loses its intended functions. The CVE 2006-3921 is about the Java system application server. The vulnerability allows remote authenticated users to read files outside of the intended directory. The directory traversal can mean high security breaches for sensitive data. The CVE 2005-3620 is about interface vulnerabilities for VMware server. The VMware server integrates computer processor, memory, and other hardware into several virtual machines to optimize their utilization. This vulnerability is directly related to computer system resource allocation and management; thus, it is directly related to system performance. Two vulnerabilities were reported for the web clients with CVE numbers in 2006-3918 and 2006-3574 respectively. The CVE 2006-3918 is about the IBM HTTP server 6.0 and the Apache HTTP server 1.3 which is not clear in defining the expected header from an HTTP request. This vulnerability is related to the denied web access requests. In previous security analysis, the data showed that the denied http request header can result in the information request failure. The CVE 2006-3574 is about

TABLE 7
The Judge Set Constructed by Probability [109]

V1	V2	V3	V4	V5	V6	V7
Ignorable	Very low	Low	Medium	High	Very High	Extreme

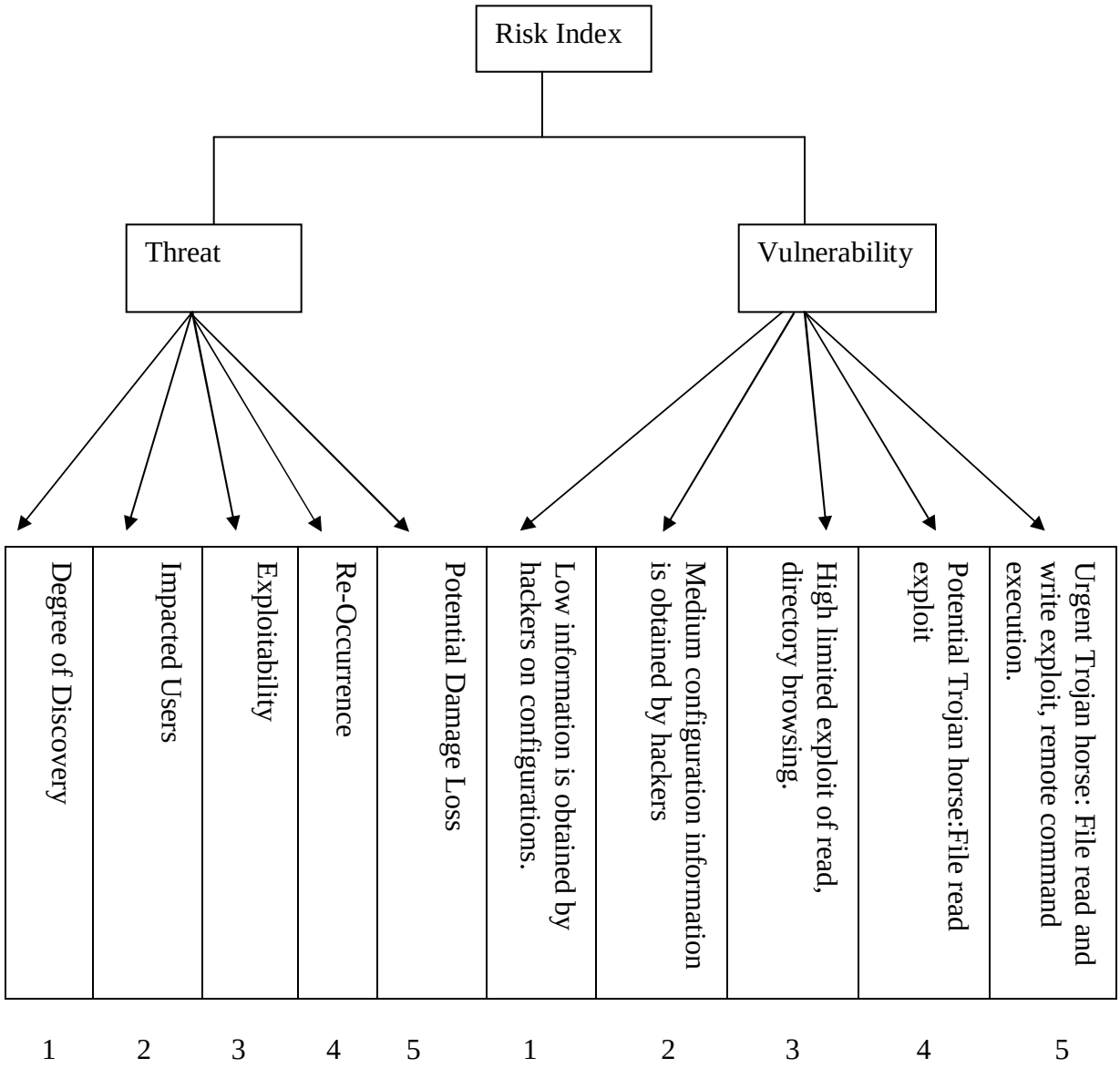

Fig. 5. The synthesized risk assessment scheme (developed based on [30], [32], [62], [109]).

multiple cross-site scripting (XSS) vulnerabilities in a web client [75]. The cross-site scripting is a special defect to the web-based applications. The vulnerability exposes the

user data to the vulnerable web server in the cookies, which means it can be accessed by other unauthorized sources. Therefore, this vulnerability poses a high risk to the users [35]. Let us construct the fuzzy set H = {H_1, H_2,…,H_5}, where H_1, H_2, H_3, H_4, and H_5 represents “potential damage loss”, “re-occurence”, “exploitability”, “impacted users”, and “degree of discovery”. For the rule of risk factor, judge set S to set H is defined as S = {S_1, S_2,…, S_7}, which indicates the risk level in TABLE 7.

Analytic hierarchy process (AHP) uses many rules to estimate the values from the variables which are difficult to define [109]. For the risk analysis, it can be expressed as follows in Fig. 5. In Fig. 5, threat is regarding computer hackers with the intent and ability to exploit a vulnerability in a system. Vulnerability is weak point in a system that can be exploited.

Risk Analysis for the Database Server

For the database server, only one vulnerability is reported in December 2006 according to the access. The experts in the web-based applications make the probability estimate for the risk set H. The probability of each risk factor is decided by a group of security experts. According to the assessments from the security experts (graduate students in computer science at TAMU), the subjective matrix R_p is:

	0.0	0.3	0.2	0.1	0.2	0.2	0.0
	0.1	0.2	0.3	0.2	0.1	0.1	0.0
R_p=	0.0	0.1	0.2	0.2	0.2	0.3	0.0
	0.0	0.2	0.2	0.2	0.2	0.2	0.0
	0.0	0.1	0.2	0.3	0.2	0.1	0.1

From equation (13), we can get e_i from R_p as (0.8002, 0.8714, 0.8002, 0.8271, 0.8714). Then the weight vector of each risk factor can be calculated in equation (16):

$\Lambda_p = (\psi 1, \psi 2, \psi 3, \psi 4, \psi 5) = (0.2408, 0.1550, 0.2408, 0.2080, 0.1550)$.

The weight for the judge rule in (V1, V2, V3, V4, V5, V6, V7) is estimated by the security experts as (1/10, 1/10, 2/10, 1/10, 1/10, 2/10, 2/10) in B_p, and the risk events can be calculated in equation (13) as:

$P_v = \Lambda_p R_p B_p^T = (0.155, 0.1844, 0.2656, 0.1913, 0.1875, 0.193, 0.0155) * B_p^T$

$= 0.1651$

where, Λ_p is 1 x 5 matrix, R_p is 5 x 7 matrix, and B_p^T is 7 x 1 matrix.

For the risk from this vulnerability in the database server, the judge set of the risk factor set V = {V1, V2, V3, V4, V5, V6, V7) which shows threat severity level as in TABLE 12.

The computer security experts (graduate students in computer Science) make a risk assessment of the threat from the vulnerability R_T as follows:

	0.0	0.0	0.1	0.2	0.2	0.2	0.3
	0.0	0.0	0.1	0.2	0.2	0.3	0.2
R_T =	0.0	0.1	0.2	0.2	0.2	0.3	0.0
	0.0	0.2	0.2	0.2	0.2	0.2	0.0
	0.0	0.0	0.0	0.2	0.2	0.3	0.3

From equation (15), we can get e_i as: (0.8002, 0.8002, 0.8002, 0.8271, 0.7021). The weight factor of each risk factor can be calculated by equation (16). The result is as follows:

Λ_i = (0.1867, 0.1867, 0.1867, 0.1616, 0.2784).

The weight for the judge rule in (V1, V2, V3, V4, V5, V6, V7) is estimated by the security experts as (0, 0, 0, 1/10, 2/10, 3/10, 4/10) in B_i, and the risk event can be calculated in equation (13) as:

$$P_T = \Lambda_i R_T B_i^T = (0, 0.051, 0.107, 0.2, 0.2, 0.6613, 0.1769) * B_i^T$$

$$= 0.3291$$

where, Λ_i is 1 x 5 matrix, R_T is 5 x 7 matrix, and B_i^T is 7 x 1 matrix.

Risk index for database server = $1-(1-R_{threat}) * (1-R_{vulnerability})$

$$= R_{threat} + R_{vulnerability} - R_{vulnerability} * R_{threat}$$

$$= 0.4399$$

Risk Analysis in the Web Server

For the web server, there are three vulnerabilities existing in the web server during the modeling time. We can treat the three vulnerabilities in the three different components of one system in equation (18).

