THESIS

AN ANALYSIS OF VULNERABILITIES IN WEB SERVERS AND BROWSER USING TIME-BASE AND EFFORT-BASED MODELS

Submitted by

Sung-Whan Woo

Department of Computer Science

In partial fulfillment of the requirements

For the Degree of Master of Science

Colorado State University

Fort Collins, Colorado

Fall 2006

ABSTRACT OF THESIS

AN ANALYSIS OF VULNERABILITIES IN WEB SERVERS AND BROWSER USING TIME-BASE AND EFFORT-BASED MODELS

With the rapid increase in the number of vulnerabilities discovered in major software systems, security in computing and internet-based transactions is greatly threatened. These vulnerabilities can be exploited to damage a computer system's security attributes - confidentiality, integrity and availability. The known vulnerabilities are used by viruses, worms, spywares and individuals with malicious intentions. The vulnerabilities in web applications cover a major fraction of all reported vulnerabilities. Web servers and browsers are at the core of web applications. However, detailed analysis of vulnerabilities in the web applications has not been attempted in the past. In this research vulnerabilities are studied using the Vulnerabilities Discovery Models (VDMs), categorization of vulnerabilities by origin and severity and the concept of vulnerability density. The feasibility of quantitatively characterizing the vulnerabilities in the two major HTTP servers and three popular web browsers is examined in this research. In particular, we investigate the applicability of quantitative empirical models to the vulnerabilities discovery process for these servers and browsers. Such models allow us to predict the number of vulnerabilities that may potentially be present in a server or browser but may not yet have been found. The data on vulnerabilities found in the two servers and three browsers is mined and analyzed. We explore the applicability of both time-based and effort-based vulnerability discovery models. This investigation shows that both types of vulnerabilities discovery models can fit the data for servers and browsers well. We also investigate the applicability of the models for the two separate classification schemes for server and browser vulnerabilities, one based on the source of error and the other based on severity, and demonstrate the applicability of the quantitative methods to individual classes.

Sung-Whan Woo
Department of Computer Science
Colorado State University
Fort Collins, Colorado 80523
Fall 2006

ACKNOWLEDGMENTS

Completing a master degree is truly good experience, and I would not have been able to complete this journey without the aid and support of countless people over the past two years. I must first express my gratitude towards my advisor, Professor Yashwant K. Malaiya for his guideance in the creation of this document, as well as the rest of the Colorado State University Computer Science department for their continued dedication to my education.

I would like to thank my father and mother for all of the love, support, and encouragement.

I also would like to thank some of my fellow students DaeGon Kim, Omar H. Alhazmi and Jin-Yoo Kim.

Fianlly, many thanks to my patient and loving wife Jina Lee, and two children Warren Woo and Elizabeth Woo, who have been a great source of strength all through this work.

TABLE OF CONTENTS

1 Introduction **1**

1.1 Vulnerability Discovery Trend 2

1.2 Vulnerability, Malicious Intent and Economics 3

1.3 Web Servers and Web Browsers 5

1.4 Analysis of Vulnerabilities in Web Servers and Web Browsers 7

2 Vulnerability Discovery Models **9**

2.1 Time-based Model 10

2.1.1 Alhazmi-Malaiya Logistic Model 11

2.1.2 The Linear Model 12

2.2 The Alhazmi-Malaiya Effort-based Model 13

2.3 Limitations of Each Model 15

3 Three Significant Factors Impacting Vulnerability **17**

3.1 Software Code Size 17

3.2 Software Age 18

3.3 Installed-based Systems 20

3.4 Relationship between Installed Systems and Software Age 20

4 Vulnerabilities in Web Servers **22**

4.1 Web Server Market Share for the Effort-based Model 23

4.2 Aggregate Vulnerabilities in Web Servers 25

4.2.1 Modeling Apache Vulnerabilities 25

4.2.2 Modeling IIS Vulnerabilities 27

4.2.3 Chi-square Analysis of Goodness of Fit for Aggregate Vulnerabilities in Web Servers 28

4.3 Individual Vulnerability Categories 29

4.4 Modeling Vulnerabilities by Severity 34

4.5 Discussion for HTTP Server Vulnerabilities and Models 41

5 Analysis of Vulnerabilities in Web Browsers **43**

5.1 Web Browser Market Share 45

5.2 Aggregate Vulnerabilities in Web Browsers 46

5.3 Individual Vulnerability Categories 52

5.4 Vulnerability Severity Levels 57

5.5 Discussion for Web Browsers' Vulnerability and Models 62

6 Discussion **66**

7 Conclusions **69**

REFERENCES **71**

Chapter 1

Introduction

In recent years, the number of vulnerabilities found has increased rapidly. The concern over the potential impact of serious security problems is rapidly increasing as well. This makes it necessary for organizations and individual users to spend significant time and resources to contain the potential damage due to security problems. Many companies have incurred significant financial losses occurred through theft of proprietary information. Viruses and cracking attacks are increasing targeting not just commercial or government systems but also individual internet users. The National Vulnerability Database (NVD) [15] and Symantec Internet Security Threat Report [54] show that many published vulnerabilities migrate from server software to client software, such as web browsers. A vulnerability may arise due to defects in software and hardware. A software vulnerability is defined as "a defect which enables an attacker to bypass security measures" [47] or "a weakness in the security system that might be exploited to cause loss or harm" [41]. In general, vulnerability is class of defect or error that can violate security policy.

There has been considerable discussion about system security problems. While many techniques and optimizations approaches are aimed to preventing specific attacks or malicious behaviors, much of the examination has been qualitative, often focused on detection and prevention of individual vulnerabilities. Quantitative data is sometimes cited [13, 19, 20, 29], but without any significant critical analysis. Methods need to be developed to allow security-related risks to be evaluated quantitatively in a systematic manner. This study focuses on the vulnerabilities trends in HTTP servers and web browsers as a group rather than on the specific vulnerabilities.

1.1 Vulnerability Discovery Trend

Vulnerabilities are reported and collected by several databases. Four major vulnerability databases are the National Vulnerability Database (NVD) [15], the Computer Emergency Response Team Coordination Center (CERT/CC) [14], the Open-Source Vulnerability Database (OSVD) [16] and the Symantec Vulnerability Research [44]. The number of vulnerabilities that has been collected into these four major databases has differences since each vulnerability database has their own definition and vulnerability reporting system is different. However four major databases show that the vulnerabilities discovery rate is rapidly growing year by year. More than twelve vulnerabilities were found during each day of 2005. Figure 1.1 presents data obtained from NVD [15], showing the number of all reported vulnerabilities by year from 1994 to 2005. Although during the middle of 1990 just a few vulnerabilities were found, the overall number of vulnerabilities is increasing dramatically. For example, comparing the 1994 and 2005 vulnerabilities found, more than 189 times the vulnerabilities were found in 2005. Figure 1.1 indicates that more and more vulnerabilities will most likely be found in the future.

Increasing vulnerabilities is a serious problems because such a large number of vulnerabilities provides a good exploitation opportunities for viruses, malicious codes, and attackers. We can foresee that more viruses, worms and spywares will be employed in the near future. Moreover, malicious attackers have a greater chance of attacking systems with such published and known vulnerability.

The available data indicates that client software's vulnerabilities are more frequently

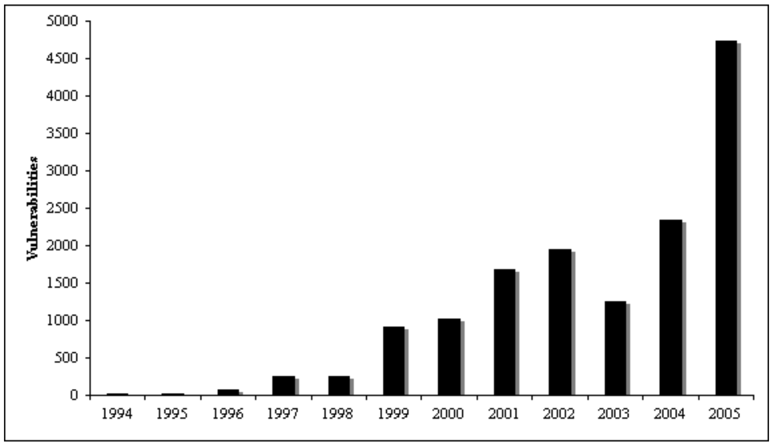

Figure 1.1: Vulnerabilities Discovery Trend

reported than server softwares vulnerabilities. According to the National Vulnerability Database [15], approximately 60% of all reported vulnerabilities in early 2000 were related to server software. However, most currently reported vulnerabilities (over 60% of all reported vulnerabilities) are related to client software, especially web applications such as web browsers, instant messenger program, etc [54].

1.2 Vulnerability, Malicious Intent and Economics

The computer virus is a good example of how software vulnerability can be exploited. Computer security problems surfaced before computer networking had become popular. The first computer virus, *Elk Cloner*, was found in the Apple system in 1982. This virus spread slowly and was limited because the infection had been transmitted from a floppy disk to a floppy disk. This virus, like many other early viruses, also was not harmful to systems. However, recent virus or malicious codes using software or hardware vulnerabilities pose a significant risk, and the time-line between disclosure of vulnerability and impact to system is becoming shorter [34].

Computer viruses, worms and spywares spread and attack through computer software

Year	1997	1998	1999	2000	2001	2002	2003	2004	2005
Incidents	2,134	3,374	9,859	21,756	52,658	82,094	137,529	N/A	N/A
Vulnerabilites	252	246	916	1016	1672	1948	1264	2349	4827

Table 1.1: Computer Security Incidents and Vulnerabilities

vulnerabilities or flaws. For example, Code Red [35], which exploited a buffer overflow vulnerability in IIS Indexing Service DLL (described in Microsoft Security Bulletin MS01-033, June 18, 2001), appeared on July 13, 2001, and soon spread world-wide in unpatched systems. Nimda, which was spread to IIS and Windows 95, 98, ME, NT and 2000 using a buffer overflow vulnerability (described in Microsoft Security Bulletin MS00-078, October 17, 2000), appeared on September 18, 2001, and also widely infected unpatched Windows systems. As we can see from the above examples, there were sufficient time to patch the system's vulnerabilities; Microsoft had released the patch for Code Red 28 days and Nimda 336 days earlier. This demonstrates that most viruses and worms exploit known vulnerabilities. Table 1.1 present a number of computer security incidents obtained from CERT [14] since 1994. During ten years, there has been a 57-fold increase in computer security incidents related to viruses and system vulnerabilities.

These computer security incidents exert a harmful influence on economics. Figure 1.2 shows the relationship between vulnerability, malicious intent and economics. One vulnerability can be used for a variety of malicious intents, such as virus, worm, cracking, etc, consequently, one malicious intent-related incident can cause economic loss for either companies or individual users. Thus, reducing vulnerabilities can minimize economic loss.

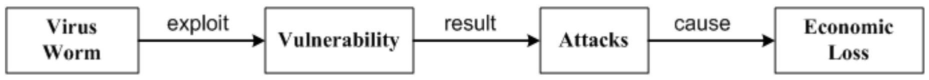

Figure 1.2: Relationship Between Vulnerability, Malicious Intents and Economics

Year	1997	1998	1999	2000	2001	2002	2003	2004	2005
Total No. of Vulns	252	246	916	1016	1672	1948	1264	2349	4827
IIS	3 (1.19%)	3 (1.22%)	32 (3.49%)	23 (2.26%)	22 (1.32%)	28 (1.44%)	5 (0.04%)	3 (0.13%)	3 (0.06%)
Apache	3 (1.19%)	1 (0.41%)	8 (0.87%)	7 (0.69%)	12 (0.72%)	19 (0.98%)	14 (1.11%)	20 (0.85%)	8 (0.17%)
IE	5 (1.98%)	6 (2.44%)	39 (4.26%)	19 (1.87%)	34 (2.03%)	53 (2.72%)	21 (1.66%)	58 (2.47%)	32 (0.66%)
Firefox	N/A	N/A	N/A	N/A	N/A	N/A	N/A	22 (0.94%)	74 (1.53%)
Mozilla	N/A	N/A	N/A	1 (0.1%)	1 (0.06%)	6 (0.31%)	1 (0.08%)	17 (0.72%)	12 (0.25%)

Table 1.2: Web Application Vulnerability Trends

1.3 Web Servers and Web Browsers

There has been considerable discussion on web server and web client software's security in recent years. According to the Symantec Internet Security Threat Report [54], 69% of total vulnerabilities in 2005 were associated with web applications and the percentage of this trend is increasing. Table 1.2 shows the number of vulnerabilities and the percentages of all of published vulnerabilities for two web servers and three web browsers from 1997 to 2005. These five applications cover more than 2.6% of the total number of vulnerabilities for each year. The average percentage of total number of vulnerabilities during this nine years period is 4.78%.

Two of the major software components of the Internet are an HTTP (Hyper Text Transfer Protocol) server (also termed a web server) and a web browser, which serves as the client side. Both of these components were first introduced in 1991 by Tim Berners-Lee of CERN. They have now become an indispensable part of both organizational and personal interactions. The early web server provided information using static HTML pages, while the early web browsers only offered this static and inactive information to clients. The web server now provides dynamic and interactive service between the server and client using database queries, executable scripts, etc. The web server is able to support a variety of functions such as serving streaming media, mail, etc. An HTTP server has thus emerged as a focal point for the Internet. Also, web browsers have the ability to handle such information which servers send.

In this research the vulnerabilities in the two most widely-used HTTP servers and three well-accepted web browser to clients are examined. The Apache server, introduced in 1995, and the Microsoft IIS (Internet Information Services) server originally supplied as part of the NT operating systems in 1995-96. While Apache has a much larger overall market share, roughly 65%, IIS may have a higher share of the corporate websites. The market share for other servers is very small and thus they are not examined here. IIS is the only HTTP server that is not open-source. Both Apache and IIS are generally comparable in features. IIS runs only under the Windows operating systems, whereas Apache supports all the major operating systems.

Today, web browser provide a variety of office and entertainment functions, such as e-commercial, e-mail, flash games, movies, etc. These variety of functions may provide attackers or malicious users and opportunity to widen the Internet security hole. However, these functions cannot be eliminated from web browser.

The first public version of Microsoft IE (Internet Explorer) was released in August, 1995. When IE appeared, only a small number of clients used this web browser. Netscape Navigator, which is Firefox and Mozilla's antecedents, was employed as the Internet client's web browser during middle of the 90s. IE currently holds more or less 80% of the overall market share. Firefox is expanding its area.

Servers and web browsers are very attractive targets for malicious attacks. Servers can represent the first line of defense that, if bypassed, can compromise the integrity, confidentiality and availability attributes of the enterprise security. Web browsers are the first gate that common Internet users utilize to connect to the world wide web. This first gate can also be the crack that leak a client's personal information, such as credit card number, e-mail address, and so on. Thus, it is essential to understand the threat posed both by recently discovered vulnerabilities for which a patch has not been developed or applied and undiscovered vulnerabilities.

At this time, despite the significance of security in the HTTP servers and web browsers, very little quantitative work has been done to model the vulnerabilities discovery process for the servers. Such work would permit both the developers and the users to better estimate future vulnerabilities discovery rates. It would also be highly desirable to be able to project what types of vulnerabilities are more likely to be discovered. Some of the available work on HTTP servers and web browsers discuss some specific problem or attacks that the servers and web browsers face, such as denial of service attacks (DoS) [9, 26], the author suggests some countermeasures to be applied when an attack of this type takes place. In this research, the focus is rather on all kinds of vulnerabilities than on vulnerability trends of HTTP severs and web browsers .

1.4 Analysis of Vulnerabilities in Web Servers and Web Browsers

The security of systems connected to the Internet depends on several components of the system. These include the operating systems, HTTP servers and the browsers. Some of the major security compromises arise because of vulnerabilities in the HTTP servers and web browsers. The vulnerabilities found are disclosed by the finders using some of the common reporting mechanisms available in the field. The databases for the vulnerabilities and defects are maintained by organizations such as National Vulnerabilities Database [15], MITRE , Bugzilla [55], BugTraq [49], etc., as well as the developers of the software.

All computing systems connected to the network are subject to some security risks.

However, there has been little research carried out to analyze vulnerability in web servers and web browsers. The existing research merely shows quantitative data of vulnerability without any significant analysis. In this research, a variety of approaches have been chosen to analyze web servers' and browsers' vulnerabilities.

While there have been many studies attempting to identify causes of vulnerabilities and potential counter-measures, the development of systematic quantitative methods to characterize security has begun only recently. There has been considerable debate comparing the security attributes of open source and commercial software [6]. However, for a careful interpretation of the data, rigorous quantitative modeling methods are needed. The likelihood of a systems being compromised depends on the probability that a newly discovered vulnerability will be exploited. Thus, the risk is better represented by the not yet discovered vulnerabilities and the vulnerabilities discovery rate rather than by the vulnerabilities that have been discovered in the past and remedied by patches.

Possible approaches for a quantitative perspective are presented in Sarah Brocklehurst et al. 's "On measurement of operational security"[12] and Less Hatton's "Reexamining the fault density-component size connection" [24]. Probabilistic examinations of intrusions have been presented by several researchers Hilary K. Browne et al.'s "A trend analysis of exploitations" [13] and Bharat B. Madan et al.'s "A method for modeling and quantifying the security attributes of intrusion tolerant systems" [32]. In [42], Rescorla studied vulnerabilities in open source software; however, the vulnerabilities discovery process in operating systems has just recently been examined by Rescorla [43] and by Alhazmi and Malaiya [2, 3, 5].

Chapter 2

Vulnerability Discovery Models

Use of Reliability Growth Models is now common in software reliability engineering [37]. As bugs are found and removed, fewer bugs remain. As a result, the bug finding rate gradually drops and the cumulative number of bugs eventually approaches saturation point, which means that only a limited number of potential or undiscovered bugs remain. Such growth models are used to determine when a software system is ready to be released, and what failure rates can be expected in actual use.

Reliability Growth Model is closely related to the Vulnerability Discovery Models since vulnerabilities are a special class of defects or bugs that can permit circumvention of the security measures. Some vulnerability discovery models were recently proposed by Anderson [6], Rescorla [43], and Alhazmi and Malaiya [3]. Most of these models maintain the cumulative number of vulnerability by calendar time. These models are distributed into the time-based model. The time-based model considers calendar time as the independent variable. This model incorporates the effect of the rising and declining market share on the software. The other model, which is the effort-based model, requires explicit estimation of the effort using an effort function, which is then used as an independent variable. Figure 2.1 shows the distribution of vulnerability discovery model (VDM). Vulnerability discovery models are separated into the time-based model and effort-based model. The time-based

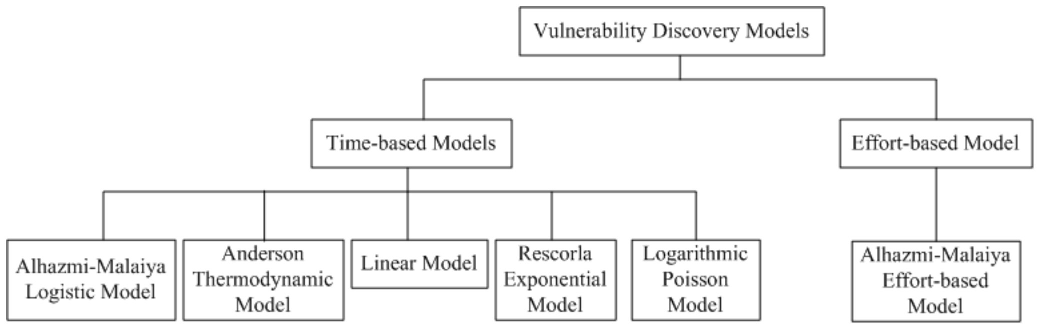
Vulnerability Discovery Models
Time-based Models
Effort-based Model
Alhazmi-Malaiya Logistic Model
Anderson Thermodynamic Model
Linear Model
Rescorla Exponential Model
Logarithmic Poisson Model
Alhazmi-Malaiya Effort-based Model

Figure 2.1: Vulnerability Discovery Model Distribution

model uses calendar time as the main factor and the effort-based model uses installed system as main factor. Example of the time-based models are the Alhazmi-Malaiya Logistic Model, Anderson Thermodynamic Model, Linear Model, Rescorla Exponential Model and Logarithmic Poisson Model.

The applicability of these models to several operating systems was examined in [2]. The results indicate that while some of the models fit the data for most operating systems, others do not fit well or provide a good fit only during a specific phase. We investigate the applicability of two of the most successful models for HTTP servers [57] and web browsers. The models used here are the Alhazmi-Malaiya Logistic Model and Alhazmi-Malaiya Effort-based model proposed by Alhazmi and Malaiya [3]: These two models have been found to fit data sets for several of the major Windows and Linux operating systems, as determined by goodness of fit and other measures. In this thesis, the *Alhazmi-Malaiya Logistic Model* is called the *Time-based model* and the *Alhazmi-Malaiya Effort-based model* is called the *Effort-based model* since these two models provide more accurate fit.

2.1 Time-based Model

The time-based model shows the variation of cumulative number of vulnerabilities with time. In Figure 2.1, Alhazmi-Malaiya Logistic, Logarithmic, Exponential and Linear models examine the cumulative number of vulnerability with calendar time. Each model demonstrates vulnerability discovery trend-shape, for exampel, the Alhazmi-Malaiya Logistic model suggests that the cumulative number of vulnerabilities demonstrates an S-shape with time and the logarithmic model demonstrates logarithmic-shape with time. However Alhazmi-Malaiya Logistic and Linear models provide the more accurate fit than other models. The next sections present a summary of the Alhazmi-Malaiya Logistic and Linear models' main feature.

2.1.1 Alhazmi-Malaiya Logistic Model

This model assumes that the rate of change of the cumulative number of vulnerabilities Ω is governed by two factors, as given in equation 2.1 [3]. The first factor declines as the number of remaining undetected vulnerabilities declines. The other factor increases with the time needed to take into account the rising share of the installed base. The saturation effect is modeled by the first factor. While it is possible to obtain a more complex model, this model provides a good fit to the data, as shown below. Let us assume that the vulnerability discovery rate is given by the differential equation:

$$\frac{d\Omega}{dt} = A\Omega(B - \Omega) \tag{2.1}$$

where Ω is the cumulative number of vulnerabilities, t is the calendar time, and initially $t = 0$. A and B are empirical constants determined from the recorded data. By solving the differential equation, we obtain

$$\Omega(t) = \frac{B}{BCe^{-ABt} + 1} \tag{2.2}$$

where C is a constant introduced while solving Equation 2.1. Equation 2.2 gives us a three-parameter model given by the logistic function. In Equation 2.2, as t approaches

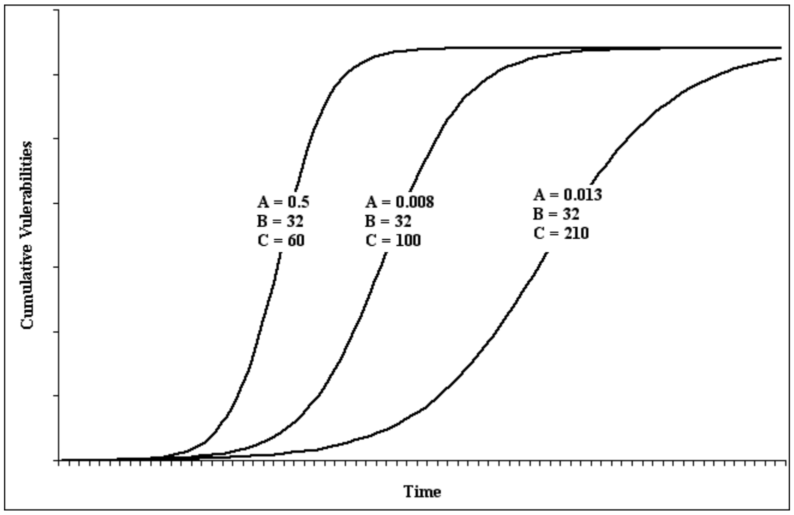

Figure 2.2: Alhazmi-Malaiya Logistic (AML) Model

infinity, Ω approaches B. Thus, the parameter B represents the total number of accumulated vulnerabilities that will eventually be found.

Figure 2.2 shows hypothetical plots for the Time-based model for different values of A, B and C. Thus, the vulnerability discovery rate increases at the beginning, reaches a steady rate and then starts declining. Consequently, the cumulative number of vulnerabilities shows an increasing rate at the beginning as the system begins to attract an increasing share of the installed base. After some time, a steady rate of vulnerability finding yields a linear curve. Eventually, as the vulnerability discovery rate begins to drop, there is saturation due both to reduced attention and a smaller pool of remaining vulnerabilities.

2.1.2 The Linear Model

The Linear Vulnerability Discovery (LVD) Model was proposed by Alhazmi and Malaiya [4]. LVD also uses a calendar time base like that of the time-based model. LVD assumes the vulnerability discovery rate to be steady during software life time. However, it is difficult to apply LVD to long-aged software or long-term life cycle software since the vulnerability discovery rate varies for each time period and the number of vulnerabilities in each software is finite. This model should be applied to the linear phase part of the Time-based model and the newly released software, which has a prior version.

The linear model is given by a linear equation,

$$\Omega(t) = p + qt \tag{2.3}$$

where q is the slope and p is a constant factor. The LVD model may fit cases in which much of the data is linear and largely falls within the linear phase in the time-based model. Also statistical analysis in [2] shows that the LVD model's goodness of fit is less significant than that of Time-based and Effort-based models. This means that it is difficult to apply the LVD model to real vulnerability data.

2.2 The Alhazmi-Malaiya Effort-based Model

Vulnerabilities are usually reported using calendar time as the main factor. The reason for this is that it is easy to record vulnerabilities and link them to the time of discovery. This, however, does not take into consideration the changes occurring in the environment during the lifetime of the system. A major environmental factor is the number of installations, which depends on the share of the installed base of the specific system. It is much more rewarding to exploit vulnerabilities that exist in a large number of computers. Hence, it can be expected that a larger share of the effort going into discovery of vulnerabilities, both in-house and external or non-experts and experts, would go toward a system with larger installed base.

Using effort as a factor was first discussed in [12, 31]. However, the authors did not suggest a unit or a way of measuring effort. The Effort-based Model utilizes a measure termed Equivalent Effort (E), which is calculated using

$$E = \sum_{i=0}^{n}(U_i \times P_i) = \sum_{i=0}^{n} N_i \tag{2.4}$$

where U_i is the total number of all HTTP servers or web browsers at the period of time i, n represents the last period of usage time, and P_i is the percentage of the servers using the specific server for which we are measuring E. The result is given in system-months. The measure E can be calculated for the servers using the data available at the National Vulnerability Database (NVD) [15] and for the web browsers using the data available at the Net Applications [7]. These data provide each HTTP server's and web browser's market share for each time period.

The model employs equivalent effort as a factor to model vulnerability discovery. Equivalent effort reflects the effort that would have gone into finding vulnerabilities more accurately than using time alone. This is somewhat analogous to using CPU time for software reliability growth models (SRGMs).

If we assume that the vulnerability detection rate with respect to effort is proportional to the fraction of remaining vulnerability, then we get an exponential model, just like the exponential SRGM. The model can be given as follows:

$$\Omega(E) = B(1 - e^{-\lambda_{vu}}) \tag{2.5}$$

where λ_{vu} is a parameter analogous to failure intensity in SRGMs and B is another parameter. B represents the number of vulnerabilities that will eventually be found. We will refer to the model given by Equation 2.5 as the Effort-based Model.

Figure 2.3 shows hypothetical plots for the Effort-based model for different values of B and λ_{vu}. Thus, the vulnerability discovery rate increases rapidly at the beginning while the cumulative number of system increase. Later, the vulnerability discovery rate begins to decline. Consequently, the cumulative number of vulnerabilities shows an increasing

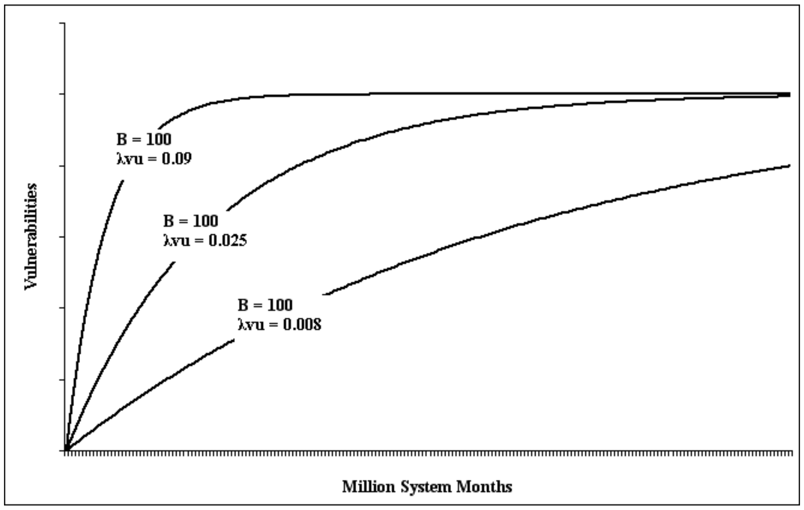

Figure 2.3: Alhazmi-Malaiya Effort-based Model

rate at the beginning as the system starts to attract an increasing share of the installed base. After some time, the steady rate of vulnerability finding is drop quickly. Finally, as the vulnerability discovery rate is close to the zero, there is saturation due to the smaller number of undiscovered vulnerabilities.

2.3 Limitations of Each Model

Three vulnerability discovery models are shown above. The Time-based and Linear Vulnerability Discovery models employ calendar time as the independent variable, while the Effort-based model uses the share of the installed base of the specific system as the major environmental factor. Thus, the Effort-based model can be expected to be much more effective in discovering vulnerabilities, since a software that is shared by large number of explorer have a greater chance of being exploited. In general, more vulnerability is found in a application or O.S that has a higher market share, since it does not depend on software lifetime.

Two statistical goodness of fit tests evaluate the Time-based and Effort-based models.

The first is the chi-square (x^2) goodness of fit test, which compares a observed data to a model's expected data and shows difference significant. The chi-square statistic formula as follows:

$$x^2 = \sum_{i=0}^{n} \frac{(x_i - e_i)^2}{e_i} \tag{2.6}$$

In the Equation 2.6, x_i is the observed value and e_i is the model's expected value. To be the acceptable significant difference, chi-square value is less than chi-square critical value for a given probability level (alpha) and the degrees of freedom. The P-value of a statistical significance test represents the probability that $x^2_{critical}$ values should be equal to or greater in magnitude than x^2. The acceptable P-value range is higher than 0.05 since we use an alpha level of 5%. A P-value closer to 1 that indicates a better fit. Except the Time-based model (Alhazmi-Malaiya Logistic Model) and Effort-based model, models in Figure 2.1 do not fit quite well [2].

In spite of the fact that the Effort-based model can provide more accurate fit, it cannot be applied to every software because the number of specific software installed systems cannot always be acquired and hence it is difficult to obtain accurate data concerning the number of installed system for specific software. The Time-based model is a better fit when the number of installed instances is not known. However, the results after examining each HTTP server and web browser shows similar accurate fitting if the software has enough market share: hence, if a software has enough market share, one of models shows similar results. In this study, the Time-based and Effort-based models are provided for HTTP servers and web browsers. Moreover we assume that Apache, IIS, IE and Firefox have sufficient market share data to be explored.

Chapter 3

Three Significant Factors Impacting Vulnerability

Many factors produce an effect on the vulnerability discovery rate. Even the most expert and the best programmer or team is unable to develop a perfect, secure and defect or error free software. This means that all software has defects, errors and security flaws, all of which can be used for malicious purposes. This is called software vulnerability. Nobody can state with any degree of certainty how many vulnerabilities may be involved in a software or how many vulnerabilities likely to be discovered monthly or in the near future. However, we can predict vulnerability trends through a variety factors such as popularity of software, software age, code size of software, relationship to other components, economic, value, etc. The major three factors that influence software vulnerability are software code size, software age and software popularity. The first factor, software code size, indicates the potential number of vulnerabilities, and last two factors, software age and popularity, provide the clues for vulnerability discovery rate.

3.1 Software Code Size

Several studies [1, 18, 25, 45] have researched the prediction of software defects and the relationship between code or module size and defects or bugs. Even though the largest size of software has lower *defect density* (the number of defect is divided by lines of code (LOC)), these studies show that the number of defects or errors increases as code size increases. As this aspect, the number of vulnerabilities increases as code size increases, since software vulnerability is defined as "a defect which enables an attacker to bypass security measures" [47]. A first order approximation assumes a linear relationship, which allows a measure of defect density to be defined. Since vulnerabilities are a class of defects, we can similarly define a measure called *vulnerability density* (the number of vulnerabilities is divided by LOC). Available data allows us to calculate the densities of the discovered vulnerabilities for the Apache web server, Firefox and Mozilla, as shown following chapters.

The linear equation for the vulnerability growth with code size is given,

$$\Omega(l) = p + q \times L \tag{3.1}$$

where q is the potential vulnerability rate and p is a constant factor. the parameter q varies with various factors such as size of module, complexity and so on.

Equation 3.1 differs with the Linear Vulnerability Discovery model Equation 2.3. The Linear Vulnerability Discovery model Equation 2.3 in previous section corresponds to the cumulative vulnerability that corresponds to time, since Equation 3.1 presents a number of potential vulnerabilities related to software code size.

The number of vulnerabilities should be increased in proportion to the size of the software. However, this does not imply that the vulnerability discovery rate grows with the code size of the software but merely demonstrates shows an association between potential vulnerabilities and software code size.

3.2 Software Age

In the previous section, we showed the Time-based model that was proposed by Alhazmi and Malaiya. The cumulative number of vulnerabilities for software increases with the lapse

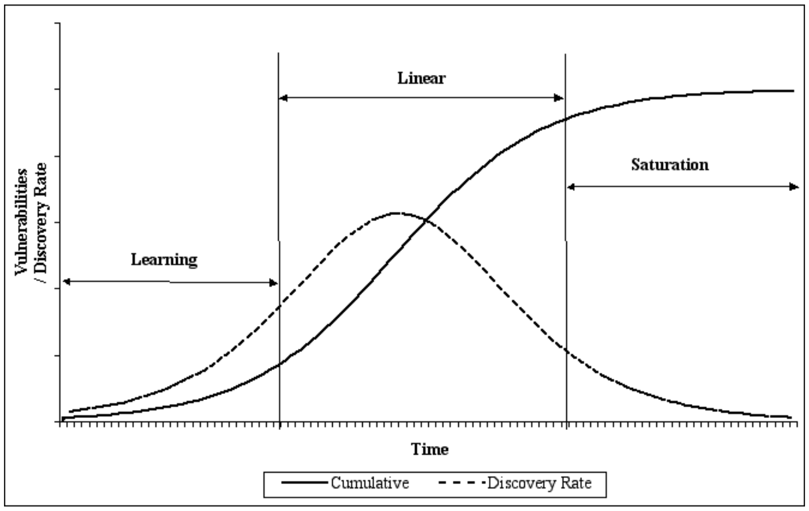

Figure 3.1: Relation between Software Age and Vulnerabilities

of time and then eventually reaches saturation phase. During the initial learning phase in Time-based model, very few of vulnerabilities are found. This indicates that the vulnerability discovery rate is low in this phase. During the next phase, termed the linear phase, a steady stream of vulnerability discoveries occurs; the number of found vulnerabilities increases rapidly until saturation phase; the vulnerability discovery rate show a bell shape during this phase. In the final saturation phase, the finding vulnerability discovery rate declines and few vulnerabilities are found since there are few residual vulnerabilities. Figure 3.1 shows this cycle.