For vulnerability 1 with the buffer overflow (CVE2006-4089), R_T from the expert evaluations is as follows:

	0.0	0.0	0.0	0.2	0.2	0.3	0.3
	0.0	0.0	0.1	0.1	0.3	0.3	0.2
R_T =	0.0	0.0	0.0	0.2	0.2	0.3	0.3
	0.0	0.0	0.2	0.2	0.2	0.2	0.2
	0.0	0.0	0.1	0.2	0.2	0.3	0.2

In the method described above, e_i = {0.7021, 0.7733, 0.7021, 0.8271, 0.8002). Λ_i = (0, 0, 0.3228, 0.6836, 0.8383, 1.0587, 0.9014). The weight for the judge rule is as follows in B_i is (0, 0, 0.1, 0.2, 0.2, 0.2, 0.3)

The risk index from the vulnerability is:

$= \Lambda_i * R_T * B_i^T$

= 0.3188

In a similar way, the threat index obtained from the vulnerability 1 is 0.3235. According to equation (5),

Risk1 from the vulnerability 1 in the web server is = $1-(1-R_{threat}) * (1-R_{vulnerability})$

$= R_{threat} + R_{vulnerability} - R_{vulnerability} * R_{threat}$

= 0.3188 + 0.3235 – 0.3188*0.3235

= 0.439.

For vulnerability 2 in the web server, the vulnerability is about remote authenticated users and their access to other documents. We can get the risk for this vulnerability.

Risk2 = 0.7016

For vulnerability 3 in web server, the vulnerability is about denial of service from the web server for interface of the VMware server. Using the method discussed above, we can get risk index as:

Risk3 = 0.5012

Since we can treat the three vulnerabilities in different places in the web server, Total chance of success (TCOS) = (1-risk1)*(1-risk2)*(1-risk3)

= (1-0.439)*(1-0.7016)*(1-0.5002)

= 0.3257 * 0.2984 * 0.5

= 0.0841

Risk in the web server is 1- TCOS = 0.9159.

Risk Analysis in the Web Client

For the web client, there are two vulnerabilities. The first vulnerability is about the unexpected header from the HTTP server, while the second is about the cross-site scripts vulnerabilities. In a way similar to the one described above,

Risk1 for vulnerability1 = 0.5421

Risk2 for vulnerability2 = 0.2348

TCOS for the web client = (1-risk1)*(1-risk2)

= (1-0.5421)*(1-0.2348)

=0.4579 * 0.7652 = 0.3504

Risk in web client = 1-TCOS = 1-0.3504 = 0.6496

Risk Classification

From the specifications we define, system security risk is at failed level if the TCOS index is from 0 to 0.25, at a high risk level if the TCOS index is from 0.26 to 0.75, and at a low risk level if the TCOS index from 0.76 to 1.0. For web based applications, we have risk measurement as:

Total chance of success = $(1\text{-}Risk_{web\_client})*(1\text{-}Risk_{web\_server})*(1\text{-}Risk_{database\_server})$

= (1-0.440) * (1- 0.916) * (1- 0.6496)

= 0.56 * 0.084* 0.3504

= 0.1646

Because TCOS is 0.1646, the system security is very low. In other words, the system risk is very high, and this index indicates that the web-based application system is classified as a failed risk level. From the partial result, we also can see the web server has high-secure failure rate. If the web server has secure failure, the web client request cannot finish the process request, whether the database server works or not.

5.7 Summary

In this chapter, I have conducted security analysis and vulnerability based risk assessment for the web-based applications. The security issues and system risk both originated from the software vulnerabilities. Because software vulnerabilities exist in web-based software systems, vulnerabilities open the door for attackers to hack the web-based applications. The attacks on the web-based applications may cause system security concerns by providing immediate access to services and possibly creating security failures in those services. The attacks on the web-based applications also inject security risk in the software systems. Immediate access availability and mean time to security failure can quantitatively measure software system security. In this chapter, I adapt Markov Process Model to compute these two parameters. The correctness of mean time to security failure is validated by simulation. System risk assessment also provides an important parameter to measure system security related reliability.

Several different approaches have been proposed to improve software system security. In the next chapter, I will present a popular approach in access control to improve software system security.

CHAPTER VI

SECURITY IMPROVEMENT MODEL*

A few models have been suggested to address web-based application security. These include coalition-based access control (CBAC) [23], discretionary access control (DAC), mandatory access control (MAC), and role-based access control (RBAC) [21], [42]. Coalition-based access control supports a variety of functionality, expressiveness, and flexibility for resource access specifications. The CBAC model integrates team-based (TMAC) and task-based (TBAC) access control together to define resource access policies. The discretionary access control specifies authorization rules in the subjects and objects. Subjects can appear to be a user, a group, and a process identifier in the web-based applications. If a subject has control on an object, the subject can decide access rights to other requested subjects. Mandatory access control (MAC) specifically addresses the information security in confidentiality and integrity. In MAC model, subjects and objects are arranged in different levels and used in access decisions. For example, to improve information confidentiality in battle fields, MAC model may adapt a multilevel security mechanism to enhance its function. Multi-level information classification in the web-based applications can separate different types of information subscribers. Role-based access control is popular for different organizations because of its flexibility. The role-based access control can be defined according to the specifications in different organizations. Role-based access control has become an

* This chapter is reprinted from "Enhanced enterprise web-based application security using GeoIP service" by Yong Wang and Dick Simmons 2006 in the Proceedings of the 10th IASTED SEA Conference with permission from IASTED.

alternative to MAC and DAC approaches [43]. Policy flexibility has made role-based access control more attractive in different applications.

As role-based access control draws increasing attention, Bertino et. al. (2001) developed a temporal role-based access control (TRBAC) [12]. Time can play a key role in time-sensitive access [44]. TRBAC applies a group of temporal conditions in role operation. In the temporal role-based access control, a role is examined when the requester obtains access permissions.

Recently, Atluri and Chun (2004) described an authorization model using geospatial data [7]. In their model, authorizations are defined in spatial and temporal attributes. Atluri and Chun thought that combining images and geospatial data would present more security threats than individual data itself. The access control model used a public geo-referenced profile to allocate user access to the different spatial data. This profile consists of property ownership and physical location information. When an access request is received, the computer system computes the specific area for authorization purposes. Several function operations are provided in the model. These include read, insert, delete, modify, and some other privileges for the different data.

Schmit, et. al. (2005) used the web service complexity to describe two authorization models for indirect and direct access [82]. Indrakanti, et al. (2005) revised Microsoft .NET MyServices utility to define different authorization policies in the health-care application domain [37]. The XML access control language was used in the extended authorization model.

Shen and Hong (2006) proposed an attribute-based access control (ABAC) for web services [85]. An authorization decision relied on attributes to decide access to a

resource. Attributes are a group of characteristics that describe an entity. An entity is a subject, resource, or environment that is associated with the users and applications. There are several types of attributes in a web-based application: Subject attributes, resource attributes, and environmental attributes. Subject attributes include users, company names, organization names, and membership, etc. Resource attributes include resource identity, location, space, etc. Environmental attributes include time, date, system configurtaions, and other parameters. Digital credentials are composed with the attributes. In the ABAC model, an authorization depends on the attributes of the associated entities. The ABAC does not give permissions to each requester before the request is submitted. The ABAC also applies automated trust negotiation (ATN) mechanisms to access control, specifically in the authentication process [85]. Peng and Wu (2006) developed an algorithm to handle secure communication and access control for web-based applications [70]. In the secure communication, they used a secure token to authenticate the user in data integrity and confidentiality in a SOAP message. In authorization, they defined an attribute-based role access control.

Previously, we have reviewed different access control models briefly. Authorization is a high level of access control for objects. The web pages, web applications, and etc. are some examples of the objects. Access control is for low-level objects. The objects include rows, tables, and documents in a data source [11], [33]. Hollar and Murphy (2006) classified the access control into four basic categories. These are user-based access control, role-based access control, attribute-based access control, and mandatory access control. The access model decides whether rights are granted or declined for a specific permission to a resource.

In the keys of a role-based access control (RBAC), RBAC for a system has a set of users, roles, resources, and access permissions. The roles contain the access rights. A user access to a resource is defined by a set of roles. The resource manager provides an access interface to the designated resources. A role-based access control model holds two security properties: Separation of duty and least privilege. Separation of duty means that the duty is assigned to different users. Least privilege means that a user is restricted by some roles according to the task requirements.

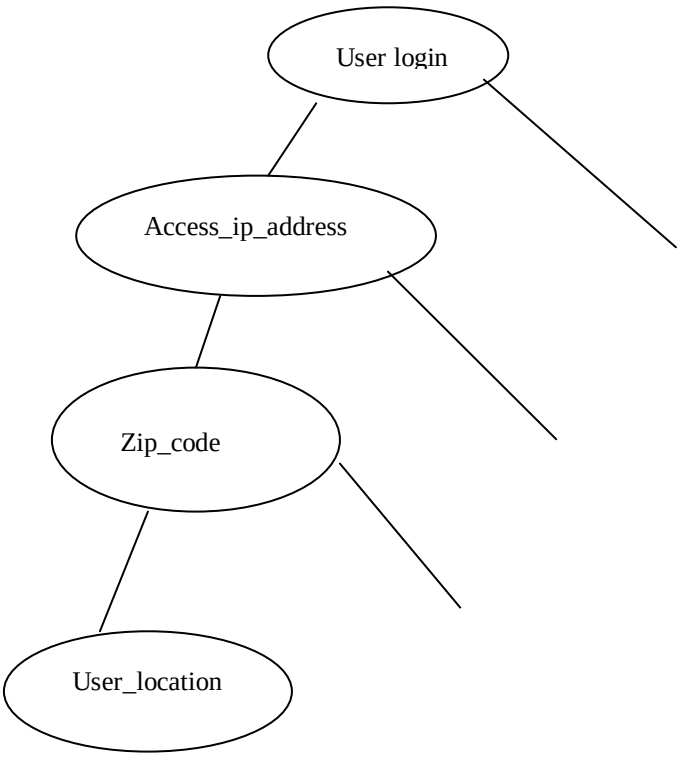

Fig. 6. The user credential type hierarchy (revised from Wang and Simmons 2006) [96].

6.1 Authorization Model

Authorization is an important approach for Internet computing. As the Internet becomes a major platform of computing, access authorization encounters a big challenge. Similar to the geospatial data authorization (Atluri and Chun 2004), the proposed model will adapt the subject credentials, object credentials, and access privilege for access control. Each subject credential normally contains several elements [7], [17], [40], [41], [42], [96]. The proposed model will use enhanced subject credentials in user information, access computer identity, and access location to combine with object credential and access privilege for authorization.

6.1.1 Authorization Subjects

One of the most important components for authorization is subject authorization. Subject authorization is about users, groups, organizations, etc. Credential types are expressed in the credential hierarchy tree [17]. Fig. 6 describes a user credential type in a hierarchy tree in the web-based applications. A credential type contains a set of attributes. In Fig. 6, an IP address and a zip code are the owner of the attributes of the credential type because of its unique characteristics. A subject may contain a set of credentials.

Definition 1 [Credential Type]. A credential type has an unique identifier in ct_id_i and a set of attributes in A_i corresponding to the credentials. Each attribute consists of an attribute name, attribute type, and an attribute mode. The attribute name is the identifier of the attribute. The attribute type defines the data type for the attribute. The attribute

model specifies the attribute in the credential type optional or required. The following is an example of credential–type.

For example,

(login_name, {(IP_address, string, required), (zip_code, integer, required), (city, string, optional), (state, string, optional), (country, string, required)})

In Fig. 5, the credential-type appears to be in a hierarchy tree. This type of credential type is called a credential type hierarchy.

Definition 2 [Subject Credential Expression] A subject credential can be defined as $\{c_1, c_2, \ldots\}$, where c_i is represented as pairs (ct_id, SC). ct_id describes the credential type unique identifier, SC is defined as $\{(at_1, k_1), (at_2, k_2)\ldots\}$. In the SC, at_i is an attribute name and k_i is its corresponding value for the attribute. A subject credential expression, for example, can be expressed as:

Tom1 = (user, {(name, “Tom Phillips”, (ip-address, “210.325.123.20”), (physical-location, “Houston, Texas, USA 75208”), (owner-period, [2005, now])}).

The proposed subject credential expression is composed of an user name, computer IP address, geospatial location information, and temporal attributes for the valid access time.

Definition 3 [Secure Credential Representation] If CT and SC are credential types and subject credential expressions for a subject identifier $\{sc_1, sc_2, sc_3, \ldots\}$, the credential expression operation is defined as:

If k Є CT and w Є SC, k(w) is still a credential expression. When multiple credential expressions have a logical operation, the new expressions still belong to the credential expressions.

For example,

Smith(w): is a credential expression to define a subject with a user name in Smith. Authorization will exam a credential user using the credential expressions to get user access right.

6.1.2 Authorization Objects

Object authorization is also very important. It will decide how much of the resources will be allocated to the access requesters. Requested objects in the web-based applications could be online digital materials, commercial product information, or satellite image data [1], [7], [96]. The information format may include text, image, and videos. Different kinds of information may have different access protections. For a community development information, it may be open access. For a product technical support material, the product introduction material may be available to all the users, the detailed support information may be only open access to the specifically subscribed users.

Access request on the web-based applications may be conceptually defined as the triple in (http_link_num, http_source_num, http_destination_num). An electronic document on the web can be represented in (id, slots, links, concepts), where id is the access object number. A slot is a slide window of information embedded in a document that can be identified in slot_name. When the slot can not be identified by a name, the object is not a named slot. Access authorization is assigned to a specific object with its identifier. For example, an object access request can be expressed as:
If concept (ro) = {miscellaneous fee}, then C(ro) = {miscellaneous fee, tuition, billing statement}.

Definition 4. (Conceptional Expression). A subset of concept expression still belongs to a concept expression. When two concept expressions have logical operations in union and/or intersection, the result is still concept expressions.

Definition 5. (Object Entity Identification). http_link-num and http_slot-num are a set of link identifiers and slot labels respectively.

6.1.3 Privileges Modes

Privilege modes will decide incoming user access rights in web-based applications. Proposed web access privilege modes include browsing, copying, and editing. Browsing privileges grant requested users the ability to scan or query the information in a website. The browsing privilege has view, link, and view-all three types. A user uses view access right to read the requested information. Link privilege allows a user to understand the particular information on a link. View-all integrates the access right from the view and link [1].

Copying allows a user to store the requested materials from websites. Editing grants a user to modify the material using delete, insert, update, and compose operations. Please see TABLE 8 for detailed information.

6.1.4 Access Authorization

The goal of authorization is to control service access [33]. Authorization process is generally evaluating the access request credential using request identity. Access policy

TABLE 8
Authorization Privilege Model (Revised from Wang and Simmons 2006) [96]

Type	Privilege mode	Description
Browsing	View	Understand requested information
	Link	Observe the existence of contents
	View-all	Get information in the requested links
Copying	Download and Store	Save the information from the requested website
Editing	Insert	Put more information into website
	Delete	Move information out from a website
	Update	Change information objects as desired
	Compose	Add contents to a website

will decide every user to access the web-based applications. Access policy can be defined in subject credentials, object credentials, access privileges, and requested access time [1], [7]. The proposed authorization process will emphasize subject credential evaluations in user profiles, IP addresses, and geographical location information. The authorization process also combines request object and access privilege. The authorization can be formally defined as follows:

Definition 6 [Authorization] Authorization can be expressed in a policy specified by (ce, ao, pm, t), where ce is a credential expression used to describe authorized subjects, ao is an authorized object identifier for requested materials, pm represents a privilege mode, and t is a time variable used to define the valid access time.

An example for authorization can be expressed as follows:

au = (Jeff(x)Λ(geoip(x) equals “128.165.12.10, Forest Lane, Texas Instrument, Dallas, Texas 77081”, type(y)=(self-support materials), {view}, [10/5/2005, 12/2/2007]).

The above example means: au defines that an user Jeff is granted to access to the technical support materials from the IP address in 128.165.12.10 at Texas Instrument, Forest Lane, Dallas, TX 77081 with a view access right starting from October 5, 2005 to December 2, 2007. We can develop the authorization expressions into an authorization knowledge base.

6.1.5 Rules in Authorization Knowledge Base

Rules in authorization base can be directly derived from the specific authorization expressions [7]. In Atluri and Chun’s approach, derived rule is ordered from the privilege mode. In our proposed scheme, the authorization knowledge base is based on rule expressions in subject. For example, au = (David(x) Λ (geoip(x) equals “165.128.10.3, Welcome Blvd, Houston, TX 77805”, type(y) = (self-support materials), {insert}, {10/2/2006, 12/2/2007}). For the ip address in 165.128.10.3 from the above physical location, rule au’ = (David(x) Λ (geoip(x) equals “165.128.10.3, Welcome Blvd, Houston, TX 77805”, type(y) = (self-support materials), {delete}, {10/2/2006, 12/2/2007}) is true also since the request IP address has edit access privilege.

In the knowledge base, we can develop and refine rules from empirical data to do access control. Then, we may conduct user profile analysis, knowledge analysis, and utility evaluation for the role base. Specifically, usability and performance will be evaluated [86].

6.2 Access Control [96]

Anderson classified access controls into four different levels. The access control can apply to applications, middleware, operating systems, and the hardware level [3]. For the web-based applications, the access control is applied to the application level. The access control strategy is to map request identities to their access privileges and the privileges to the request materials [11], [35]. Hollar and Murphy brought up the system performance issues in the web-based applications. The performance is related to the number of roles used in the access decisions. There are several access control schemes existing. These include role-based access control, attribute-based access control, mandatory based access control, etc. The role-based access control is selected in this dissertation. Formally, access control can be defined in access request expression. The proposed access control algorithms are defined as follows [1], [7]:

Definition 7 [Access Request Expression]. Access request expression can be represented as ARE = (s, ro, pm), here s is the requested subject ID, ro is requested object expression, and pm is the privilege mode that the access request has.

For example,

ARE1 = (Jefferson, 18(y), delete)

ARE2 = (Daniel, type(k)=”Nortel technical support materials”, view)

The first statement is integrated as Jefferson plans to delete an object with identifier number 18. If we have access request ARE = (s, ro, m) and an authorization base $\{z_1, z_2, z_3, \ldots z_n\}$ where z_i is defined as (ce, ao, pm, t), the access control in the web server will examine the authorizations for ARE from the authorization base. The authorization is taken in several steps. When a remote user submits a request to the web-

based applications to request some materials, the system will first examine the subject credential expression. If the request user passes the examination, the system will look into the authorized object to see whether the requested object characteristics are satisfied. If the examination meets the credentials, the privilege mode m will be searched. If all the criteria meet the rules, the requested objects will be queried and transported. To outline the discussions above, the authorization process adapts four major steps: Subject credential match-up, object expression check, privilege examination, and finally delivering the requested information. However, all these processes are defined using the access control algorithms.

6.2.1 Access Control Algorithms

In access control algorithms, the most important access control is authorization. The authorization algorithms start from searching the user credentials. If all the user credentials are found, the users will get into the system. User access is evaluated using the access control algorithms in Fig. 7. Please see Fig. 7 for details. In the authorization process, the most important examination is searching the authorized users (Fig. 8). User credential is composed with user credential expressions and queried from the authorization base. In this dissertation, an exact user authorization is represented

```
Access control algorithm 1

Input: Remote user request in RUR = (s, ro, pm), where s is a
       subject credential expression, ro is request objects, pm is a
       privilege mode.
Output: An access request is authorized or declined.

AU = search-authorized-user(s)
Initialize list re={};

Begin:
If AU = {} {
   return (access_declined);
}
else {

    for each aoi = {si, roi, pm} Є AUi  {
        If pm == ('view' or 'link' or 'view-all')
              re = re ∪ view(aoi ) ∪ link(aoi ) ∪ view-all (aoi );
         else if (pm == 'copy')
              re = re ∪ aoi;
        else if (pm == 'insert' or 'delete' or 'update' or 'compose')
             re = re ∪ insert(aoi) ∪ delete(aoi) ∪ update(aoi) ∪
                  compose(aoi);
     }
      return (re);
   }

End
```

Fig.7. Authorization in access control (Revised from Wang and Simmons 2006) [96].

```
Access control algorithm 2: Search Authorization Users

Input: Remote user request: RUR= (s, ro, pm), where s is a request
       subject credential expression, ro is requested objects, and pm
       is a request privilege mode. Access material set: O; privilege
       set: m.
Output: Requested users are granted or declined.


Begin:

Initialize stack AU to the empty;

If  requested_user_login and ip_address are found {
    If zip_code and city_name and country_name match {

        If (ro Є O) and (pm Є m)
            push RUR to Stack AU;
    }
 popup   Stack AU;
 return (AU) ;
}
End
```

Fig. 8. Request access user evaluation (Revised from Wang and Simmons 2006) [96]

Algorithm 3: Subject credential evaluation

Procedure Evaluate-subject-credentials (ce, P).