The relationship between software age and number of found vulnerabilities follows the Time-based model since the Time-based model demonstrates a cumulative number of vulnerabilities by calendar time.

Figure 3.1 shows the relationship between software age, the cumulative vulnerabilities Ω and the vulnerability discovery rate. The durations of the three phases in Figure 3.1 should be shrunk or expanded depending on other factors such as market share, potential number of vulnerabilities, etc.

3.3 Installed-based Systems

Market share is one of the most significant factors impacting the effort expended in exploring potential or residual vulnerabilities. A higher market share indicates more incentive to explore and exploit vulnerabilities for both experts and non-experts, since both would find it more profitable or satisfying to spend their time on a software with a higher market share. The effect of the market share rise and fall is implicit in the Time-based model and Effort-based model. These two model cannot applied to a software that has a lower market share.

A significant number of vulnerabilities have been found in Apache, IIS, IE and Firefox, illustrating the impact of the market share on the motivation for exploiting or finding vulnerabilities. We can use market share as an indicator of effort for the Effort-based model. However we did not use the effort-based model for Mozilla. Only thirty-nine vulnerabilities were found in Mozilla during seven years. Over half of Mozilla's total vulnerabilities are shared vulnerabilities with Firefox. This demonstrates that the vulnerability discovery rate is related more to the amount of usage or market share than to software age. More detail explains are shown in Chapter 4.1 for HTTP server's market share and 5.1 for web browser's market share with real data.

3.4 Relationship between Installed Systems and Software Age

Figure 3.2 is derived from the Time-based model and installed systems. Figure 3.2 shows the cumulative number of vulnerabilities by time and installed system. The equation can be given by follows:

$$\Omega(Sys,t) = \frac{B}{BCe^{-ABt}+1}DSys \tag{3.2}$$

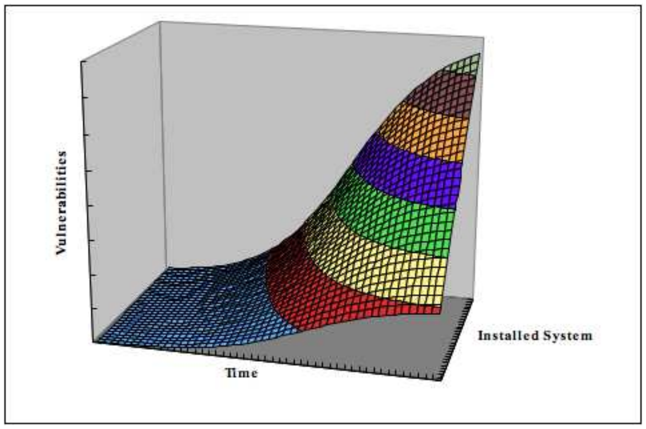

Figure 3.2: Market Share and Software Age

where *Sys* represents the number of installed system, B is the number of vulnerabilities that will eventually be found, and C and D are constant factors.

Figure 3.2 shows the number of installed systems give more major effect to vulnerabilities discovery than software age. Early time of software age, number of installed system does not provide big advantage finding vulnerabilities since user (that include malicious user) analysis the software during this early time. After this time, the gaps between number of vulnerability, which lower number of installed systems and higher number of installed systems find, wide is getting wider.

Chapter 4

Vulnerabilities in Web Servers

In this chapter, we examine the vulnerabilities in the two most widely-used HTTP servers, the Apache server, introduced in 1995, and the Microsoft IIS (Internet Information Services) server, originally supplied as part of the NT operating systems in 1995-96. While Apache has a much larger overall market share, roughly 65%, IIS may have a higher share of corporate websites. The market share for other servers is very small and thus they are not examined here. IIS is the only HTTP server that is not open-source. Both Apache and IIS are generally comparable in features. IIS runs only under the Windows operating systems, whereas Apache supports all the major operating systems.

The data sets for the aggregate vulnerabilities, number of vulnerabilities by categories and severity for the Apache and Microsoft IIS web servers are fitted to the Time-based and Effort-based models. The goodness of fit is evaluated to determine how well the models reflect the actual vulnerabilities discovery process. The vulnerabilities data are from the National Vulnerabilities Database maintained by NIST. The market share data from Netcraft [39] was used. We note that Apache represents an open source software and IIS represents a closed source, i.e., a commercial system. It should also be noted that the number of vulnerabilities, either found or estimated as remaining, should not be the only measurement of a security threat. Other factors such as patch development and application delays and vul-

Web Server	Apache	IIS	SJSWS	Zeus	Others
Market Share	64.76%	25.46%	2.35%	0.67%	6.76%
Vulnerabilities	95	122	3	5	N/A
Release Year	1995	1995	2002	1995	N/A
Latest Version	2.2.0	6.0	6.1	4.3	N/A

Table 4.1: Market Share and Vulnerabilities Found

nerabilities' exploitation rates also need to be considered. In this chapter, all vulnerabilities are considered without regard to how they arise or the extent of their impact. Distinctions among the vulnerabilities will be considered in subsequent chapters.

4.1 Web Server Market Share for the Effort-based Model

Market share is one of the most significant factors impacting the effort expended in exploring potential vulnerabilities. Higher market share indicates more incentive to explore and exploit vulnerabilities for both exports and non-exports, since both would find it more profitable or satisfying to spend their time on a software with a higher market share.

Table 4.1 presents data obtained from NVD and Netcraft, showing the current web server market share and total number of vulnerabilities found to date. As we can see from the table, for servers with a lower percentage of the market, such as Sun Java System Web Server (SJSWS) and Zeus, the total number of vulnerabilities found is low. That does not mean that these systems are more secure, but merely that only limited effort has gone into detecting their vulnerabilities. A significant number of vulnerabilities have been found in both Apache and IIS, illustrating the impact of the market share on the motivation for exploring or finding vulnerabilities. In this study, we use market share as an indicator of effort for the Effort-based model.

Figure 4.1 shows the web server market share for Apache and IIS. As demonstrated by Figure 4.2, the number of web servers continues to grow steadily. Among the various web servers, Apache and Microsoft IIS dominate the web server market. Other web servers such

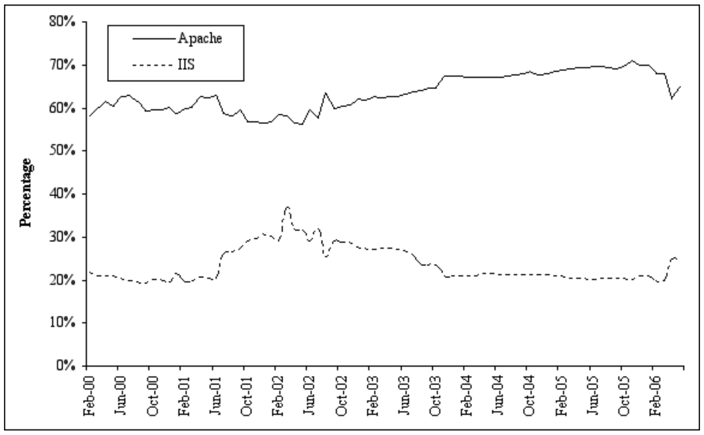

Figure 4.1: Percentage of Market Share

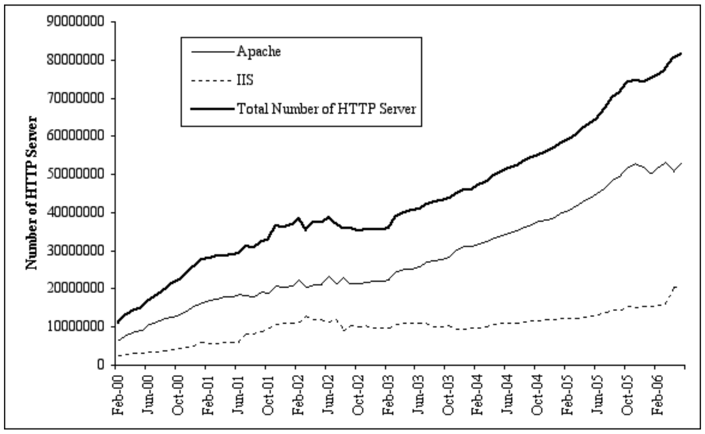

Figure 4.2: Number of Web Servers

as Sun Java System Web Server and Zeus occupy a very small share of the market, as shown in the Table 4.1. Since the total share of all of SJSWS and Zeus added together represents less than 10% of the market share, very few vulnerabilities have been found in them and hence the data for these servers has not been used in our study.

Even though Apache and IIS are the top web servers, there is a marked gap between the Apache and IIS market shares, as shown in Figure 4.1. This difference in market share may be due to several factors. Perhaps the most important of these is that Apache is available for all major operating system platforms and can be obtained without cost. However IIS is only provided for the Microsoft Windows operating system platform. Apache may also have benefited from not having been exposed to serious security issues such as the Code Red [35]or Nimda worms that were faced by IIS in 2001.

4.2 Aggregate Vulnerabilities in Web Servers

A total 95 vulnerabilities were found in Apache until May 2006 and 122 vulnerabilities were published in IIS. In this section we use these vulnerability data for the two major web servers and determine whether the Apache's and IIS's vulnerability trends are fitted for the Time-based and Effort-based models.

4.2.1 Modeling Apache Vulnerabilities

The Apache HTTP server was first released in middle of 1995. Since then it has gained wide popularity and is used by over 50 million web server systems. In this section, we fit the vulnerabilities data for Apache to the Time-based and Effort-based model. Figure 4.3 and 4.4 give the vulnerabilities data from NVD for the period between March 1996 and May 2006, and the Netcraft market share data also coves the period from March 1996 to May 2006.

In Figure 4.3 and 4.4, the bold black lines indicate the fitted models, while the other thin lines show cumulative vulnerabilities for Apache. Figure 4.3 shows cumulative vulnerabilities by time period for the time-based model. At the beginning, the slope of the curve for Apache rises gently until about January 2000, after which the slope has remained steady. From the point of the three phases of the vulnerabilities discovery process [3], Apache has not yet entered the saturation phase; one or two vulnerabilities have still been found recently. Apache currently appears to be at the end of linear phase, since the number of vulnerabilities still appears to be growing linearly. Despite having been on the market for several years, Apache has not reached the saturation phase possibly because of its larger market share;

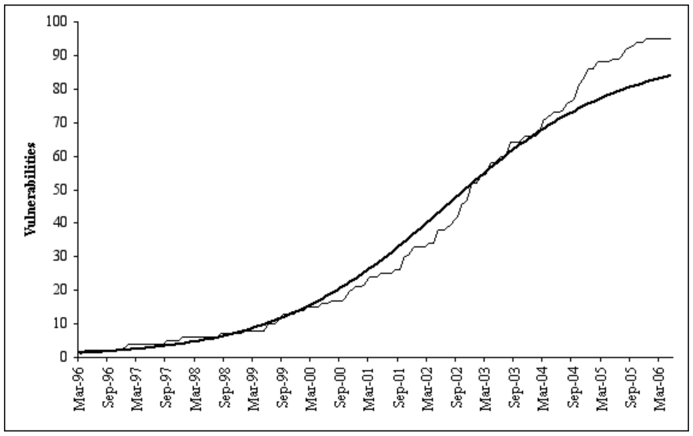

Figure 4.3: Time-Based Model for Aggregate Vulnerabilities in Apache

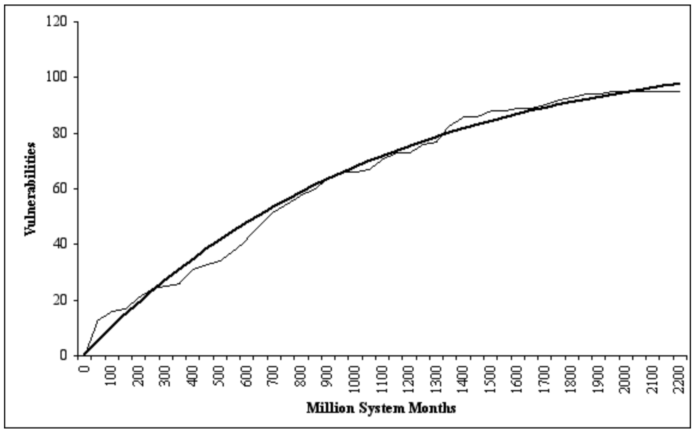

Figure 4.4: Effort-Based Model for Aggregate Vulnerabilities in Apache

moreover, the number of systems using the Apache web server is still increasing. This means that vulnerabilities discovery for Apache can be expected to continue at a significant pace in near future.

Figure 4.4 shows cumulative vulnerabilities by the number of installed Apache system in terms of million system-months and the fitted Effort-based model. The Effort-based model shows that Apache has not yet approached the saturation phase since the number of vulnerabilities continues to increase approximately linearly as the number of Apache severs increases. The results of the analysis are given in Table 4.2.

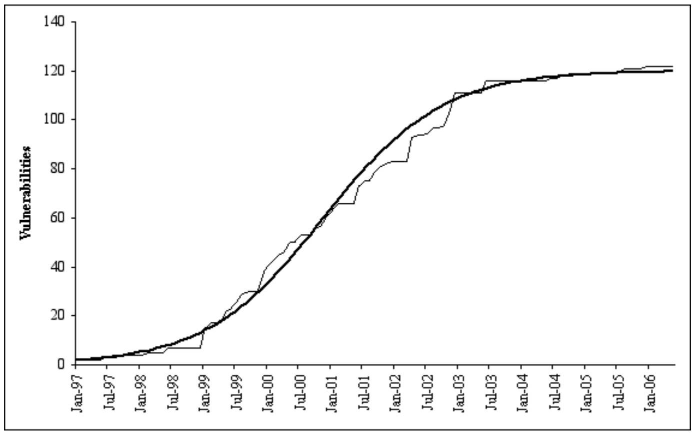

Figure 4.5: Time-based Model for Aggregate Vulnerabilities in IIS

4.2.2 Modeling IIS Vulnerabilities

Microsoft IIS was released in the early part of 1996. IIS is a popular commercial web server with about 15 million installations currently. In this section, we fit the IIS data to the Time-based and Effort-based models. We have used the vulnerabilities data and the market share data from January 1997 to May 2006.

Figure 4.5 shows the cumulative vulnerabilities by month and the fitted time-based model for the IIS web server. The Time-based model and the Effort-based model fit the data for IIS very well. The IIS web server appears to have reached the saturation phase. In recent months, the vulnerabilities discovery rate for IIS has dropped to a very low point. A possible explanation for this can be that the number of IIS web servers installed appears to be stationary, unlike the Apache server which is still gaining in terms of new installations. Another possibility is that the number of remaining undiscovered vulnerabilities may actually have dropped significantly.

Figure 4.6 shows cumulative vulnerabilities by the number of installed IIS web servers and the Effort-based model by million system-months. Unlike Figure 4.4, Figure 4.6 shows a significant degree of saturation.

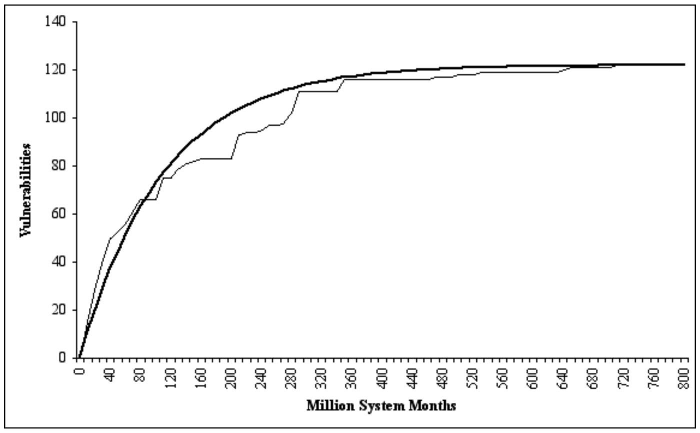

Figure 4.6: Effort-based Model for Aggregate Vulnerabilities in IIS

Model	Parameter	Apache	IIS	Win 98	Win NT4
Time-base Model	A	0.00062	0.00075	0.0048	0.0006
	B	90.01	120	37.73	136
	C	0.7675	0.5959	0.554	0.522
	x^2	64.24	35.54	7.365	35.58
	$x^2_{critical}$	148.78	138.81	60.481	103.01
	P-value	0.999	1	1	1
Effort-base Model	B	112.5	122	37	108
	λ_{VU}	0.00092	0.0009	0.0005	0.003
	x^2	23.726	46.6	3.52	25.05
	$x^2_{critical}$	61.66	103	44.985	42.5569
	P-value	0.992	0.998	1	0.985

Table 4.2: x^2Goodness of Fit Test Result for Aggregate Vulnerabilities

4.2.3 Chi-square Analysis of Goodness of Fit for Aggregate Vulnerabilities in Web Servers

We examine the fit of the models to the data as shown in Figures 4.3, 4.4, 4.5 and 4.6. For x^2 goodness of fit test, we chose an alpha level of 5%. Table 4.2 gives the *chi-square* values and parameter values for both the Time-based model and Effort-based model. For comparison, this table also provide corresponding parameter values for the Windows 98 and NT operating systems, as well as the *chi-square* values.