Input: ce is a subject credential expression, P is the existing user credential profiles.
Output: TRUE or FALSE.
Initilize a Link-list S

```
For each pair (attribute_name, a_value) Є ce_i  {
    If (ce(attribute_name) = P(attribute_name)) and
       ce(a_value) = P (a_value))
         flag = 1;
    else
         flag =-1;
}
While ((flag ==1) and (ce_i != NULL)){
      Add ce_i in string to Link-list S;
      ce_i = Link-list[i];
      (exp1, operator, exp2) = Split (ce_i);
      // operators take one of the operations in {>, <, =, ≠, ≤, ≥, υ, etc}
      er = (exp1 operator exp2);
                        /*evaluate the expression */
      return er;
}
```

Fig 9. Subject credential expression evaluation (Revised from Atluri and Chun 2004 [7])

using remote user information, ip address, and geographical location information. The user credential checks also include the role operations in delete, insert, update, etc.

Subject credential expression evaluation algorithm is presented in Fig. 9. The evaluation also includes subject credential expression operations. These include insert, delete, update, etc. Privilege evaluation algorithm is listed on Fig. 10. In the credential expression evaluation, there are some logical operations that need to translate into mathematical expressions. The authorization base system either grants the request privilege or declines it.

```
Access control algorithm 4: Privilege mode evaluation

Input: An user request privilege mode: pm, a set of privilege modes
in authorization: m.
Output: TRUE or FALSE

Initialize m={a group of privilege modes}

If  ( pm Є m)

      Return (TRUE);

Else
      Return (FALSE);
```

Fig. 10. Privilege mode evaluation (Revised from Wang and Simmons 2006) [96].

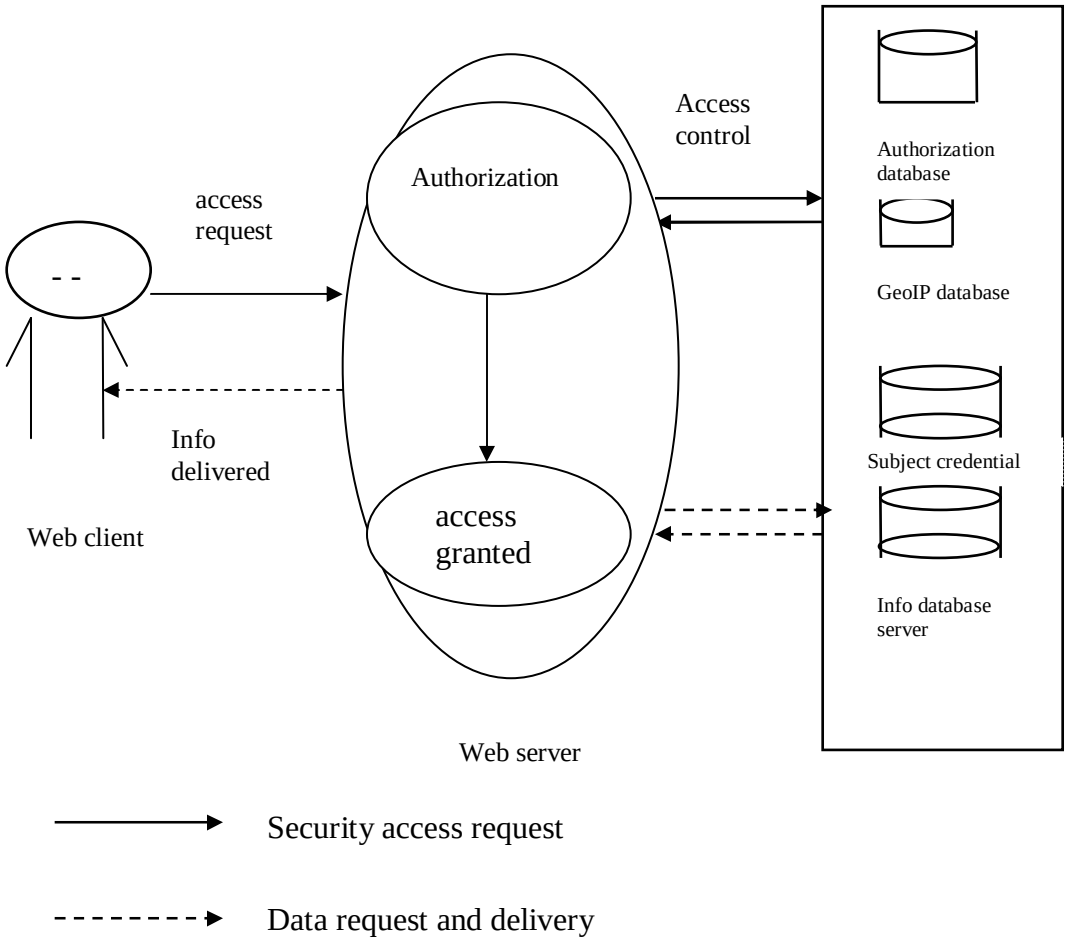

Fig. 11. System architecture (Modified from Netgeo 2002; Wang and Simmons 2006) [64], [96].

6.3 System Architecture for Implementation [96]

The proposed system architecture is described in Fig. 11. Because the proposed authorization model mostly relies on the subject credential for access evaluation, the system architecture will appraise the conditions from subject credential expressions first. Next, the requested object expressions will be examined using the object identifiers. Lastly, the privilege model will be checked. If all the evaluations are satisfied, remote users will obtain permission to access the requested information. Hollar and Murphy (2006) proposed three options for the access control decision point [34]. The three options are integrating the access control: (1). Within the web server as a component; (2). As a external procedure; (3). As part of database management system access control. In the opinion of Hollar and Murphy, taking use of database management system access control as the access control decision point is the best practice. Integrating the access control with the web service is the least attractive approach. In our proposed approach, we intend to integrate the access control in the authorization base with the database management system in Fig. 11.

6.4 Major Contributions

Atluri and Chun [7] developed an authorization model in geospatial data. The requested access is granted according to the image location, resolution, and the downloaded time in the data. The authorization relies on the geospatial data. Netgeo (2002) developed an architecture using ip addresses and physical locations in credit card fraud detection [64]. The approach is a good practice for credit access verifications. The method is not flexible for different organizations though. The method belongs to an exact

user authorization. Adam, Atluri, and Bertino (2002) specified a content-based authorization model used in digital libraries [1]. The model authorizes the requested users only using their requested contents.

In our proposed solution for the web-based applications, we propose to use a role-based access control using the enhanced subject credential evaluations. Our proposed model has a high flexibility in order to specify authorization rules according to the web product requirements. The proposed model integrates flexible role-based access control with an exact subject credential expression in ip address and geographic location information. The model appears to be prominent in all the approaches available. The access control and authorization are both regarding computer systems which allocate the resources based on the identity and their privileges. Both are associated with the authentication process. As the role-based access control has received more attention recently, the role-based access control has being combined with other property in the designed system. For example, Xu et. al. (2004) developed a service–oriented role-based access control [102]. They specified all the services in their roles. Our approach is to adapt the user Internet service property in ip address and their geographic information as subject credentials to authenticate the users. The information can keep the rules updated in the web-based application systems as the remote users use the intended services and have some network activities associated with the provided services. Our approach has appeared to be an attractive approach in comparison with the existing approaches.

6.5 Future Performance Analysis

Future performance analysis will concentrate on scalability analysis and RBAC flow analysis. The scalability analysis will be conducted to simulate the number of access users and system performance. The system performance in response time, throughputs, and availability will be measured for different number of access users. Using these parameters, we can estimate whether the designed system can perform the intended functions. If the performance is demanding, we may consider using multiple web servers in parallel to improve the system performance in throughputs, response time, and availability in the web-based applications. For the detailed measurements, please refer to Menasce and Almeida's book on "Capacity planning for web services: Metrics, models, and methods" [57].

The RBAC flow analysis will use role graphs to simulate role hierarchies and analyze the RBAC. In the analysis, we need to pay attention to the separation of duty and role conflicts. For instance, role-based access controls need continual human operations in being kept updated. The operations include role addition, role deletion, role-privilege update, and etc. Access permissions are assigned to different roles. Users are assigned to a set of roles. Access decisions are based on users and associated roles [11]. The information flow in the RBAC can be expressed as role graphs with directed edges for access privileges. The role graphs are used to track information flows among objects in the RBAC system. The approach assumes that the content in an object can be duplicated to another object. The duplicated operation is realized using read and write privileges. For the detailed analysis, please refer to Benantar's book in "Access control system: Security, identity management and trust model".

In the role-base access control, we have challenges in separating duty. Duty separation has two possible ways: Static separation (authorization time separation) and dynamic separation (runtime separation). The dynamic duty separation has simple dynamic, object-based, operational, and history-based separation of duty. Simple dynamic separation is that users are not allowed to use two limited roles in multiple sessions at a time point. Object-based separation means that no two separated roles can be assigned to an object that has been operated. Object-based separation restricts a user to a single action on only one object.

We also need to handle role conflicts. For example, separation of duty is one type of handling conflicts. The role conflicts include conflicting permissions, users, and tasks. For conflicting permissions, we need to rearrange different permissions as unordered pairs. For conflicting users, any set of users is reduced to a single user and each user is assigned to non-conflicting roles. For conflicting tasks, a set of tasks for a business process require conflicting access permissions to accomplish. Conflicting tasks are assigned to different roles. All these routine operations can become expensive and difficult when there are extremely large access systems involved. The good news is that the research used in role-based trust management language and runtime engine to map entities with roles in their properties is in progress. The new system can retrieve digital credentials remotely.

In the dissertation, I present the improved security framework. The proposed framework improves application security in the web-based applications using enhanced subject credential authorization in computer geographical information. The future work may center on converting the proposed framework to a system. The system performance

parameters will be measured and analyzed. Taking into consideration the system performance parameters, we can make our model become an adaptive system with a satisfied performance. I would like to conclude this chapter with an opinion of Bindiganavale and Ouyang (2006) [13]. The role-based access control has become an excellent approach for access control because of the reduced complexity and cost for system administrators. Using RBAC, security system management in user profiles is easily operated in roles, hierarchies, and access privileges.

CHAPTER VII

RESEARCH SUMMARY

7.1 Research Summary

In this dissertation, I conducted both a security analysis and a security related risk assessment. Preliminary analysis indicates that there are several vulnerabilities existing in the open web proxy honeypot. Because these vulnerabilities exist, computer hackers attack the vulnerabilities and penetrate into the systems, causing security concerns for the web-based applications. The security is measured by immediate system availability and mean time to security failure in Markov Process Model. This is the first attempt to apply Markov Process Model to multiple component software systems to measure security in complex systems. This approach can be applied to other complex systems. Also, because software vulnerabilities are present in the web-based applications, security related system risk is assessed in fuzzy logic. Vulnerability based risk assessment is a new way to measure software system reliability in security. The root cause of security issues and security related risks is software vulnerabilities. As computing applications become pervasive, security issues and their risks have become an important concern for trusted computing. To improve software security, several security enhancement approaches have been proposed to address security related problems. Specifically, I presented an enhanced security improvement model in access control algorithms using GeoIP service as a new approach.

The first part of the research explored the security analysis for the web-based applications. In the beginning chapters, I presented some security analysis of the studied

systems as a background, and the following chapters discussed security measurement in access availability and mean time to security using Markov Process model. Each subsystem in web client, web server, and database server was modeled as good, vulnerable, attacked, and security-failure states. The whole system was modeled by the Markov Process Model. The subsystem interactions were also modeled by the Markov Model in the state transition diagram. The security failure in the system was analyzed by an example using the real data. The method for system risk analysis was developed using the reported vulnerabilities and fuzzy logic methods. To overcome the subjective measurements in fuzzy logic method, information entropy theory was applied as weight coefficient to adjust the measurement. The outlined methods have potential to become a standardized method for the security analysis in the web-based application systems.

The second part of the dissertation presents the specification and implementation architecture for the web-based application security systems. The proposed methods are easy to adapt to different web-based applications according to the organization requirements (technical support materials in commercial companies, digital storage deposit, patient information systems, etc). The existing authorization models do not emphasize subject credential evaluation enough, which is the most important in access control. In this dissertation, the subject credential in remote user logins, user IP addresses, zip codes, cities, states, and countries are integrated with the access privilege and requested objects to authorize access requests. The proposed approach incorporates the merits of existing methods in this research area. Simply put, the first part of the dissertation analyzes security issues and its related risk, and the second part proposes a solution to improve security in the web-based applications.

7.2 Future Research

Future work for the security analysis in the web-based applications will focus on the model performance evaluation. More case studies, however, are needed to validate the proposed approach; future work will likely center on sensitivity analysis for the outlined methods. As any risk study in other systems, these studies in the web-based applications have dependent and independent variables that are connected by a system equation that combines technical, financial, and other factors. Each factor is evaluated by the risk model with the expected output that indicates whether the security risk factor is suitable or not. The key risk input variable will be decided by the evaluation.

Many trials have been conducted to develop algorithms to indicate which variables have more contribution to risk index. The risk index for all the variables will be different with each round of inputs. To overcome this, hold-all-but-one-constant (HABOC) method has been developed. For the detailed information and case study procedures, please refer to Koller's book "Risk assessment and decision making in business and industry: A practical guide" (1999) [48]. The security analysis data used in this dissertation was collected by an open web proxy honeypot. In an operating web-based application system, the data may have some variance from the data collected by the open proxy honeypot. Nevertheless, open web proxy honeypots are still the most popular method for data collection in security analysis.

For the enhanced security model in the web-based applications, I plan to extend the framework here to implement a secure web-based application. As the web-based applications become a major platform of computing, security requirements for web application are also higher. The negative aspect of the proposed approach may result in

some privacy concerns, because request user locations are revealed in the subject credential expressions for authorization purposes. Performance evaluation will be conducted as the scalability analysis needs to be measured. Although the security algorithms developed in this dissertation are good, the system in the proposed approach may have some performance limitations in throughput and processing time when having an extremely large number of access requests. By incorporating the performance parameter considerations, the proposed system will have good performance as expected.

REFERENCES

[1] N. R. Adam, V. Atluri, E. Bertino, and E. Ferrari, "A Content-based Authorization Model for Digital Libraries," *IEEE Transactions on Knowledge and Data Engineer,* vol. 14, no.2, pp. 296-315, 2002.

[2] N. Aghdaie, and Y. Tamir, "Implementation and Evaluation of Transparent Fault-tolerant Web Service with Kernel-level Support," *Proceedings of the IEEE International Conference on Computer Communications and Networks,* Miami, FL, October, 2002.

[3] R. Anderson. *Security Engineering: A Guide to Building Dependable Distributed Systems*, Wiley & Sons, Inc., New York, 2003.

[4] M. Andrews, "The State of Web Security," *IEEE Security & Privacy,* vol. 4, no. 4, pp.14-15, 2006.

[5] S. Apel, *Software Reliability Growth Prediction – State of the Art*. Fraunhofer Institute Experimentelles Software Engineering, Kaiserslautern, Germany, 2005.

[6] A. Arora, and R. Telang, "Economics of Software Vulnerability Disclosure," *IEEE Security & Privacy,* vol. 3, no.1, pp. 20-25, January/February, 2005.

[7] V. Atluri, and S. A. Chun, "An Authorization Model for Geospatial Data," *IEEE Transactions on Dependable and Security Computing,* vol. 1, no.4, pp. 238-254, 2004.

[8] R. Barnett, "Open Proxy Honeypots," http://honeypots.sourceforge.net. 2004

[9] R. Barnett, *Preventing Web Attacks with Apache*, Pearson Education Inc., Upper Saddle River, NJ, 2006.

[10] P. Barford, A. Bestavros, A. Bradley, and M. Crovella, "Changes in Web Client Access Patterns: Characteristics and caching implications," *World Wide Web*, Special Issue on Characterization and Performance Evaluation, pp. 15-28, 1999.

[11] M. Benantar, *Access Control Systems: Security, Identity Management and Trust Models*. Springer Science + Business Media, Inc. New York, 2006.

[12] E. Bertino, P. A. Bonatti, and E. Ferrari, "TRBAC: A Temporal Role–based Access Control Model," *ACM Transactions on Information and System Security*, vol. 4, no. 3, pp. 191-223, 2001.

[13] V. Bindiganavale, and J. S. Ouyang, "Role-based Access Control in Enterprise Application – Security Administration and User Management," *IEEE International Conference on Information Reuse and Integration*, Hawaii, September 2006.

[14] K. P. Birman, *Reliable Distributed Systems: Technologies, Web Services, and Applications*, Springer Verlag, New York, 2005.

[15] S. Biyani and P. Santhnam, "Exploring Defect Data from Development and Customer Usage on Software Modules over Multiple Releases," *Proceedings of the Ninth International Conference on Software Reliability Engineering*, Paderborn, Germany, pp. 316-320, 1998.

[16] G. Bolch, S. Greiner, H. de Meer, and K. S. Trivedi, *Queueing Networks and Markov Chains*, John Wiley & Sons, New York, 1998.

[17] P. A. Bonatti, & P. Samarati, "A Uniform Framework for Regulating Service Access and Information Release on the Web," *Journal of Computer Security*, vol. 10, no.3, pp. 241-271, 2002.

[18] L. Briand, K. El Eman, and B. Freimut, "A Comparison and Integration of Capture-Recapture Models and Detection Profile Method," *Proceedings of the Ninth International Conference on Software Reliability Engineering*, Novemver 1998.

[19] P. T. Chen, C. S. Laih, F. Pouget, and M. Dacier, "Comparative Survey of Local Honeypot Sensor to Assist Network Forensics," *Proceedings of the First International Workshop on Systematic Approaches to Digital Forensic Engineering*, November 2005.

[20] A. Chuvakin, "Honeynet Project Scan of the Month Challenges #31 and #34" http://www.honeynet.org/scan, 2005.

[21] M. Coetzee, and J. H. P. Eloff, "An Access Control Framework for Web Services," *Information Management & Computer Security*, vol. 13, no. 1, pp.29-38, 2005.

[22] E. Cohen, R. K. Thomas, W. Winsborough, and D. Shands, "Models for Coalition-based Access Control (CBAC)," *SACMAT'02*, pp.97-106, Monterey, CA, 2002.

[23] M. Curphey, and R. Araujo, "Web Application Security Assessment Tools," *IEEE Security & Privacy*, vol. 4, no. 4, pp. 32-41, July/August 2006.

[24] M. Dacier, F. Pouget, and H. Debar, "Honeypots: Practical Means to Validate Malicious Fault Assumption," *Proceedings of the 10th Pacific Rim International Symposium on Dependable Computing*, March, 2004.

[25] Y. Deswarte, and D. Powell, "Internet Security: An Intrusion-tolerance Approach," *Proceedings of the IEEE*, vol. 94, no. 2, pp. 432-441, February 2006.