Table 4.2 shows that the *chi-square (x^2)* values are less than the *chi-square critical*

($x^2_{critical}$) values. This demonstrates that the fit for Apache, IIS, Windows 98 and NT is significant. The fit was obtained by minimizing the *chi-square* value. Both data sets fit both models with P-values ranging from 0.959 to nearly 1, indicating that the fit is quite significant. We can also note that parameter A is always less than 0.005 and parameter C is always less then 0.85, while parameter B corresponds approximately to the number of vulnerabilities.

4.3 Individual Vulnerability Categories

In the previous section we examined the application of the Time-based and Effort-based Models for the total number of vulnerabilities of Apache and IIS. In this and the following subsection, we apply these models to two separate classification schemes for servers' vulnerabilities.

Distinguishing among vulnerabilities is useful when we want to examine the nature and extent of the problem. It can help determine what protective actions would be most effective. Vulnerabilities taxonomy is still an evolving area of research. Several taxonomies have been proposed [8, 10, 28, 51]. An ideal taxonomy should have such desirable properties as mutual exclusiveness, clear and unique definition, and coverage of all software vulnerabilities. Vulnerabilities can be classified using schemes based on cause, severity, impact and source, etc. In this analysis, we use the classification scheme employed by the National Vulnerability Database of the National Institute of Standards and Technology. This classification is based on the causes of vulnerabilities. The eight classes are as follows [15, 49]:

1. Input Validation Error (boundary condition error, buffer overflow error): Such types of vulnerabilities include failure to verify the incorrect input and read/write involving an invalid memory address.

2. Access Validation Error: These vulnerabilities cause failure in enforcing the correct

privilege for a user.

3. Exceptional Condition Error: These vulnerabilities arise due to failures in responding to unexpected data or conditions.

4. Environmental Error: These vulnerabilities are triggered by specific conditions of the computational environment.

5. Configuration Error: These vulnerabilities result from improper system settings.

6. Race Condition Error: These are caused by the improper serialization of the sequences of processes.

7. Design Error: These are caused by improper design of the software structure.

8. Others: Includes vulnerabilities that do not belong to the types listed above, sometimes referred to as nonstandard.

Unfortunately, the eight classes are not completely mutually exclusive. Table 4.3 shows how vulnerabilities are distributed among categories for both the data sets studied. The number of input validation errors is much higher than other types of vulnerabilities for both Apache and IIS. A similar distribution is observed in both operating systems, with input validation errors forming the largest category. Because a vulnerability can belong to more than one category, the summation of all categories for a single software system may add up to more than the total number of vulnerabilities (also the percentages may exceed 100%). This is shown in Table 4.3.

Figure 4.7 compares vulnerabilities distributions in Apache and IIS. The categories with the highest proportions are input validation errors, followed by design errors. There is a slight difference in category ordering between Apache and IIS, with Apache having more configuration errors than access validation errors; however, IIS has more access validation

Category	Apache	IIS	Win 2000	Win XP
IV	42 (37.61%)	59 (45.04%)	113 (44.84%)	88 (55%)
DE	22 (20.18%)	26 (19.85%)	67 (26.59%)	30 (18.75%)
ECE	18 (16.51%)	15 (11.45%)	45 (17.86%)	27 (16.88%)
AVE	6 (5.5%)	16 (12.21%)	20 (7.94%)	10 (6.25%)
CE	12 (11.01%)	6 (4.58%)	9 (3.97%)	0 (0%)
EE	4 (3.67%)	4 (3.05%)	5 (1.59%)	2 (1.25%)
RCE	2 (1.83%)	1 (0.76%)	1 (0.4%)	3 (1.88%)
Other	4 (3.67%)	4 (3.05%)	3 (1.19%)	0 (0%)
Total	95	122	252	160

Table 4.3: Web Server Vulnerabilities Classified by Category

errors. While IIS has been more vulnerable to access validation errors, Apache's greater vulnerability to configuration errors may be due to the fact that Apache has more complex installation requirements.

When we compare HTTP servers and other software, we find a comparable pattern demonstrating that the input validation error or design error is higher in proportion than other type of error. Otherwise, they are within close range of each other.

We plot the vulnerabilities for the major categories to determine whether there is an observable pattern at the level of individual classes. Since we noted a similar pattern for the uncategorized vulnerabilities (the total number of vulnerabilities), a possible fit was examined. Figures 4.8 and 4.9 show the fit for the Apache in the Time-based and Effort-based models by category, respectively. In Figure 4.8 and 4.9, we only consider the three major categories, examining only: input validation errors, design errors and exceptional handling condition errors since other types of error are too in relation to the low number of

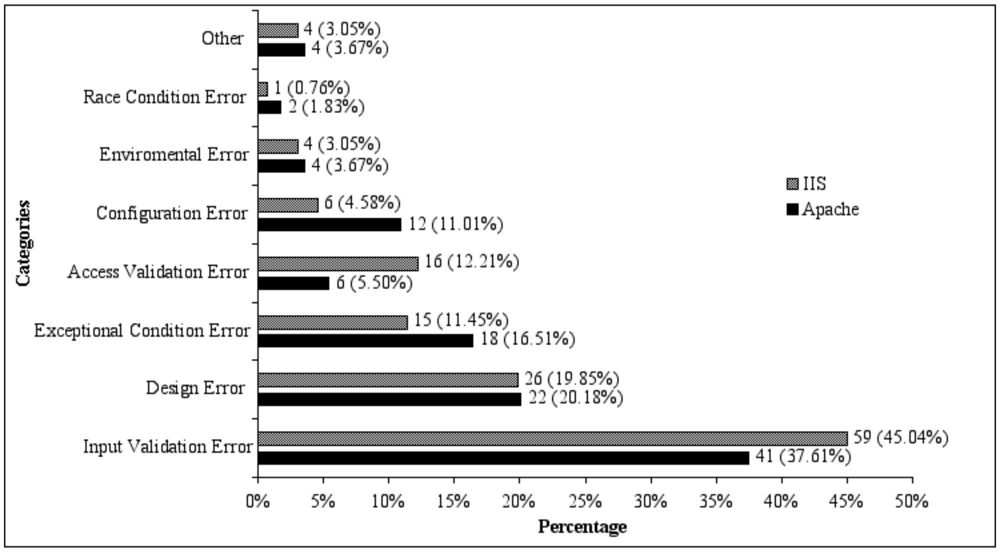

Figure 4.7: Vulnerabilities Distributions in Web Server by Category

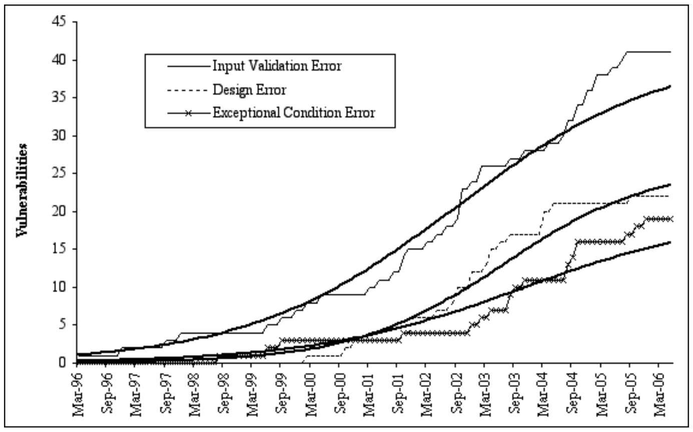

Figure 4.8: Apache Time-based Fitting by Category

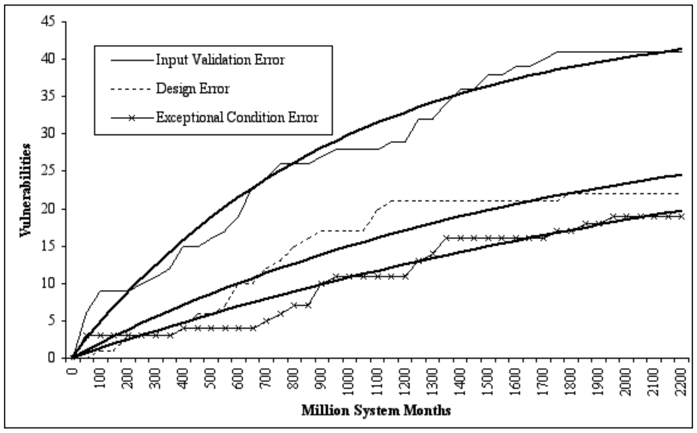

Figure 4.9: Apache Effort-based Fitting by Category

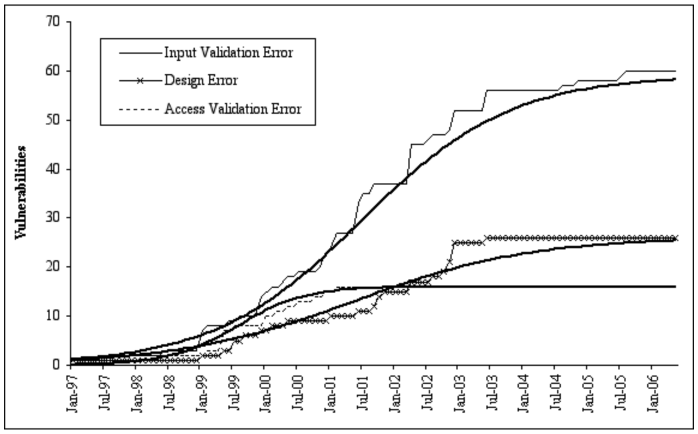

Figure 4.10: IIS Time-based Model Fitting by Category

vulnerabilities to be examined.

Figure 4.10 and 4.11 show the Time-based and effort-based models' fitting of IIS vulnerabilities by category. As we mentioned above, the IIS model has a better fit than the Apache model, since IIS has reached the saturation phase. The categorized number of vulnerabilities shows the same pattern as demonstrated by the total number of vulnerabilities. Thus, each category shows a related pattern with regard to total number of vulnerabilities. Our time-based and effort-based models are fitted for each category. It may be noted that the number of input validation errors and design errors are the most common category in

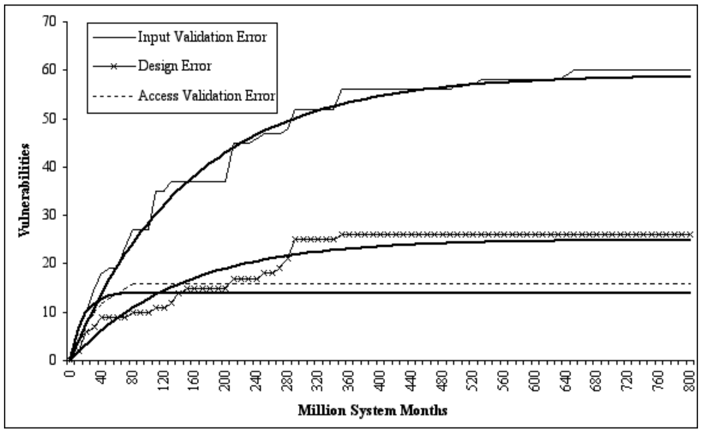

Figure 4.11: IIS Effort-based Model Fitting by Category

Apache and IIS.

Table 4.4 shows the chi-square goodness of fit tests for the Apache and IIS models by category. Table 4.4 demonstrates that the *chi-square* values for each category are less than the critical values. Since chi-square (x^2) values are less than chi-square critical values ($x^2_{critical}$) and the *P-values* are close to 1, the fit of input validation, design and exceptional condition error classes are significant for both models.

4.4 Modeling Vulnerabilities by Severity

Severity is another way of classifying vulnerabilities. The severity of a vulnerability indicates how serious the impact of an exploitation can be. Severity is usually subdivided into three categories; high, medium and low. Some other organizations use three to five level and their own definition for severity. Recently, NVD used CVSS metric for vulnerability severity with ranges from 1 to 10, CVSS uses many factors to determine the severity where the range 1-3.99 corresponds to low severity, 4-6.99 to medium severity and 7-10 to high severity. The National Vulnerability Database of the National Institute of Standards and Technology describes severity levels as follows [15]:

		Apache			IIS		
Model	Parameter	IVE	DE	ECE	IVE	DE	AVE
Time-base Model	A	0.00113	0.00248	0.00238	0.00122	0.00234	0.01
	B	41.06	25.998	19.686	59	26	16
	C	0.902	11.05	3.704	0.8899	1	10
	x^2	49.96	45.08	54.7	27.12	41.14	33.17
	$x^2_{critical}$	148.78	148.78	148.78	138.81	138.81	138.81
	P-value	1	1	0.999	1	1	1
Effort-base Model	B	45.67	34	37.6	59	25	14
	λ_{VU}	0.00105	0.00058	0.00034	0.0065	0.0071	0.0604
	x^2	13.42	14.77	19.3	13.71	21.14	24.66
	$x^2_{critical}$	61.66	61.66	61.66	103	103	103
	P-value	0.999	0.999	0.997	1	1	1

Table 4.4: Apache and IIS's Category Chi-square Analysis of Goodness of Fit

1. High Severity: "This makes it possible for a remote attacker to violate the security protection of a system (i.e., gain some sort of user, root or application account), or permits a local attack that gains complete control of a system, or if it is important enough to have an associated CERT/CC advisory or US-CERT alert. "

2. Medium Severity: "This does not meet the definition of either 'high' or 'low' severity. "

3. Low Severity: "The vulnerability typically does not yield valuable information or control over a system but rather gives the attacker knowledge provides the attacker with information that may help him find and exploit other vulnerabilities or we feel that the vulnerability is inconsequential for most organizations ."

The distributions of the severity of the Apache and IIS vulnerabilities show similarity. About 60% of total vulnerabilities have low severity, followed by about 30% with high severity, with medium severity vulnerabilities at about 4 to 10%. This shows that while low severity vulnerabilities, i.e., those that do not cause serious impact such as system access, DoS attack, exposure sensitive information, etc., constitute the majority, the fraction of high

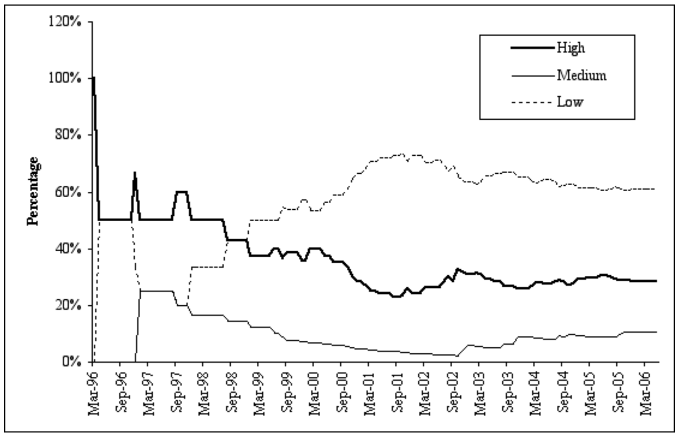

Figure 4.12: Apache Severity Variation

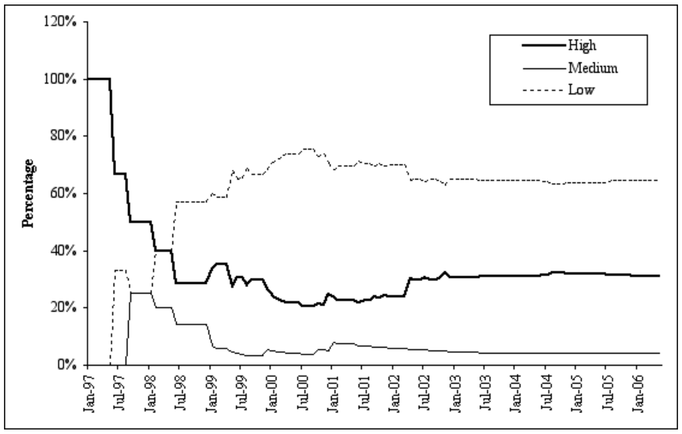

Figure 4.13: IIS Severity Variation

severity vulnerabilities is nevertheless substantial and represents a significant threat to the server. Figure 4.12 and 4.13 plot the Apache's and IIS's percentage of the cumulative number of vulnerabilities for each severity class for each month. Surprisingly, both Apache and IIS show a similar pattern. A large fraction of the high severity vulnerabilities is found early, while the discovery of low severity vulnerabilities is at about 80% after two or three years. Later, high severity vulnerabilities start to form a larger proportion at the expense of low severity vulnerabilities.

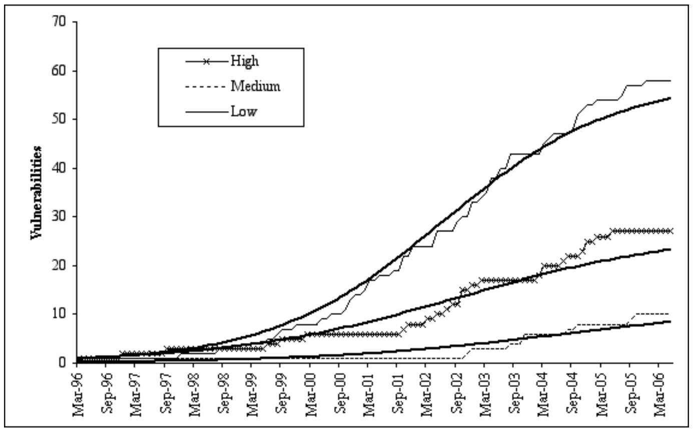

Figure 4.14: Apache Time-based Model Fitting by Severity

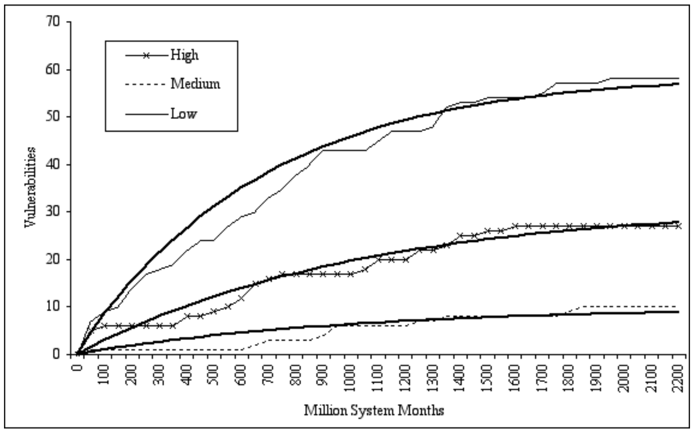

Figure 4.15: Apache Effort-based Model Fitting by Severity

We apply the Time-based and Effort-based model to the three Apache's and IIS's severity classes and observe the each server's severity pattern. In Figures 4.14, 4.15, 4.16 and 4.17, the bold lines indicate the fitted the Time-based and effort-based models for each severity level. Figure 4.14 and 4.15 show the result of fitting the Time-based and Effort-based models to the three severity classes. Figure 4.16 and 4.17 show the fit for the Time-based model and the Effort-based model for IIS's severity classes. In severity classes, the IIS vulnerabilities data had attained the saturation phase; while the Apache's vulnerabilities are still being discovered.