[26] R. M. Feldman, and C. Valdez-Flores, *Applied Probability and Stochastic Processes*, 2nd Edition, PWS Publishing Company, St. Paul, MN, 2006.

[27] M. Dowd, J. Mcdonald, and J. Schuh, *The Art of Software Security Assessment: Identifying and Preventing Software Vulnerabilities*, Pearson Education, Inc. Boston, MA, 2007.

[28] S. Friedl, "SQL injection attacks by example," http://www.unixwiz.net/techtips/sql-injection.html, 2005.

[29] K. Goseva-Popstojjanova, F. Y. Wang, R. Wang, F. M. Gong, K. Vaidyanathan, K. Trivedi, and B. Muthusamy, "Characterizing Intrusion Tolerant Systems Using a State Transition Model," *DARPA Information Survivability Conference & Exposition II*, IEEE, June 2001.

[30] J. Grossman, "Analyze Web Application Attack Data," http://www.webappsec.org/lists/websecurity/archive/2006-07/msg00062.html, 2006.

[31] C. Griffin, B. Madan, and K. Trivedi, "State Space Approach to Security Quantification," *Proceedings of the 29th Annual International Computer Software and Applications Conference*, IEEE, July, 2005.

[32] B. Hickman, "Application Security and Threat Modeling," http://cpd.ogi.edu/seminar04/hickmanthreatmodeling.pdf, 2004.

[33] R. Hollar, and R. Murphy, *Enterprise Web Services Security*. Charles River Media, Hingham, MA, 2006.

[34] Y. S. Hong, J. H. No, and I. Han, "Evaluation of Fault-tolerant Distributed Web Systems," *Proceedings of the 10th IEEE International Workshop on Objected-Oriented Real-Time Dependable Systems*, January 2005.

[35] M. Howard, D. Leblanc, and J. Viega, *19 Deadly Sins of Software Security: Programming Flaws and How to Fix Them*, McGraw-Hill/Osdborne, Emeryville, CA, USA, 2005.

[36]. M. A. Howard, "A Process for Performing Security Code Reviews," *IEEE Security & Privacy*, vol. 4, no.4, pp. 74-79, July/August, 2006.

[37] S. Indrakanti, V. Varadharajan, and M. Hitchens, "Authorization Services for Web Services and its Application in a Health Care Domain," *International Journal of Web Services Research*, vol. 2, no. 4, pp. 94-119, 2005.

[38] R. Isermann, *Fault-Diagnosis Systems: An Introduction from Fault Detection to Fault Tolerance*, Springer-Verlag, Berlin, Heideberg, Germany, 2006.

[39] G. Janakiraman, J. R. Santos, D. Subhraveti, and Y. S. Turner, "Cruz: Application-Transparent Distributed Checkpoint-restart on Standard Operating Systems," *Proceedings of the 2005 International Conference on Dependable Systems and Networks*, June 2005.

[40] S. Jha, O. Sheyner, and J. Wing, "Minimization and Reliability Analysis of Attacked Graphs," Technical Report CMU-CS-2-109, Carnegie Mellon University, May 2002.

[41] D. A. Jones, M. A. Turnquist, C. E. Davis, and L. K. Nozick, "Physical Security and Vulnerability Modeling for Infrastructure Facilities," *Procceedings of the 39th Hawaii International Conference on System Sciences*, IEEE, January, 2006

[42] J. B. D. Joshi, W. G. Aref, A Ghafoor, and E. H. Spafford, "Security Models for Web-based Applications," *Communication of ACM*, vol. 44, no. 2, pp. 38-43, 2001.

[43] J. Joshi, E. Bertino, U. Latif, and A. Ghafoor, "Generalized Temporal Role Based Access Control Model," *IEEE Transactions on Knowledge and Data Engineer*, vol. 17, no.1, pp. 4-23, 2005.

[44] J. B. D. Joshi, E. Bertino, and A. Ghafoor, "An Analysis of Expressiveness and Design Issues for the Generalized Temporal Role-based Access Control Model," *IEEE Transactions on Dependable and Secure Computing*, vol. 2, no. 2, pp. 157-175, 2005.

[45] A. H. Karp, G. J. Rozas, A. Banerji, and R. Gupta, "Using Split Capabilities for Access Control," *IEEE Software*, vol. 20, no. 1, pp. 42-29, 2003.

[46] G. R. Koller, *Risk Assessment and Decision Making in Business and Industry: A Practical Guide*, CRC Press, Boca Raton, FL, 1999.

[47] D. Lake, "Asleep at the Wheel," *The Industry Standard*, December 4, 2000.

[48] C. Landwehr, "Formal Methods for Computer Security," *Computer Surveys*, vol. 13, no. 3, pp. 247-278, 1981.

[49] V. C. S. Lee, and L. Shao, "Estimating Potential IT Security Losses: An Alternative Approach," *IEEE Security & Privacy Magazine*, vol. 4, no. 6, pp. 44-52, 2006.

[50] N. G. Leveson, "Completeness in Formal Specification Language Design for Process Control System," *Proceedings of Formal Methods in Software Practice Conference*, August, 2000.

[51] Z. Li, and J. Tian, "Testing the Suitability of Markov Chains as Web Usage Models," *Proceedings of the 27th Annual International Computer Software and Applications Conference*, November 2003.

[52] P. Lindstrom, "Security: Measuring up," http://www.searchsecurity.techtarget.com/tip/1,289483,sid14_gci1060349,00.html, 2005.

[53] J. Lowry, "An Initial Foray into Understanding Adversary Planning and Courses of Action," *Proceedings of DARPA Information Survivability Conference and Exposition II*, pp. 123-133, June 2001.

[54] H. P. Lu, C. L. Hsu, and H. Y. Hsu, "An Empirical Study of the Effect of Perceived Risk upon Intention to Use Online Applications," *Information Management and Computer Security*, vol. 13, no. 2, pp. 106-120, 2005.

[55] B. B. Maden, K. Goseva-Poptojanova, K. Vaidyanathan, and K. S. Trivedi, "Modeling and Quantification of Security Attributes of Software Systems," *Proceedings of the International Conference on Dependable Systems and Networks*, IEEE, June 2002.

[56] J. Mclean, "Twenty Years of Formal Methods," *Proceedings of the 1999 IEEE Symposium on Security and Privacy*, May 1999.

[57] D. A. Menasce, and V. A. F. Almeida, *Capacity Planning for Web Service: Metrics, Models, and Methods*. Prentice-Hall, Inc. Upper Saddle River, NJ, 2005.

[58] D. A. Menasce, "Performance and Availability of Internet Data Centers," *IEEE Internet Computing*, vol. 10, no.3, pp. 94-96, May/June 2004.

[59] J. F. Meyer, "On Evaluating the Performability of Degradable Computing Systems," *IEEE Transactions on Computers*, vol. 29, no. 8, pp.720-773, August 1980.

[60] J. Mitchell, 2002, "Formal Methods and Computer Security," http://theory .stanford.edu/~jem/slides/jcm-usenix-02.ppt.

[61] L. E. Moser, P. M. Melliar-Smith, W. B. Zhao, "Making Web Services Dependable," *Proceedings of the First International Conference on Availability, Reliability, and Security*, April 2006.

[62] M. A. Mustafa, and J. Fai-Bahar, "Project Risk Assessment Using the Analytic Hierarchy Process," *IEEE Transactions on Engineering Management*, vol. 38, no. 1, pp. 46-52, 1991.

[63] National Vulnerability Database, 2006, http://nvd.nist.gov/, April 2006.

[64] Netgeo, "Internet Geography Solution," http://www.netgeo.com, 2002.

[65] Netgeo, "DaVita Chooses Netgeo for Territory-based Online Content Delivery", http://www.netgeo.com/PR_09102004.html.

[66] P. Neumann, "Principled Assuredly Trustworthy Composable Architectures," Final Report for DARPA's Composable High-assurance Trustworthy System Program, http://www.csl.sri.com/users/neumann/chats4.pdf, 2004.

[67] Newbug@chroot.org, "Newbug Report," http://packetstormsecurity.nl/0501-exploit/AWstateVulnAnalysis.pdf, 2007.

[68] D. M. Nicol, W. H. Sanders, and K. S Trivedi, "Model-based Evaluation: From Dependability to Security," *IEEE Transactions on Dependable and Secure Computing*, vol. 1, no. 1, pp. 48-65, 2004.

[69] S. Northcutt, M. Cooper, M. Fearnow, and K. Frederick, *Intrusion Signatures and Analysis*, Sams, Indianapolis, 2002.

[70] Y. Peng, and Q. Y. Wu, "Secure Communication and Access Control for Web Services Container," *Proceedings of the Fifth International Conference on Grid and Cooperative Computing*, November 2006.

[71] G. Peterson, "Introduction to Identity Management Risk Metrics," *IEEE Security & Privacy*, vol.4, no.4, pp.88-91, July/August, 2006.

[72] K. Pousen, "Nimda Worm Hits Net," http://www.securityfocus.com, 2001.

[73] R. S. Pressman, *Software Engineering: A Practitioner's Approach*, 6th edition, The McGraw-Hill Companies, Inc, New York, 2005.

[74] Purdue University, "Vulnerability Report," https://cirdb.cerias.purdue.edu/ccopvdb/public.

[75] F. Raynal, Y. Berthier, P. Biondi, and D. Kaminsky, "Honeypot Forensics Part 1: Analyzing the Network," *IEEE Security & Privacy Magazine*, vol. 2, no. 4, pp. 72-78, 2005.

[76] J. P. Ravenel, "Effective Operational Security Metrics," *Information Systems Security*, vol. 15, no. 3, pp. 10-17, Jul/Aug, 2006.

[77] W. Ren, and H, Jin, "A Recursion Nearness Based Method for Characterizing IP Address," *Proceedings of the Six International Conference on Parallel and Distributed Computing, Applications and Technologies*, December 2005.

[78] N. A. Renfroe, and J. L. Smith, "Threat/Vulnerability Assessments and Risk Analysis," Applied Research Associates, Inc, http://www.wbdg.org/desihn/riskanalysis.php, 2006.

[79] RFC, "RFC-2616-HTTP/1.1," http://www.ietf.org/rfc2616.txt, 1999.

[80] S. M. Ross, *Stochastic Process*, 2nd Edition, John Wiley & Sons, New York, 1996.

[81] S. E. Schechter, "Toward Econometric Models of the Security Risk from Remote Attacks," *IEEE Security & Privacy*, vol. 3, no. 1, pp. 40-44, January/February, 2005.

[82] T. Schmidt, G. Wippel, K. Glanzer, and K.Furst, "Security System for Distributed Business Applications," *International Journal of Web Service Research*, vol. 2, no.1, pp. 77-88, 2005.

[83] B. Schneier, *Secrets and Lies*, John Wiley & Sons, New York, 2000.

[84] S. Shah, “Detecting web application security vulnerabilities,” http://www.oreilly.com/pub/a/sysadmin/2006/11/02/webapp_security_scans.html, 2006

[85] H. B. Shen, and F. Hong, “An Attribute-based Access Control Model for Web Services,” *Proceedings of the Seven International Conferences on Parallel and Distributed Computing, Applications, and Technologies*, December 2006.

[86] P. Shyr, G. Tecuci, and M. Boicu, “Evaluation of Mixed-Initiative Knowledge Base Development Methods and Tools,” *Proceedings of IJCAI-2001 Workshop on Empirical Methods in AI*, May 2001.

[87] I. Sommerville, *Software Engineering*, 7th Edition, Pearson Education Limited, Harlow, United Kingdom, 2004.

[88] P. Sousa, N. F. Neves, and P. Verissimo, “How Resilient are Distributed Fault/Intrusion-tolerant Systems?” *Proceedings of the 2005 International Conference on Dependable Systems and Networks*, IEEE, June 2005.

[89] C. Stringfellow, and A. A. Andrews, “An Empirical Method for Selecting Software Reliability Growth Models,” *Empirical Software Engineering*, vol.7, pp. 319-343, 2002.

[90] O. Sheyner, J. Haines, S. Jha, R. Lippmann, and J. Wing, “Automated Generation and Analysis of Attack Graphs,” *Proceedings of IEEE Symposium on Security and Privacy*, pp. 273-284, 2002.

[91] S. Splaine, *Testing Web Security*, Wiley Publishing, Indianapolis, 2002.

[92] Terms definitions, http://www.whatis.com, 2006

[93] J. Tian, S. Rudraraju, and Z. Li, “Evaluating Web Software Reliability Based on Workload and Failure Data Extracted from Server Logs,” *IEEE Transactions on Software Engineering*, vol. 30, no. 11, pp.754-769, November 2004.

[94] K. S. Trivedi, *Probability and Statistics with Reliability, Queuing, and Computer Science Applications*, 2nd edition, John Wiley & Sons, New York, 2001.

[95] J. Viega, and G. McGraw, *Building Secure Software: How to Avoid Security Problems the Right Way*, Addison-Wesley, MA, 2002.

[96] Y. Wang, and D. Simmons, "Enhanced Enterprise Web Based Application Security Using GeoIP Services", *Proceedings of the 10th IASTED Conference on Software Engineering and Applications*, November 2006.

[97] Y. Wang, W. M. Lively, and D. B. Simmons, "Software Security Analysis and Assessment for Web-based Applications", *Proceedings of the 17th ISCA International Conference on Software Engineering and Data Engineering*, July 2008.

[98] Wikipedia, "Formal Methods," http://en.wikipedia.org/wiki/ Category:Formal_methods, 2006.

[99] Web Application Security Consortium, "Threat Classification," http://www .webappsec.org/projects/threat/v1/WASC-TC-v1_0.txt, 2004.

[100] C. Wohlin and P. Runeson, "Defect Content Estimation from Review Data", *Proceedings of the International Conference on Software Engineering*, April 1998.

[101] D. S. Xing and J. Y. Shen, "A New Markov Model for Web Access Prediction", *Computing in Science & Engineering*, pp.34-39, November/December, 2002.

[102] F. Xu, J. Xie, H. Huang, and L. Xie, "Context-Aware Role-based Access Control Model for Web Services," *Proceedings of International Workshop on Information Security and Survivability for Grid*, October 2004.

[103] N. Ye, X. Y. Li, Q. Chen, S. M. Emran, and M. M. Xu. “Probabilistic Techniques for Intrusion Detection Based on Computer Audit Data,” *IEEE Transactions on Systems, Man, Cybernetics-Part A: Systems and Humans*, vol. 31, no. 4, pp. 266-274, July 2001.

[104] N. Ye, Y. B. Zhang, and C. M. Borror,”Robustness of the Markov-chain Model for Cyber-attack Detection,” *IEEE Transactions on Reliability*, vol. 53, no. 1, pp.116-123, March 2004.

[105] N. Ye, C. Newman, and T. Farley, “A System-fault-risk Framework for Cyber Attack Classification,” *Information Knowledge Systems Management*, vol. 5, no.2, pp. 135-151, 2006.

[106] J. Yen, R. Langari, and L. A. Zadeh, *Industrial Applications of Fuzzy Logic and Intelligent Systems*, IEEE Press, New York, 1995.

[107] J. Zhang, & L. J. Zhang, A Framework to Ensure Trustworthy Web Services, *International Journal of Web Services Research*, vol. 2, no.3, pp. i-xi, 2005.

[108] W. P. Zhao, L. E. Moser, and P. M. Melliar-Smith, “Unification of Transactions and Replication in Three-tier Architectures Based on Corba,” *IEEE Transactions on Dependable and Secure Computing*, vol. 2, no. 1, pp.20-33, 2005.

[109] D. M. Zhao, J. H. Wang, J. Wu, J. F. Ma, “Using Fuzzy Logic and Entropy Theory to Risk Assessment of the Information Security,” *Proceedings of the Fourth International Conference on Machine Learning and Cybernetics*, August 2005.

APPENDIX A

GLOSSARY[†]

ActiveX is a set of programming technologies and tools. ActiveX is a dynamic link of library modular. When you enable a program to run in active X environment, you create a component. The component is called Active X control. ActiveX is Microsoft's response to emerging java technology, and an Active X control is similar to a java applet [92].

AJAX: Asynchronous JavaScript and XML is a tool to develop interactive applications for the web and execute user requests right away. Ajax integrates several programming tools in JavaScript, dynamic HTML, Extensible Markup Language (XML), cascading style sheets (CSS), and the Document Object Model (DOM), etc. Ajax displays web page content change immediately when a user makes an action. This is very different from the http requests that are uploading the whole page [92].

ASP stands for active server page. ASP is an HTML page that runs one or more scripts. The scripts are executed on web server before the html page is delivered to the user. An ASP is similar to a server side include or a common gateway interface. The scripts are running on the server and making a page ready for user. Usually, the script on the server takes input from user request to access data sources, then composes the page before sending it to the requester [92].

[†] The references cited in this appendix are listed in the reference section.

ATN (Automated Trust Negotiation) is an approach that manages the sensitive information communication using access control policies. ATN provides a practical credential language, acknowledge policies, and distributed credential repository [85].

Awstats is a popular log graph software from open source development. It can make statistical graphs from log files. The log files may be from web, ftp or mail server.

CERT: is an abbreviation for the Computer Emergency Readiness Team. It was founded by the Defense Advanced Research Project Agency (DARPA) in November 1988 after the Internet was attacked by an Internet worm. Today, CERT handles major Internet incidents and provides avoidance advice for security breaches. CERT is located at Carnegie-Mellon University, funded by the US federal government [92].

CGI: is an abbreviation for Common Gateway Interface. CGI is a program running on web server that executes a web user request and sends data to the user. When a user fills out a form on a web page, it needs to be processed by an application script. The method for passing data between web servers and applications is called common gateway interface [92].

Code Red: was a computer worm that attacked the Internet on July 13, 2001. The Code Red specifically attacked computers running Microsoft IIS web server. The worm can spread and cause victim machine buffer overflow. In the buffer overflow, the worm uses a long string to infect the computer. Code Red system footprint is default.ida; Code Red network footprint uses port 80 to infect the machine [92].

Denial of Service Attack: is an attack that makes computer resources unavailable to their intended users. Typically, attackers make many requests to flood web servers or network

servers so that the servers cannot provide normal services to their users. Denial of service is one of the popular attacks on the Internet [92].

DCOM: is an abbreviation for Distributed Component Object Model. DCOM is a set of program interfaces that client program objects can request services from server program objects remotely in a network. DCOM relies on the component object model that supports some interfaces to allow clients and servers communicating within the same computer. DCOM also can adapt TCP/IP and http to provide distributed services [92].

FTP: is an abbreviation for File Transfer Protocol. FTP is an easy method for exchanging files between different computers on the network. FTP is an application protocol that uses TCP/IP protocols. FTP is very often used to download files from remote sites to a local computer [92].

Fuzzy logic: is derived from fuzzy set theory for reasoning in applications. Fuzzy set theory was invented by Lotfi Zadeh at University of California in 1965. Fuzzy set is a translation function. A fuzzy set is a mapping from the domain into the interval [0, 1]. This mapping is called the membership or characteristic function of a given fuzzy set. Fuzzy sets can be used to construct linguistic term sets. Term sets represent meaningful abstractions of a variable's value. Fuzzy logic is the mechanism for reasoning with fuzzy rules, which is a factual statement about the application. Rules are expressed in terms of fuzzy relations in Cartesian product of the domain of antecedent and consequent variables. The process of inference is produced in composition of given fact with a given rule. The net effect is a possibility distribution about the domain of definition of the consequent variable [106].

GeoIP: is a database which can map a computer Internet IP address by its physical location information.