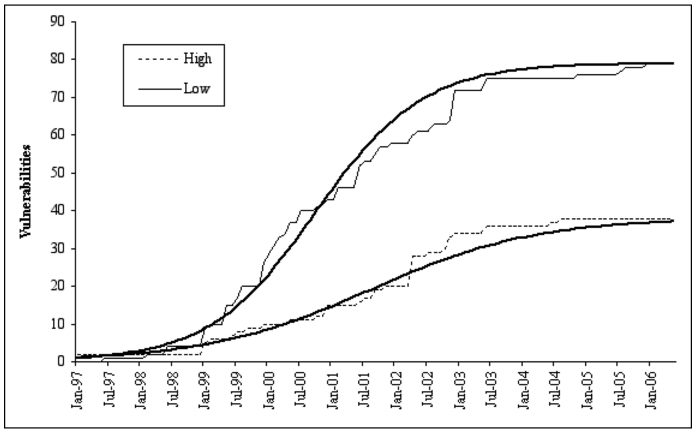

Figure 4.16: IIS Time-based Model Fitting by Severity

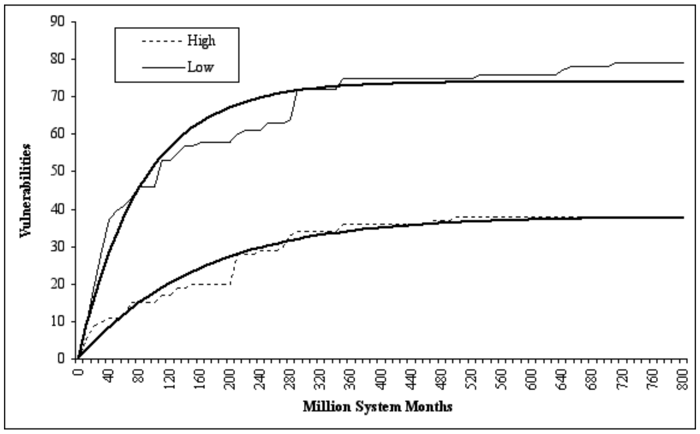

Figure 4.17: IIS Effort-based Model Fitting by Severity

		Apache			IIS	
Model	Parameter	High	Medium	Low	High	Low
Time-base Model	A	0.00155	0.00344	0.00097	0.0017	0.00126
	B	27	11.76	58	38	78.99
	C	0.999	4.83	1.1966	0.99	1.21
	x^2	45.08	54.71	33.41	33.17	51.53
	$x^2_{critical}$	148.78	148.78	148.78	138.81	138.91
	P-value	1	0.999	1	1	0.999
Effort-base Model	B	31.46	10	59	38	74
	λ_{VU}	0.0009	0.001	0.0015	0.0063	0.0119
	x^2	19.14	23.53	14.22	21.16	26.75
	$x^2_{critical}$	61.66	61.66	61.66	103	103
	P-value	0.999	0.993	0.999	1	0.999

Table 4.5: Apache and IIS's Severity Chi-square Analysis of Goodness of Fit

Table 4.5 shows *the chi-square* analysis of goodness of fit for the Apache and IIS by severity level. Using regression analysis, we obtained parameter values from Figures 4.14, 4.15, 4.16 and 4.17. As was done previously, for *chi-square goodness of fit* test, we chose an alpha level of 5%. The *chi-square* and parameter values for the Time-based model and the Effort-based models are also shown in Table 4.5. This *chi-square* test shows that the fit for the three severity categories is significant, and the *chi-square* test shows that the vulnerabilities classified by severity data sets fit the model.

Figures 4.18 and 4.19 illustrate how severity level correlates with cause classification. It is noticeable that the input validation error constituted the majority among high severity vulnerabilities for both Apache and IIS. In Apache, a relatively smaller fraction of exceptional condition errors are of high severity. In IIS as well, the exceptional condition errors tend to be from among the vulnerabilities with low severity. For IIS, most configuration errors are medium severity.

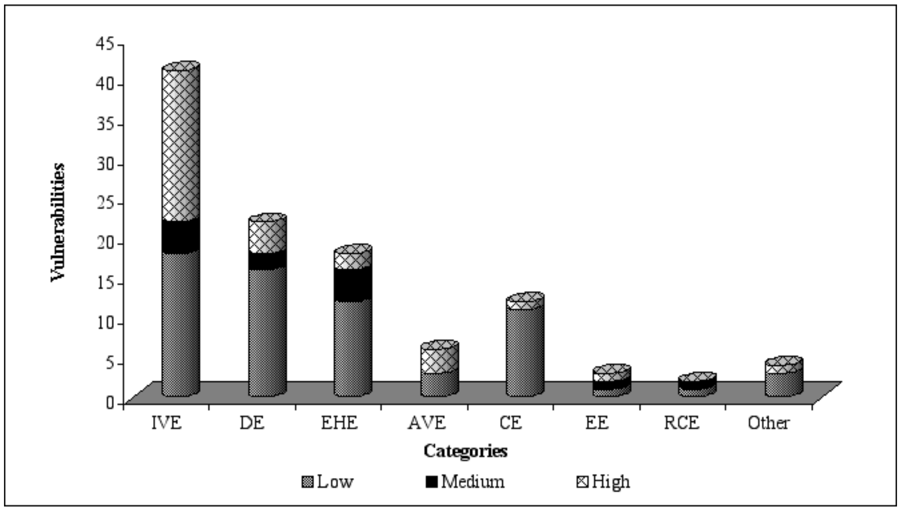

Figure 4.18: Apache Vulnerability Category by Severity

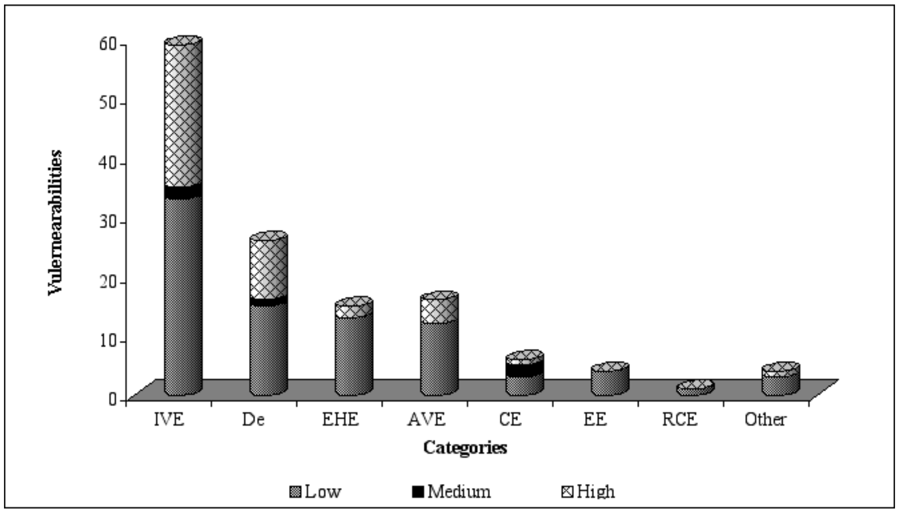

Figure 4.19: IIS Vulnerability Category by Severity

4.5 Discussion for HTTP Server Vulnerabilities and Models

When the total number of vulnerabilities is examined, both the Time-based and Effort-based models fit the data sets well, even when the vulnerabilities are categorized by type or severity. This suggests that the models can be used to estimate the number of vulnerabilities expected to be discovered in a given period, and which types and severity level are likely to dominate.

The results of model fitting for the vulnerabilities classified by type are shown in Table 4.4. The fitting was done for the most common types of vulnerabilities, input validation error, design error, access validation error and exceptional condition handling error for which the available data is statistically significant. It would be difficult to use these models to estimate the types of vulnerabilities that occur less frequently because the data may not be sufficiently statistically significant to make meaningful projections.

The results of model fitting for the vulnerabilities classified by severity are shown in Table 4.5. In all cases, there is enough data for the high and low severity vulnerabilities, and the fit is quite good. The results suggest that these two models can be used to project the expected number of high severity vulnerabilities.

The Effort-based model requires the use of market share data, which may be difficult to obtain. The Time-based model does not require this data; it can therefore be a feasible alternative when market share data is unavailable. Further research needs to be done to evaluate the predictive capabilities of the two models.

Analysis of the vulnerabilities classified by severity using the National Vulnerability Database standards for severity classification shows that a large fraction of the vulnerabilities found initially are high risk. However, subsequently a larger fraction of low severity vulnerabilities are encountered within the first and second years. Later, there is again a slight rise in the fraction of high severity vulnerabilities found. This variability was observed for both servers. Further research is needed to identify the cause of this variability.

After comparing the vulnerabilities trends of the web servers discussed, it is expected that fewer vulnerabilities will be discovered in IIS in the future. This may lead to the conclusion that IIS is more secure than Apache in this respect. However, this is in reality simply due to the fact that IIS has reached saturation phase, even though more IIS vulnerabilities have been found in the past. Other factors such as patch release, number of remaining vulnerabilities, economic aspects etc., also need to be considered when choosing a web server.

Chapter 5

Analysis of Vulnerabilities in Web Browsers

Browsers now represent the single most important Internet software. They serve as the client's platform for several security-critical applications such as Internet banking, e-commerce and on-line trading. There has been growing concern about potential insecurity in web browsers due to vulnerabilities. While the vulnerabilities and exploits of Microsoft IE (Internet Explorer) have been frequently discussed [27], its alternatives, such as Firefox, Opera, Safari and so on, are also not immune to serious vulnerability issues [38]. The existing studies of security vulnerabilities have been qualitative, focused on detection and prevention of vulnerabilities in web browsers. A number of security problems relating to the browsers are now being examined, such as spy-ware [36], phishing [17, 23, 27], web page filtering [22, 33], malicious pop-up windows [11], and e-commercial fraud [21, 30]. Many of these problems occur due to the presence of vulnerabilities in the browser software. Secure Science Corp. [40] reports a single phishing group collecting access information for 13,677 accounts in a single day by installing a malicious code through exploiting an unpatched vulnerability. Nimda, which used the buffer overflow vulnerability, affects all Windows versions of Microsoft Internet Explorer [58]. The exploitation techniques and tools utilized are no longer the exclusive possession of experts, since many such methods are now widely available and can be relatively easy to use.

While the early web server provided information using only static HTML pages, web servers now provide a dynamic and interactive service between the server and client using database queries, executable scripts, etc. The web server is increasingly adopting new features such as serving streaming media. Both the HTTP server and web browser have thus emerged as among the most critical components of the Internet.

The first public version of IE was released in August 1995. Firefox and Mozilla are based on Netscape Navigator, announced in October 1994, which emerged as the popular client web browser during the 1990s. IE currently has about 85% of the overall market share. However the popularity of Firefox has recently increased due to problems relating to IE vulnerabilities.

For the Internet service provider, servers for web (HTTP), ftp, mail, streaming media, etc., are the primary gates for the clients for whom a web browser is the main interface which connects them to the Internet. However, both servers and web browsers have numerous security holes. Web browsers provide a variety of essential and entertainment functions, such as e-commerce, e-mail, flash games, movies, etc. Some of these functions provide attackers or malicious users opportunities to exploit security holes since these processes require downloading, uploading and executing files. Browser vulnerabilities represent one of main sources of the spread of viruses or worms. However, the convenience of the dynamic technical functionalities offered by web browsers makes them indispensable.

In this section we examine the vulnerability discovery rates for the three main web browsers and explore the applicability of a vulnerability discovery model to the aggregate vulnerability data as well as data partitioned by causes and severity [56].

The next section introduces the vulnerability discovery model used and the significant factors that affect software vulnerability rates. We then consider the aggregate vulnerabilities in the three web browsers and examine how well the models fit the available data.

Web Browsers	**IE**	**Firefox**	**Mozilla**	**Safari**	**Other**
Market Share	85.18%	9.62%	0.37%	3.06%	1.77%
Vulnerabilities	265	94	38	24	N/A
Release Date	Aug 1995	Sep 2002	Dec 1998	Jan 2003	N/A
Latest Version	6.0 (sp2)	1.5.0.1	1.7.12	2.0.3	N/A

Table 5.1: Web Browsers Market Share and Vulnerabilities Found

The data sets are then partitioned into categories based on how such vulnerabilities arise, and consider the applicability of the models to individual categories is considered. Next, the vulnerabilities are divided according the severity of impact and the fit provided by the Time-base model is again examined.

5.1 Web Browser Market Share

Table 5.1 presents data obtained from NVD [15] and Net Applications [39], showing the current web browser market share and total number of vulnerabilities found to date. Other data that collect the usage share of web browsers show similar results, even though there are minor differences between monitoring web sites because they depend on the number of page hits, and the usage range of each web browser shows similar boundary such as IE usage between 87% and 83% and Firefox between 12% and 8%. As we can see from the table 5.1, for web browsers with a lower percentage of the market share, such as Mozilla and Safari, the total number of vulnerabilities found is low. This does not mean that these web browsers are more secure, but merely that only a limited effort has gone into detecting their vulnerabilities. Table 5.1 also shows that vulnerability discovery rates are more related to number of users or market share than period of usage. Although Mozilla was released earlier than Firefox and both source code sizes are similar, Firefox has a greater number of vulnerabilities than Mozilla.

5.2 Aggregate Vulnerabilities in Web Browsers

In this section we use the data for the three major web browsers, IE, Firefox and Mozilla, and determine whether vulnerability discovery trends are described by the Time-based and Effort-based models. IE controls about 85% of the Internet browser market. This high market share has made it an attractive target for exploration and exploitation by malicious users. The problem is exacerbated by the integration of IE into Windows, unlike Firefox or Mozilla. IE integration provides several benefits, such as faster start-up and easier interface with other components of Windows. However security analysts and experts consider the integration of IE to be a security disadvantage since IE connects with a variety of Windows core components. Another weakness of IE is the use of non-standard features, which do not follow the W3C standard. For example, ActiveX, which supports interfaces to provide a variety of functions and is offered as an add-in only for IE, can be used for executing arbitrary code. Even though IE is known for its many security flaws, numerous Internet users still prefer to use IE because many web sites are optimized for IE; moreover, Windows software is marketed with IE pre-installed.

Although Firefox was released in September 2002, it did not gain significant recognition until 2004; its popularity has increased because of its perceived better security, intuitive design and multi-tap features. Currently Firefox is more common in school or public computers and is expanding its market share. However, its popularity has led to a rising number of newly discovered vulnerabilities.

Figure 5.1 shows the cumulative vulnerabilities by month and the fitted the Time-based model for IE. Figure 5.2 shows the fitted Effort-based model by IE's *million system months.* The bold black lines indicate the fitted models, while the other thin lines show the cumulative number of vulnerabilities by month and IE's *million system months.* The Time-based and Effort-based models fit the data for IE very well. At the beginning, the slope of the curve

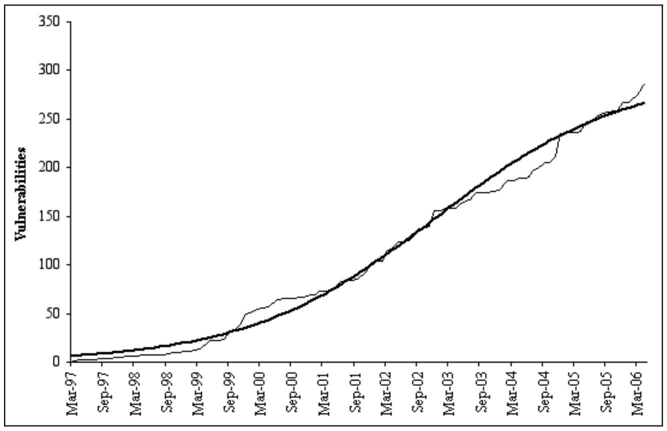

Figure 5.1: Time-based Model Fitting for Aggregate Vulnerabilities in IE

for IE rose gently until 2000, after which the slope has generally remained steady.

From the point of the three phases of the vulnerability discovery process, IE does not appear to have yet entered the saturation phase; also, the Effort-based model show similar result to the Time-based model. Rather, IE currently still appears to be in the linear phase, since the number of vulnerabilities is growing linearly in spite of IE's having been on the market for several years. This may be because of its larger market share and possibly because it may have a higher number of potential vulnerabilities. This suggests that vulnerability discovery for IE may continue at a significant pace in the near future. It is expected that the next release of IE will have more security focus.