HTTP: Hypertext transfer protocol is a group of rules for transferring files on the web. The files may be in text, graphic images, sound, video, and other multimedia formats. HTTP is an application protocol using TCP/IP protocols. HTTP can be used to request and deliver information for end application users. A web server with an HTTP daemon is to wait for HTTP requests and deliver the contents when the requests arrive. Your web browser can act as an HTTP client, sending requests to web servers. When a browser user submits a uniform resource locator (URL) or opens a hypertext link, the web browser builds up a HTTP request and delivers the contents to the Internet Protocol address specified by the URL [92].

Information Entropy: is a measure of the uncertainty associated with a random variable. In the communication area, the higher the entropy is, the more errors [94].

Markov Chain: A stochastic process is a Markov chain if 1) time is discrete, 2). The set of possible values of the process at each time is finite or countably infinite, and 3). It has memoryless property. That is, future state depends on the present state, independent of past states [92].

Markov process: A Markov process is a stochastic process where all the values are calculated from a discrete set. In a first order Markov process, the most recent state determines the result of next one. All the processes can be represented by a Markov transition density matrix [92].

MAC: is an abbreviation for Media Access Control. MAC is a unique hardware address that identifies each node in a computer network. In the open systems interconnection

model (OSI), the media access control is one of two sub-layers of data link control. The other sub-layer is logic link control layer [92].

MS SQL Server: MS SQL server is a relational database management system developed by Microsoft. MS SQL server uses Transact SQL (T-SQL), which is programming extensions to Sybase. Microsoft has added some new features to standard SQL, including transaction control, exception and error handling, row processing and variable declarations, etc [92].

MySQL: is a relational database management systems using Structure Query language (SQL). MySQL is open source software systems. MySQL can support Linux, Unix, and windows platform. MySQL supports application program interfaces (APIs) for several programming languages. Some language examples include C, C++, python, Perl, PHP, TCL, etc [92].

Nimda: is a computer virus that can cause traffic slowdown across the Internet. The Nimda worm spreads in four different ways and specifically infects computers with Microsoft Internet Information server. First, Nimda probes each IP address within a random selected range in computers running Microsoft II server. Second, when visitors access the computer infected with Nimda, the Nimda can be sent to other computers in the Internet in random way. Third, Nimda also infects users within the web server's own local network. Finally, an infected system with Nimda can send an email with the attachment "readme.exe" to the computers in local window address book. To fix the Nimda problem, a patch should be applied to infected machine. Also, users should never open an attached "redame.exe" email [92].

QoS: is an abbreviation for Quality of service. QoS is the concept about transmission rates, error rates, and guaranteed service quality in advance. For example, QoS is a major concern for continuous transmission of video and multimedia information in computer networks [92].

RSS: is a XML-based approach for delivering web content in feeds. Feeds let the user have new content delivered to a computer once when it is published. RSS readers provide the user with summaries of all the feeds in one place. RSS is an abbreviation that refers to one of three different formats, which include RDF Site Summary, Rich Site Summary, and Really Simple Syndication. RSS formats are defined in XML [92].

SOAP: Simple Object Access Protocol is a method for a program running in one operating systems (i.e. Windows) to communicate with a program in the same or different operating systems (i.e. Unix) using HTTP and XML for information exchange. SOAP defines how to code the http header and XML file so that the programs in different operating systems can communicate with each other [92].

SQL Injection: is a kind of software vulnerability in which attackers can use Structured Query Language string operations to gain access to computer and data resources or operate on the data. SQL injection happens when SQL server accepts user input in the SQL statement and the SQL server does not remove dangerous characters from the input [92].

SSI: is an abbreviation for Server Side Include. SSI is a variable value. A server can include SSI in the HTML files before it send to the requestor. For example, last modified date can be inserted in the HTML file as an embedded variable in html. The server can

obtain the last modified date for the file and insert into HTML file before HTML is sent to web requestor [92].

SSL: is an abbreviation for Secure Socket Layer. SSL is a secure protocol for delivering information over the Internet. SSL has been replaced by Transport Layer Security (TSL) which relies on SSL. SSL adapts private and public key encryption scheme from RSA for data communication. SSL is implemented by a program located between HTTP and TCP layers [92].

TCP/IP: Transmission Control Protocol/Internet Protocol is the basic communication protocol for the computer network. When your computer accesses the Internet, your computer is running with a copy of TCP/IP program as is every other computer that you are communicating with (send message or get information). TCP/IP is a two layer program. Transmission Control Protocol on the top layer breaks a message into smaller packets that are transmitted over the Internet. TCP is also responsible for receiving packets and assembling the packets into the original message. The lower layer of Internet Protocol is responsible for delivering each packet to the right destination. TCP/IP uses the client/server model. TCP/IP communication is primarily used for a connection restored to two endpoints. TCP/IP and its applications are “stateless”. Each client request is treated as a new request, independent of any previous one [92].

TTL: Time-to-live is a value in an Internet Protocol (IP) packet that tells a network router whether the packet in the network is too long and should be discarded. For various reasons, packets may not get delivered to their destination in a reasonable length of time [92].

XML: is designed to transport and store data. Extensible Markup Language (XML) is intended to provide a common information format and share both format and the data on the World Wide Web and elsewhere. XML allows users to specify their own elements. XML can be used by any individual or group to share information in a consistent way [92].

XSS: cross site scripting. This is most severe type of web software vulnerability. There are two types of cross site scripting vulnerability: reflected XSS and stored XSS. The stored XSS is about one user input information that is viewed by another user who later visits the same sites. Typically, there are some web forms. One user enters information there, and information is viewed later by other users in input forms. The reflected XSS is about embedding the script into URL. The attacker can email a link to a user. When the user opens the link, the web content is changed by URL [92].

World Wide Web: is a computer system with interlinked hypertext documents. World Wide Web may contain text, video, image materials. It is accessed by computer network [92]

APPENDIX B

LETTERS OF PERMISSION FROM THE COPYRIGHT HOLDERS

1. Copyright Permission Letter for Part of Contents in Chapter V.

Date: Tue, 27 May 2008 10:45:40 -0400

From: Mary Ann Sullivan <isca@ipass.net>

To: Yong Wang <y0w4000@cs.tamu.edu>

Subject: Re: Copyright Permission for AP_2605

Parts/Attachments:

1 OK ~52 lines Text (charset: ISO-8859-1)

2 Shown ~47 lines Text (charset: ISO-8859-1)

Dear Yong Wang:

I hereby grant you permission to use the work as stated below. You should include the copyright from ISCA such as

...

Printed with permission ©ISCA 2008

Regards,

Mary Ann Sullivan

Executive Director, ISCA

At 02:27 PM 5/22/2008, you wrote:

Mary Ann Sullivan

Executive Director, ISCA

International Society for Computers

and Their Applications

975 Walnut Street, Suite 132

Cary, NC 27511 USA

Phone: 919-467-5559

FAX: 919-467-3430

Dear Ms. Mary Ann Sullivan:

I am a doctoral student at Texas A&M University and am writing for permission to include Chapter V from "Software security analysis and assessment for web-based applications" in the proceeding of the 17th ISCA Software Engineering and Data Engineering, Los Angeles, June 30-July 2, 2008 in part of my dissertation. The dissertation will be made available to the public on the web through Texas A&M University Libraries. In addition, the dissertation will be microfilmed by ProQuest Information and Learning Company, and copies of the dissertation will be sold on demand. Please supply a statement granting me permission to use the work. You can email the permission to y0w4000@cs.tamu.edu.

Please advise me with related information if I need to contact the publisher directly.

Thank you for your help.

Sincerely,

Yong Wang

Mary Ann Sullivan

Executive Director, ISCA

International Society for Computers

and Their Applications

975 Walnut Street, Suite 132

Cary, NC 27511 USA

Phone: 919-467-5559

FAX: 919-467-3430

2. Letter from Copyright Permission Holder for Chapter VI

Date: Thu, 29 May 2008 09:41:10 -0600

From: Calgary <Calgary@iasted.org>

To: 'Yong Wang' <y0w4000@cs.tamu.edu>

Subject: RE: Copyright request

Dear Contributor,

You have our permission to reuse your paper as a part of your dissertation.

Please remember that FULL credit must be provided to IASTED for permission to reprint.

Nicholas Woodard

Publications Coordinator

IASTED and ACTA Press

B6, 101 - 2509 Dieppe Avenue SW

Calgary AB T3E 7J9

CANADA

-----Original Message-----

From: Yong Wang [mailto:y0w4000@cs.tamu.edu]

Sent: Thursday, May 22, 2008 10:54 AM

To: Calgary

Cc: y0w4000@cs.tamu.edu

Subject: Copyright request

IASTED Secretariat

Building B6, Suite #101

2509 Dieppe Avenue SW

Calgary, Alberta

Canada T3E 7J9

May 22, 2008

To Whom It May Concern:

I am a doctoral student at Texas A&M University and am writing for permission to include Chapter VI from "Enhanced enterprise web-based application security using GEOIP service" in the Proceeding of the 10th IASTED Conference on Software Engineering and Application, Dallas, 2006 in my dissertation. The dissertation will be made available to the public on the web through Texas A&M University Libraries. In addition, the dissertation will be microfilmed by ProQuest Information and Learning Company, and copies of the dissertation will be sold on demand. Please supply a statement granting me permission to use the work. You can email the permission to y0w4000@cs.tamu.edu. If I need to contact Acta Press publishing directly, please provide their contact information.

Thank for your help.

Sincerely,

Yong Wang

VITA

Yong Wang received his bachelor's degree from Anhui Agricultural University, Hefei City, China in 1986. He received a master's degree from Anhui Agricultural University, Hefei, China in 1989. He graduated from Texas A&M University with a master's degree in computer science in May 2000, then started full-time employment as a software engineer at Nortel Networks in Richardson, Texas. From the summer of 2001 to September 2006, he worked as a senior network system analyst at Information and Computer Services, Texas A&M University. He became a full-time, Ph.D. student in September 2006. He can be reached at the following address:

Yong Wang

Department of Computer Science

Texas A&M University

College Station, TX 77843-3122