Firefox is currently the second most popular web browser. While there is still a considerable market share gap between IE and Firefox, this gap is shrinking. Although Firefox is only a four-years-old web browser and its market share includes about one-eighth of the market, the fitted the Time-based and Effort-based models as shown in Figure 5.3 and 5.4, suggests that Firefox is still in the linear phase. Consequently, we can expect that while more vulnerabilities will be found in the near future, the saturation phase is not likely to be reached soon.

Mozilla was first released at the end of 1998. However, since Mozilla never became

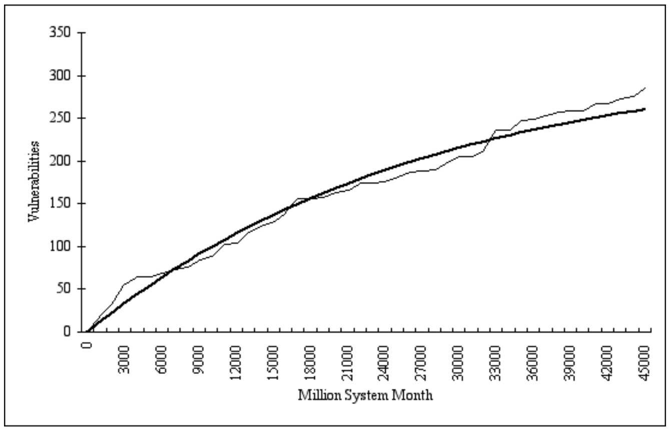

Figure 5.2: Effort-based Model Fitting for Aggregate Vulnerabilities in IE

very popular among Internet users, very few vulnerabilities have been found in Mozilla even though it was developed long before Firefox. Only eleven vulnerabilities were found through June 2004, as shown in Figure 5.5. Comparing Figure 5.3 and 5.5, we observe that the discovery rate of Firefox and Mozilla vulnerabilities suddenly increased in the later part of 2004. A large number of vulnerabilities were first found in Firefox, followed by a similar rise in the discovery of Mozilla vulnerabilities. This is likely to be due to the fact that significant parts of code are shared between Firefox and Mozilla, demonstrating that market share can be a more important contributing factor than software age.

Figure 5.7 shows that significant parts of code are shared between Firefox and Mozilla. Over 50% of Mozilla's vulnerabilities are vulnerabilities shared with Firefox, in fact most of the vulnerabilities found in Mozilla after October 2004 were actually found in the shared code. The popularity of Firefox increased markedly at that time. Figure 5.5 suggests superimposition to two s-shapes, one due to vulnerabilities found in Mozilla itself and the second due to vulnerabilities found in Firefox. Most of the second period vulnerabilities are the shared vulnerabilities, which found a vulnerability to be the same vulnerability in two or more softwares (usually one software is the offspring of the other software). This reason also make the Effort-based model is difficult to fit Mozilla's vulnerability data. Figure 5.6

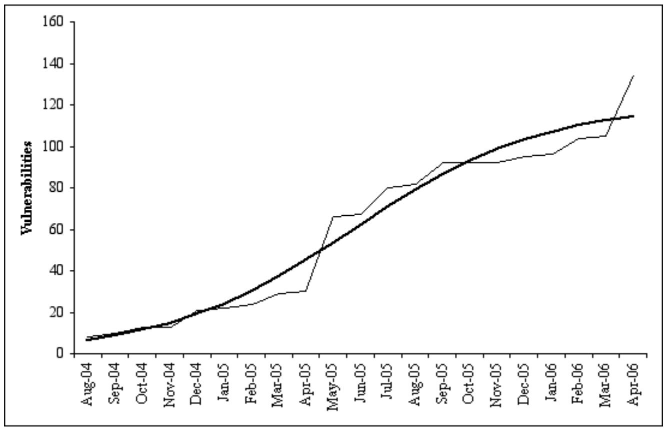

Figure 5.3: Time-based Model Fitting for Aggregate Vulnerabilities in Firefox

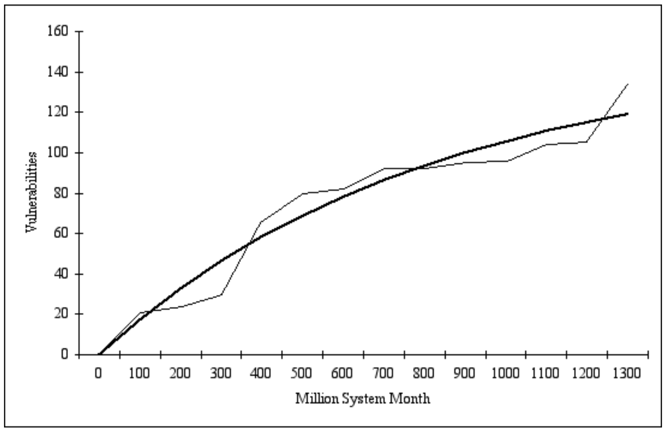

Figure 5.4: Effort-based Model Fitting for Aggregate Vulnerabilities in Firefox

shows the number of vulnerabilities by *million system months*. Further research is needed to develop methods that can accurately model such superimposition.

Table 5.2 shows parameter values obtained by fitting the models used in Figure 5.1, 5.2, 5.3, 5.4, and 5.5 with the exception of Mozilla's Effort-based model since it is difficult to apply for Mozilla's vulnerability data by *million system months*. For *chi-square* goodness of fit test, we chose an alpha level 5%. Table 5.2 gives each browser's *chi-square* values, R^2 values and parameter values for the Time-based and Effort-based model. When the *P-value* value is close to 1, the model data fit is significant; moreover, R^2 close to 1 indicates strong

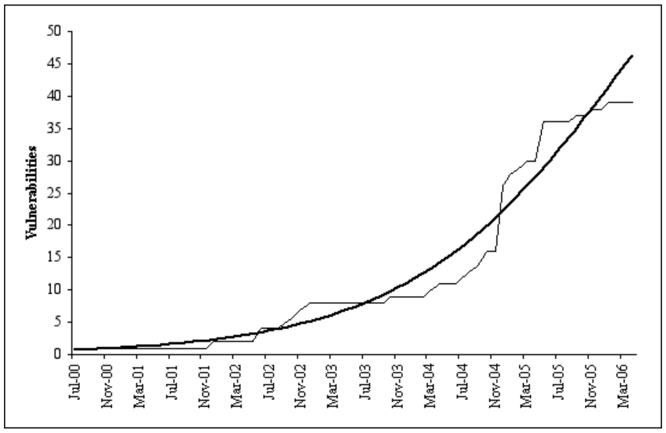

Figure 5.5: Time-based Model Fitting for Aggregate Vulnerabilities in Mozilla

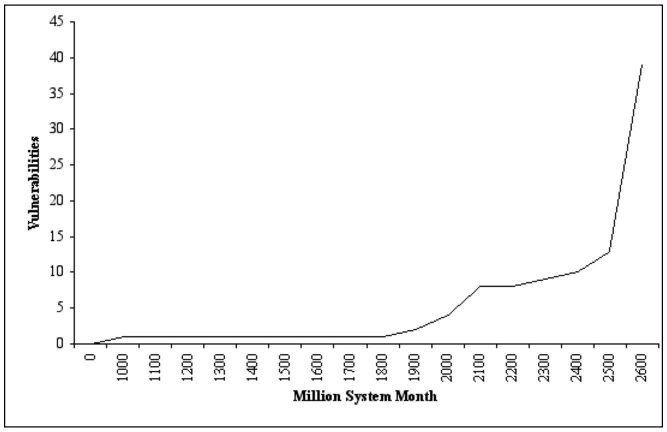

Figure 5.6: The Number of Vulnerabilities by Mozilla's Million System Months

correlation between the model and actual data. The table shows that the *chi-square* values are less than the critical value with the exception of IE's Time-based model. *P-values* for Firefox and Mozilla are in the acceptable range since p-value is greater than 0.05-that is, with an alpha level of 5%. The *chi-square* value and *P-value* for IE's Time-based model are not significant with respect to the level chosen; however, the R^2 value is very close to 1, indicating a strong correlation between the model and actual vulnerability data.

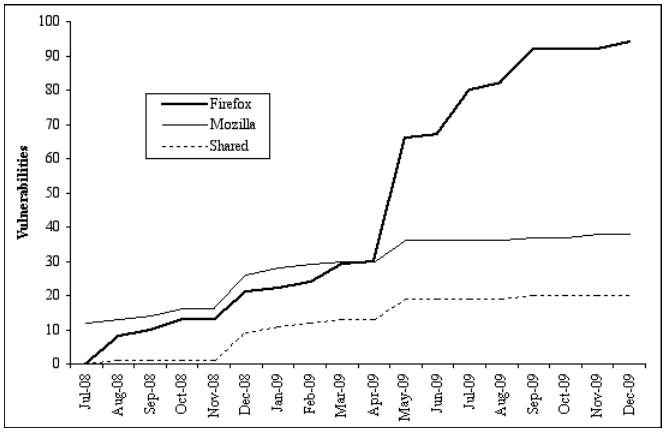

Figure 5.7: Shared Vulnerabilities Between Firefox and Mozilla

Model	Parameter	IE	Firefox	Mozilla
Time-base Model	A	0.00018	0.0024	0.0007
	B	295	119.8	95.9
	C	0.1643	0.1908	1.5310
	R^2	0.9887	0.9570	0.9572
	x^2	174.1	20.8	32.078
	$x^2_{critical}$	135.5	32.6	90.531
	P-value	7.43E-05	0.4055	0.9999
Effort-base Model	B	320.84	148.08	N/A
	λ_{VU}	0.000037	0.00124	N/A
	R^2	0.977	0.946	N/A
	x^2	59.18	19.82	N/A
	$x^2_{critical}$	61.66	22.36	N/A
	P-value	0.062	0.166	N/A

Table 5.2: x^2 Goodness of Fit Test Results for Aggregate Number of Vulnerabilities

5.3 Individual Vulnerability Categories

In this and the following sections, we apply the Time-based and Effort-based model to cause classification scheme for web browsers' vulnerabilities.

Like web servers' categorized vulnerabilities, these eight classes are not completely mutually exclusive. We apply the same scheme as server vulnerability categories. Table 5.3 shows how vulnerabilities are distributed among categories for both the data sets studied. The summation of all categories for a single software system may add up to more than the total number of vulnerabilities (also, the added up percentages may exceed 100%) because several vulnerabilities in each web browser can belong to more than one category. We define these vulnerabilities as overlap vulnerabilities. IE has twenty-five overlap vulnerabilities, Firefox has twelve and Mozilla has three. In Table 5.3, three web browsers show the same pattern - i.e., that the number of design errors is much higher than other types of vulnerabilities, followed by input validation error, etc.

Figure 5.8 compares vulnerability distributions in three web browsers. More than 60% of found vulnerabilities are related to design or input validation errors. When comparing web browsers to web servers (Apache and IIS) and operating systems (Windows 2000 and XP), we find a comparable pattern. Usually web servers and operating systems have a greater number of vulnerabilities in input validation error than in design error. Apart from these two categories, other classes show a similar priority order (exceptional condition error, access validation error, configuration error and other classified errors).

Figure 5.9, 5.10, 5.11, 5.12 and 5.13 present the Time-based and Effort-based models fitting of each web browser's vulnerabilities by category. Here, we consider only the two major categories, design errors and input validation errors, since other categories have too small a number of vulnerabilities to fit to the models. In these five figures, the bold lines indicate the fitted the models for each category, while the dotted lines and thin lines indicate

Category	IE	Firefox	Mozilla
IV	78 (27.3%)	40 (29.9%)	16 (41%)
DE	11 (38.8%)	68 (50.7%)	17 (43.6%)
ECE	41 (14.3%)	8 (6%)	3 (7.7%)
AVE	38 (13.3%)	14 (10.4%)	3 (7.7%)
CE	11 (3.8%)	2 (1.5%)	1 (2.6%)
EE	6 (2.1%)	0 (0%)	0 (0%)
RCE	3 (1%)	0 (0%)	1 (2.6%)
Other	23 (8%)	14 (10.4%)	1 (2.6%)
Total	286	134	39

Table 5.3: Web Browsers' Vulnerabilities Classified by Categories

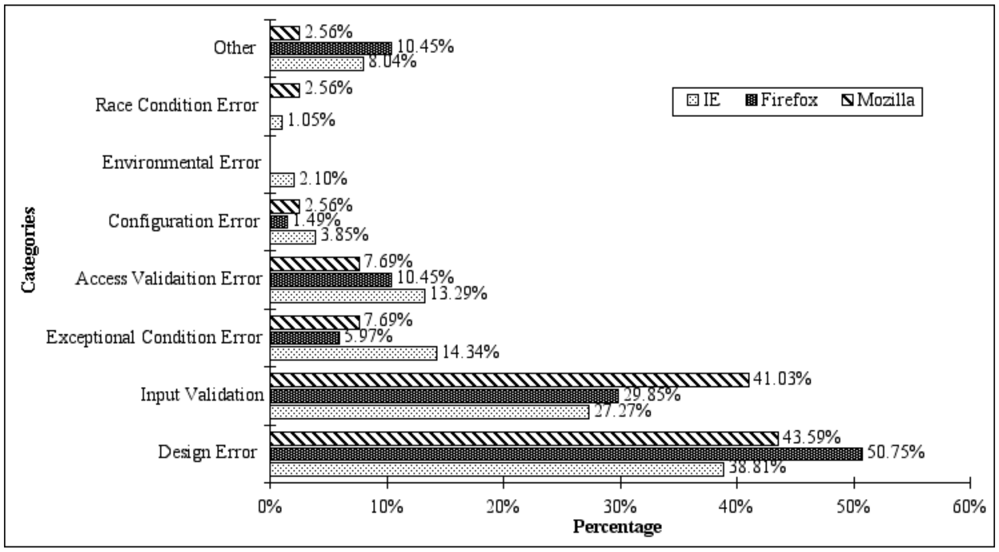

Figure 5.8: Web Browsers' Vulnerabilities by Category

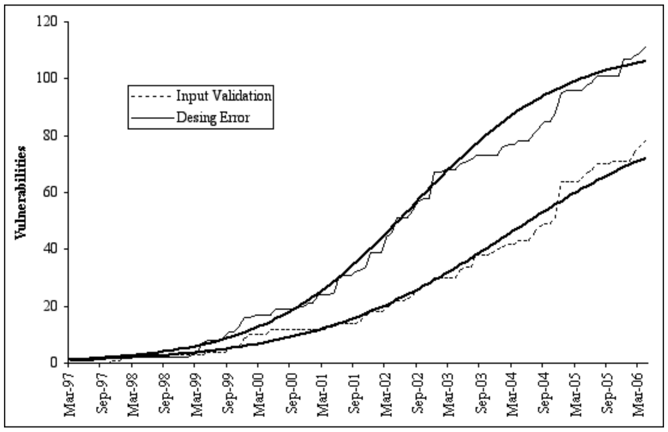

Figure 5.9: Time-based Model Fitting for IE Vulnerabilities by Category

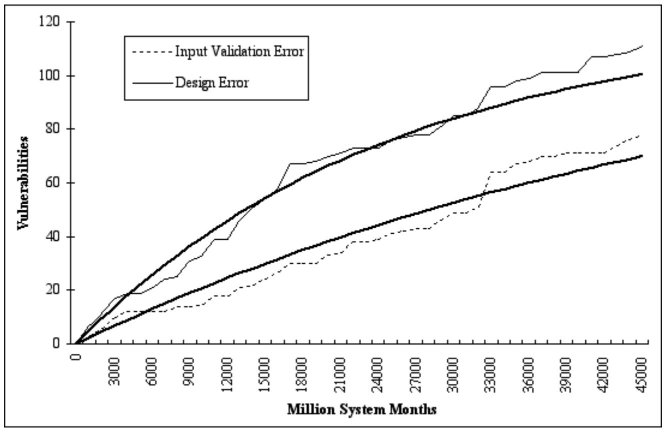

Figure 5.10: Effort-based Model Fitting for IE Vulnerabilities by Category

cumulative vulnerability data for each category.

Figure 5.9 shows the Time-based model fitting for categorized IE vulnerabilities. From the beginning to the present, the Time-based model and the cumulative data demonstrate that design errors have been found more frequently than input validation errors, and that the gap between design error and input validation error is widening. Figure 5.10 plots IE's vulnerability by *million system months* and the Effort-based model fitting. This plot also shows the same result as Figure 5.9. Both figures denote more vulnerabilities will be found in design errors and input validation errors.

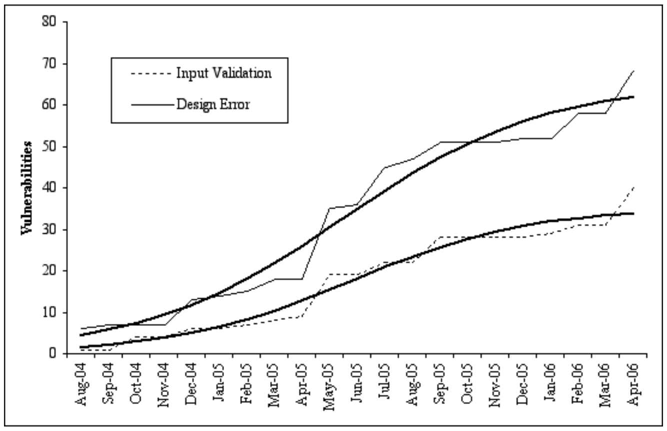

Figure 5.11: Time-based Model Fitting for Firefox Vulnerabilities by Category

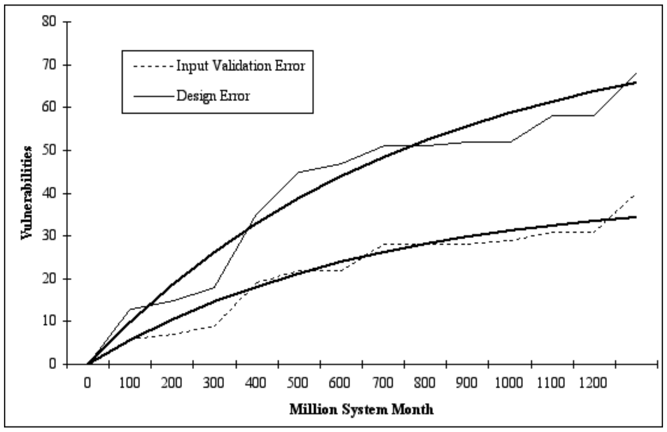

Figure 5.12: Effort-based Model Fitting for Firefox Vulnerabilities by Category

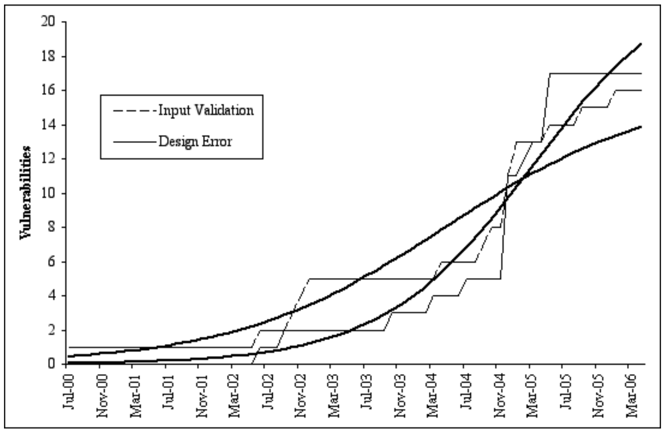

Figure 5.13: Time-based Model Fitting for Mozilla Vulnerabilities by Category

Figure 5.11 and 5.12 show the Time-based and Effort-based models' fitting for categorized Firefox vulnerabilities. Categorized Firefox vulnerabilities demonstrate a pattern similar to that of categorized IE vulnerabilities. Design errors have been found more frequently than input validation errors, and the gap between design error and input validation error is widening.

Mozilla fitting for the Time-based model in Figure 5.13 shows a different pattern from previous two web browsers. As mentioned in a previous section, it shows two s-shapes (two Time-based models), one due to vulnerabilities found in Mozilla itself (from July 2000 to March 2003), and the second due to vulnerabilities found in Firefox (from March 2003 to April 2006). This figure shows two s-shaped patterns that are exactly matched, since from the initial period, input validation errors were published more frequently than design errors, first narrowing widening the gap. Figure 5.14 shows the relationship between cumulative vulnerabilities and Mozilla's *million system months* with the exception of the Effort-based model.

Table 5.4 shows the *chi-square* goodness of fit tests for the IE, Firefox and Mozilla models by category. For each category, the *chi-square* (x^2) value is less than *chi-square critical* ($x^2_{critical}$), and the *P-values* and R^2 values are close to 1. Thus, the fit for the two

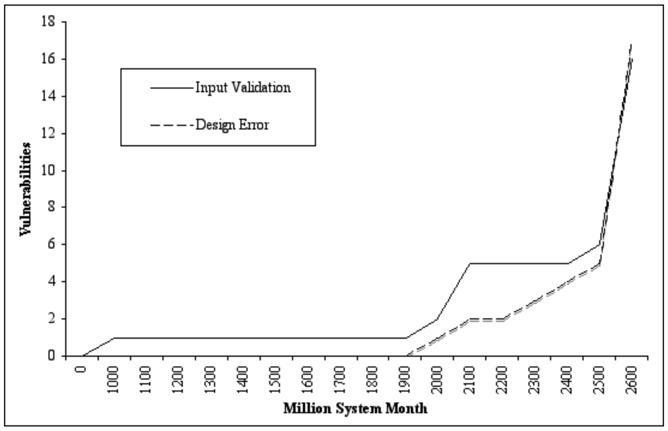

Figure 5.14: The Categorized Number of Vulnerabilities by Mozilla's Million System Months

categories is significant for both the Time-based and Effort-based models. It is interesting to note that the fit for the aggregate IE vulnerabilities considered in the previous section was not significant with respect to the significance level chosen.

We adapted the Time-based and Effort-based models to two major vulnerability categories to determine whether there are observable patterns at the level of individual classes. Since we noted a similar pattern for the uncategorized vulnerabilities, a possible fit was examined. These individual classes reflect each web browsers' own total number of vulnerabilities.

5.4 Vulnerability Severity Levels

Earlier, in the chapter 4.4, we explained the definition of vulnerability severity and each severity level were already explained. In this section, web browsers' vulnerability severity levels are fitted for the Time-based and Effort-based models. Figure 5.15, 5.16 and 5.17plot the percentages of the cumulative number of vulnerabilities for each severity class for IE, Firefox and Mozilla respectively. It should be noted that the early part of the curves represent very few vulnerabilities and thus are not significant; for example, the plots for Mozilla up to July 2002 represent only five vulnerabilities. A comparison of the IE plot in Figure

		IE		Firefox		Mozilla	
Model	Parameter	IVE	DE	IVE	DE	IVE	DE
Time-base Model	A	0.00059	0.00062	0.0089	0.0042	0.0048	0.0045
	B	89.7	110.9	35	65.1	16	23.4
	C	0.984	0.895	0.849	0.278	2.385	20.39
	R^2	0.98	0.99	0.962	0.96	0.92	0.95
	x^2	43.9	47.2	6.5	9.1	22.3	23.8
	$x^2_{critical}$	135.4	135.4	32.6	32.6	90.5	90.5
	P-value	0.999	0.999	0.999	0.999	0.999	0.999
Effort-base Model	B	125	120	40	80	N/A	N/A
	λ_{VU}	0.000018	0.00004	0.0015	0.0013	N/A	N/A
	R^2	0.96	0.98	0.95	0.96	N/A	N/A
	x^2	34.27	21.73	6.66	8.61	N/A	N/A
	$x^2_{critical}$	61.66	61.66	22.36	22.36	N/A	N/A
	P-value	0.854	0.998	0.959	0.819	N/A	N/A

Table 5.4: x^2 Goodness of Fit Test Results for Total Number of Vulnerabilities

5.15 with similar plots of vulnerabilities for the Apache and IIS servers suggest that during the middle of 2001 to middle of 2004, a proportionately larger number of high severity vulnerabilities were found. From the middle of 2004, the percentage of high and low severity vulnerabilities has remained relatively unchanged. This would suggest a shift in vulnerability detection priorities. The higher emphasis on higher severity vulnerabilities during 2001-2004 may have been a consequence of higher densities of undiscovered vulnerabilities during that time; thus, the finders may have found it more rewarding to seek higher severity vulnerabilities. Since the middle of 2004, the proportions of high and low severity vulnerabilities found appear to have remained relatively unchanged. Figure 5.16 shows relative flat curves for Firefox, which may have been because the data is from the period starting from August 2004.

We apply the Time-based and Effort-based models to the three web browsers. Figure 5.18, 5.19, 5.20, 5.21 and 5.22 show the results of fitting these two models to the three severity classes. The web browsers' severity pattern is the same as that of the web servers. Low severity vulnerabilities are easily found in all time periods, while high severity vulnerabilities

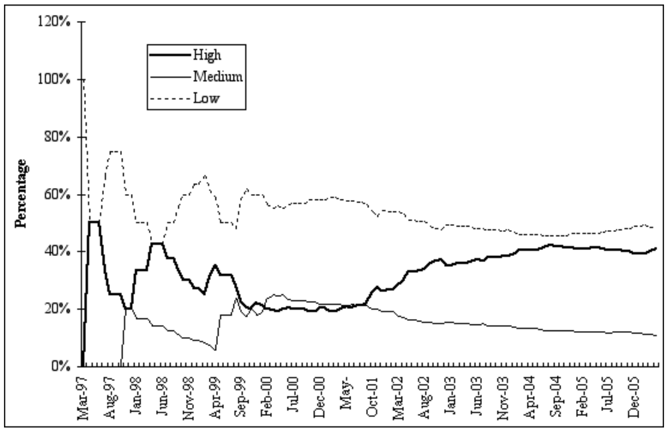

Figure 5.15: IE Severity Variation

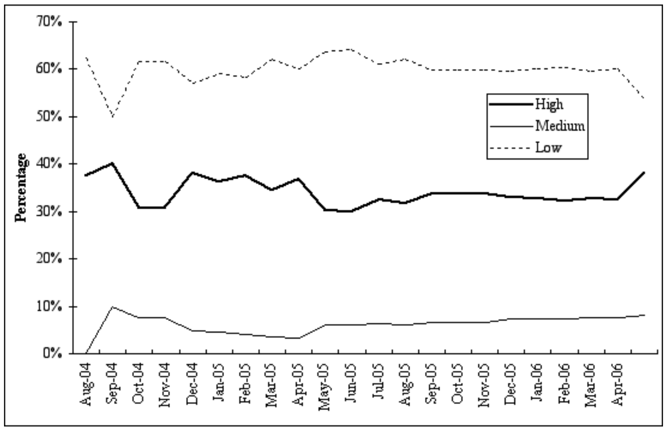

Figure 5.16: Firefox Severity Variation

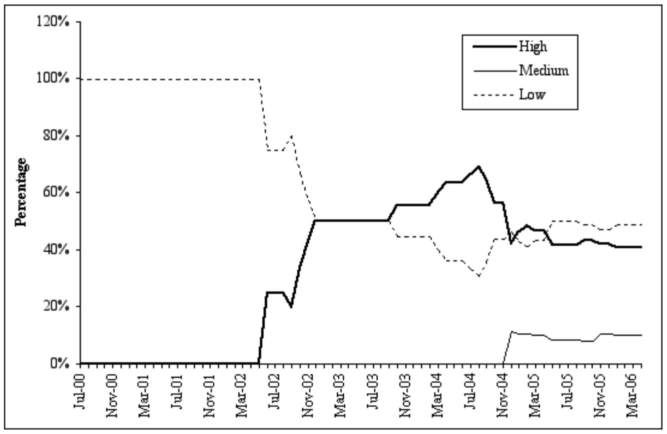

Figure 5.17: Mozilla Severity Variation

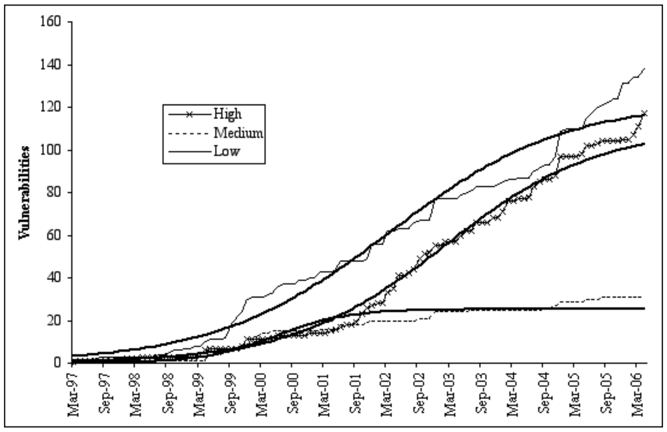

Figure 5.18: Time-based Model Fitting for IE Vulnerabilities by Severity

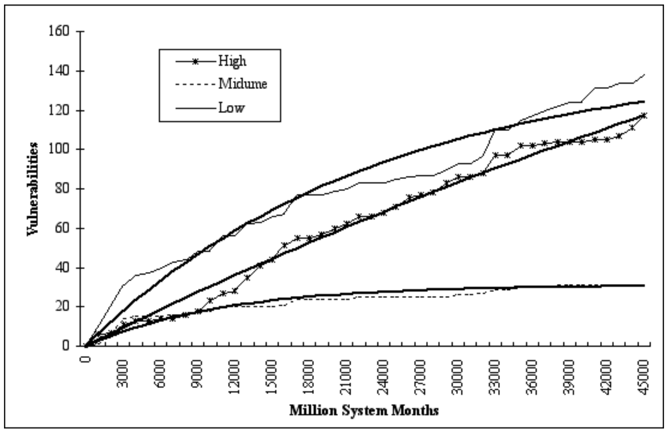

Figure 5.19: Effort-based Model Fitting for IE Vulnerabilities by Severity

rank second and medium severity vulnerabilities are rarely found. This shows that the size of the low severity level pool is largest, followed by the high severity level pool and finally the medium severity level pool. With the exception of the IE's low severity vulnerabilities, the fit is significant for all cases. Even for the low severity IE vulnerabilities, the R^2 value is quite high.

Figure 5.23, 5.24 and 5.25 show each web browser's vulnerability category by severity. Even though the design errors cover a higher proportion than input validation, many of high severity vulnerabilities are usually found in input validation errors. The same result is shown

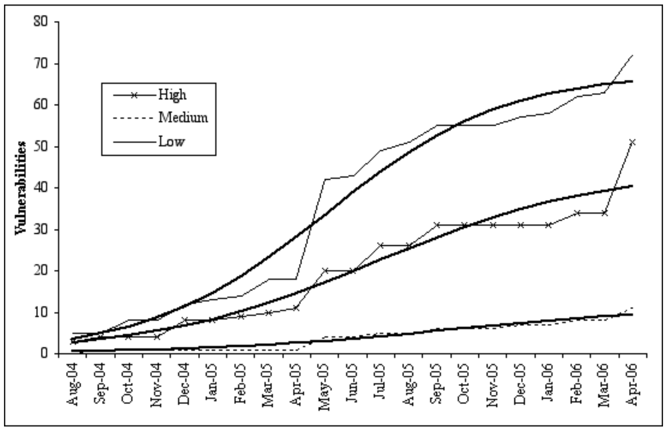

Figure 5.20: Time-based Model Fitting for Firefox Vulnerabilities by Severity

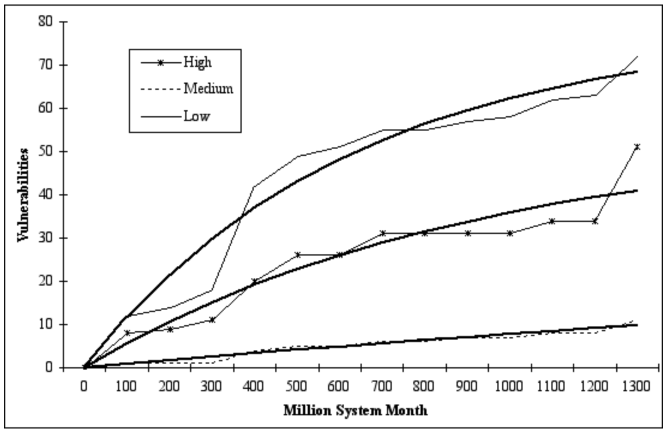

Figure 5.21: Effort-based Model Fitting for Firefox Vulnerabilities by Severity

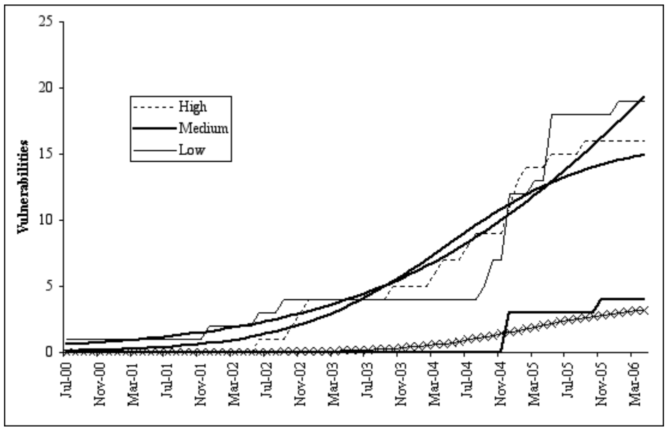

Figure 5.22: Time-based Model Fitting for Mozilla Vulnerabilities by Severity

		IE			Firefox			Mozilla		
Model	Para.	High	Med	Low	High	Med	Low	High	Med	Low
Time-base Model	A	0.0006	0.006	0.0005	.0057	0.0206	0.005	0.007	0.04	0.001
	B	110	25.1	122.2	44.1	11.3	67.6	16	3.6	40.9
	C	1.255	12.47	0.325	0.447	2.56	0.36	10	1000	1.77
	R^2	0.99	0.92	0.96	0.92	0.93	0.96	0.97	0.87	0.91
	x^2	40.23	46.38	107.42	8.83	4.02	12.1	21.65	19.84	29.67
	$x^2_{critical}$	135.4	135.4	135.4	32.6	32.6	32.6	90.5	90.5	90.5
	P-value	1	0.99	0.12	0.98	0.99	0.91	1	1	0.99
Effort-base Model	B	359.6	31	145	53.68	41.62	78	N/A	N/A	N/A
	λ_{VU}	.00001	.0001	.00004	.0011	0.0002	0.002	N/A	N/A	N/A
	R^2	0.99	0.93	0.95	0.91	0.94	0.95	N/A	N/A	N/A
	x^2	22.08	17.24	55.2	7.4	3.43	14.14	N/A	N/A	N/A
	$x^2_{critical}$	61.66	61.66	61.66	22.36	22.36	22.36	N/A	N/A	N/A
	P-value	0.997	0.999	0.119	0.818	0.999	0.619	N/A	N/A	N/A

Table 5.5: x^2 Goodness of Fit Test Results for Total Number of Vulnerabilities

in the web servers' vulnerability category by severity: most of high severity vulnerabilities are found in input validation errors.

5.5 Discussion for Web Browsers' Vulnerability and Models

The introduction of browsers has created a powerful new medium for conducting business, commercial and personal activities. Potential discovery and exploitation of vulnerabilities has become a subject of great concern, since web browsers' vulnerabilities are used as a medium of spreading viruses and worms. The vulnerability discovery rate trends provide a quantitative perspective of the problem and can be use to plan the effort needed to implement effective risk containment strategies. As an example, quantitative projections can be used to allocate resources needed for fast patch development.

We examined the data sets to determine whether the discovery process trends to follow specific patterns and whether these patterns can be modeled. The results show that when the aggregate number of vulnerabilities is examined, the Time-based model and the Effort-based model fit the data sets well, as shown in Table 5.2. The model was found to fit even

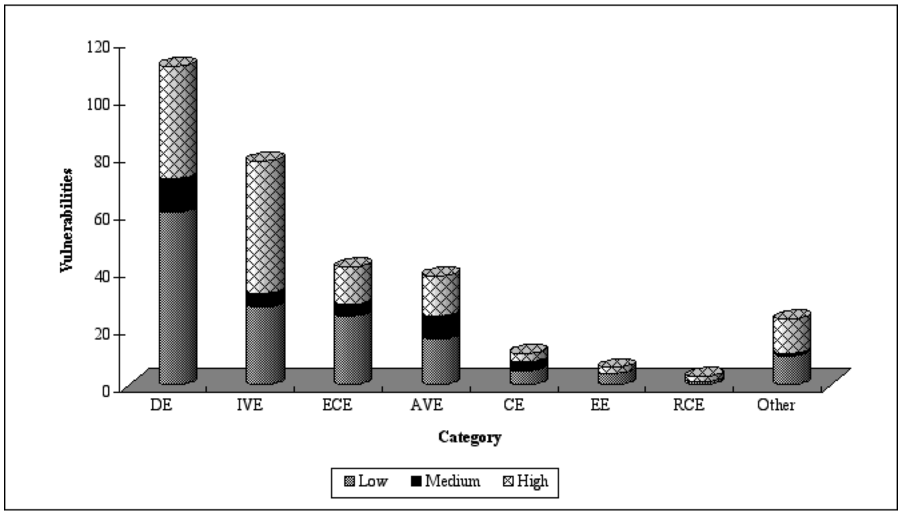

Figure 5.23: IE's Vulnerability Category by Severity

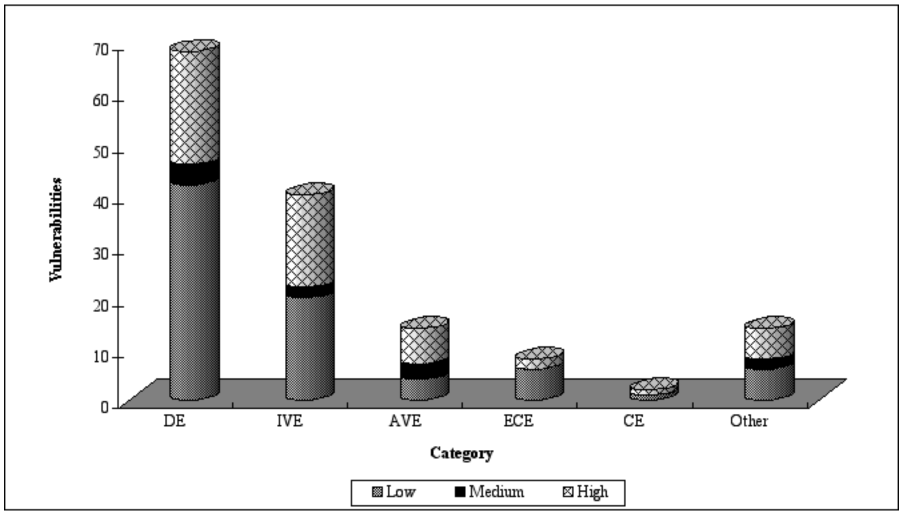

Figure 5.24: Firefox Vulnerability Category by Severity

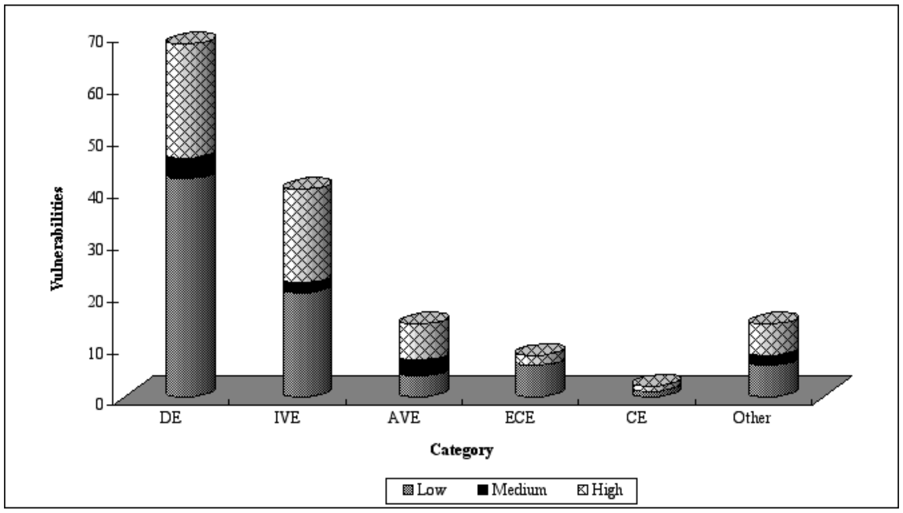

Figure 5.25: Mozilla Vulnerability Category by Severity

when vulnerabilities were partitioned by cause classification or severity levels (Table 5.3 and 5.5). This suggests that the models can potentially be used to estimate the number of vulnerabilities expected to be discovered during a given future period, and what category and severity level distributions are likely.

The fitting was done for the classes of vulnerabilities for which the available data is statistically significant. It would be difficult to use such models for types of vulnerabilities that occur less frequently because the data may not be sufficiently statistically significant to make meaningful projections.

We note that there is sufficient data for high and low severity vulnerabilities, and the fit is quite good. This suggests that the model can be used to project the expected number of high severity vulnerabilities, which may be of much greater interest than others.

The Time-based model used here does not require the use of market share data. The Effort-based model can potentially generate more stable projections because it can remove fluctuations due to variability in the discovery effort with time; however, it requires collection of reliable effort data. Further research is needed to evaluate the applicability of the

Time-based model. Further research also needs to be undertaken to evaluate the predictive capabilities of these two models.

Examining the current vulnerability discovery trends for the three web browsers, all three appear to be in the linear phase. Hence, it can be expected that more vulnerabilities will be discovered in all three.

When comparing browser security, we need to keep in mind that the vulnerability discovery rate in the near future may be more important than vulnerabilities already discovered in the past. Other factors to consider include severity levels and quick availability of patch releases. Currently, certain experts [48] regard IE to be less secure; some of them point to the integration of IE into Windows and Active X. Secunia [52] reports that IE has more unpatched vulnerabilities than Firefox. However, if Firefox's popularity continues to increase, it will attract more attempts to discover its vulnerabilities. The new version of IE7, currently in beta version, is intended to be more secure because of incorporation of additional security features.

Chapter 6

Discussion

The Time-based model and the Effort-based model are examined for HTTP servers and web browsers' vulnerability data. Both models fit the HTTP severs' and the web browsers' datasets very well. This shows that the software vulnerability trend forms a pattern even when the vulnerabilities are categorized by type, such as cause and severity. There are three phases by time period (learning, linear and saturation phase) and logarithm shape by increasing the number of users (million system months). A software that has a fixed quantity of market share follows these vulnerability-found trends. However, these two models is difficult to apply to a low number of vulnerabilities. Further research needs to be done for these low numbers of vulnerabilities.

As mentioned in chapter 2, the Effort-based model requires the usage of specific software. This data may be difficult to obtain or may be not pure. However, the usage of specific software data is not needed for the Time-based model. Fortunately, the Time-based and Effort-based models show similar results: when the Time-based model's *chi-square goodness of fit* test is significant, the Effort-based model's *chi-square goodness of fit* test is also significant. For example, in chapter 5.3, the Time-based model's *chi-square* value and *P-value* for IE and Firefox show less significance than others. Moreover, the Effort-based model's *chi-square* value and *P-value* for IE and Firefox show less significance than others. Conse-

Application	SLOC	Defects	D_{KD}	Vulnerabilities	V_{KD}	Ratio V_{KD}/D_{KD}
Apache	227,410 (Unix)	4148	18.27	96	0.423	0.0232
IIS	N/A	N/A	N/A	123	N/A	N/A
IE	N/A	N/A	N/A	286	N/A	N/A
Firefox	2,500,000	24,027	9.61	134	0.0536	0.00557
Mozilla	2,430,000	38,060	15.64	39	0.016	0.00103
Win 98	16,000,000	10,000	0.625	91	0.0057	0.0091
Win NT	18,000,000	10,000	0.556	230	0.0128	0.023

Table 6.1: Known Defect Density vs. Known Vulnerability Density

quently, the Time-based model can be an alternative method for examining vulnerability trends when the market share data is absent.

Static analysis has been used in software reliability engineering when some of the systems' attributes are estimated empirically even before testing begins. Similar static analysis can be carried out by utilizing metrics such as software size and estimated number of total defects. These methods can potentially be used to estimate *Defect density* (D_{KD}) and *Vulnerability density* (V_{KD}), which can then be used to estimate the total number of vulnerabilities of a comparable system. D_{KD} gives the defects per thousand lines of code and V_{KD} is the number of vulnerabilities per thousand lines of code. Table 6.1 shows some of the major attributes of the Apache server, Firefox, Mozilla and two other major operating systems for comparison. Unfortunately, some of the important metrics for the Microsoft IIS server and IE were not available to us at the time. For proprietary systems, such data can be hard to obtain outside of the developing organization. The code size for Apache, Firefox and Mozilla was determined using the SLOCCount tool [53] since the source code for these open source are available.

In Table 6.1, we observe that the vulnerability density values for the Windows systems are significantly less than for HTTP severs and web browsers and web browsers defects densities are much higher than others. This may be due to the fact that Windows software has large segments that do not play a role in accessibility, while severs are smaller and therefore vulnerabilities are more concentrated in the code: the relatively higher values of the vulnerability density for web servers and browsers are likely to occur because of the higher fraction of code involving access functionality. Also V_{KD}/D_{KD} ratio are within a narrow range. This assumption is supported by the fact that the defect density to vulnerability density ratio is higher in Windows NT 4.0, a server operating system and web browsers than in Windows 98, a client operating system.

IIS and IE are distributed for the Windows platform only, and all of its vulnerabilities are related to the parent platform. However, Apache, Firefox and Mozilla are used in most major operating systems, including several versions of both Windows and UNIX (Linux). In this study we did not distinguish Apache's, Firefox's and Mozilla's vulnerabilities by platform, as we assumed that it provides the same functionality regardless of operating system. However, we need more research for vulnerabilities in the different platform.

HTTP servers' and web browsers' vulnerability trends are compared in this study. In this study, two major web servers had reached or are reaching saturation phase; other than this, web browsers are in the linear phase. It is expected that many of vulnerabilities will be found in web browsers and fewer vulnerabilities will be discovered in HTTP servers. It is difficult to decide which application is more secure since there are many other factors, such as patch release, number of remaining vulnerabilities, economic aspects etc. However, this study provide one guide for deciding security level.

Chapter 7

Conclusions

This study examines the trends in vulnerability discovery of HTTP servers and web browsers and explores the applicability of quantitative models for the number of vulnerabilities. The study has also demonstrated that the vulnerabilities discovery process in HTTP servers and web browsers follows a pattern, which can be modeled. The Time-based and Effort-based models give us an insight into the vulnerabilities discovery process. The results show that the Time-based and Effort-based vulnerability discovery models generally track the available data well. Therefore, it is possible to make reasonable projections about the number of remaining vulnerabilities and vulnerabilities discovery rates.

We found that the fit is significant when aggregate vulnerabilities are divided into classes (for example, vulnerabilities arising due to design errors and input validation errors), provided there are sufficient vulnerabilities in a class. The fit was significant for both the Time-based model and Effort-based models. It was observed that a larger number of input validation error vulnerabilities are found in HTTP servers and that design error vulnerabilities cover a higher proportion of web browsers vulnerabilities. This also shows a larger number of input validation error vulnerabilities constitute a high severity level for both HTTP severs and web browsers. This suggests that more effort should be spent on testing in order to target vulnerabilities from this class, thereby minimizing the number of high risk vulnerabilities. The models can thus be used to project the classes of vulnerabilities that are more likely to be encountered, and consequently can be used to make testing more effective. The model also fits the data for high and low severity vulnerabilities. Hence, it is possible to project the high severity vulnerabilities that may be expected in the near future.

The results indicate that the models originally proposed and validated for operating systems are also applicable to web browsers. These models can be used to estimate vulnerabilities discovery rates, which can be integrated with risk assessment models in the future. A model recently proposed by Sahinoglu [46] needs such an assessment for estimating risk and cost of loss. Moreover, these models can be integrated into the development process to create more secure software systems [50].

Further work is needed to evaluate the prediction accuracy of the models so that users can measure how accurately these models can predict future vulnerabilities discovery rates. Further research is also required to evaluate the degree of confidence that can be attained when these methods are used to predict the types of vulnerabilities that are anticipated and their severity levels. Also vulnerabilities for the O.S platform is need to be distinguished.

REFERENCES

[1] Fumio Akiyama. An example of software system debugging. *IFIP Congress*, pages 353–379, 1971.

[2] Omar H. Alhazmi and Yashwant K. Malaiya. Modeling the vulnerability discovery process. *In Proceedings of Annual reliability and Maintainability Sysposium*, pages 129–138, January 2005.

[3] Omar H. Alhazmi and Yashwant K. Malaiya. Quantitative vulnerability assessment of system software. *In Proceedings of Annual Reliability and Maintainability Symposium*, pages 615–620, January 2005.

[4] Omar H. Alhazmi and Yashwant K. Malaiya. Prediction capability of vulnerability discovery process. *In Proceedings of 52nd Reliability and Maintainability Symposium*, January 2006.

[5] Omar H. Alhazmi, Yashwant K. Malaiya, and Indrajit Ray. Security vulnerabilities in software systems: A quantitative perspective. *Data and Applications Security XIX, 19th Annual IFIP WG 11.3 Working Conference on Data and Applications Security*, pages 281–294, August 2005.

[6] Ross Anderson. Security in open versus closed systems - the dance of boltzmann, coase and moore. *Conference on Open Source Software: Economics, Law and Policy*, pages 1–15, 2002.

[7] Net Applications. *http://marketshare.hitslink.com/*, 2006.

[8] Taimur Aslam and Eugene H. Spafford. A taxonomy of security faults. Technical report, Purdue University, 1996.

[9] Tuomas Aura, Matt Bishop, and Dean Sniegowski. Analyzing single-server network inhibition. *In Proceedings of the 13th IEEE Computer Security Foundations Workshop*, pages 108–117, July 2000.

[10] Matt Bishop. Vulnerabilities analysis. *In Proceedings of the Second International Symposium on Recent Advances in Intrusion Detection*, pages 125–136, September 1999.

[11] Andrew Brandt and Eric Dahl. New ad attacks. *PC World*, 23:20–22, February 2005.

[12] Sarah Brocklehurst, Bev Littlewood, Tomas Olovsson, and Erland Jonsson. On measurement of operational security. *In Proceedings of 9th Annual IEEE Conference on Computer Assurance*, pages 257–266, June 1994.

[13] Hilary K. Browne, William A. Arbaugh, John McHugh, and William L. Fithen. A trend analysis of exploitations. *In Proceedings of the 2001 IEEE Symposium on Security and Privacy*, pages 214–229, 2001.

[14] CERT Coordination Center. *http://www.cert.org/*, 2006.

[15] National Vulnerability Database. *http://nvd.nist.gov/*, 2006.

[16] Open-Source Vulnerability Database. *http://www.osvdb.org/*, 2006.

[17] Will Dormann and Jason Rafail. Securing your web browser. Technical report, CERT Coordination Center, 2006.

[18] Norman E. Fenton and Martin Neil. Prediction and control of ada software defects. *Journal of Systems and Software*, 12(3):199–207, July 1990.

[19] Richard Ford, Herbert H. Thompson, and Fabien Casteran. Role comparison report - web server role. Technical report, Security Innovation, 2005.

[20] Rajeev Gopalakrishna and Eugene H. Spafford. A trend analysis of vulnerabilities. Technical report, Purdue University, 2005.

[21] Stefano Grazioli and Sirkka L. Jarvenpaa. Perils of internet fraud: An empirical investigation of deception and trust with experienced internet consumers. *IEEE Transactions on Systems, Man, and Cybernetics Part A:Systems and Humans.*, 30(4):395–410, 2004.

[22] Gianluigi Greco, Sergio Greco, and Ester Zumpano. Collaborative filtering supporting web site navigation. *AI Communications*, 17:155–166, 2004.

[23] O. Hallaraker and G. Vigna. Detecting malicious JavaScript code in Mozilla. *Engineering of Complex Computer Systems*, June 2005.

[24] Jonas Hallberg, Amund Hunstad, and Mikael Peterson. A framework for system security assessment. *Systems, Man and Cybernetics (SMC) Information Assurance Workshop*, pages 224–231, June 2005.

[25] Les Hatton. Reexamining the fault density-component size connection. *IEEE Software*, 14(2):89–97, March 1997.

[26] Frank Kargl, Jörn Maier, and Michael Weber. Protecting web servers from distributed denial of service attacks. In *World Wide Web*, pages 514–524, 2001.

[27] Abhishek Kumar. Phishing - a new age weapon. Technical report, Open Web Application Security Project (OWASP), 2005.

[28] Carl E. Landwehr, Alan R. Bull, John P. McDermott, and William S. Choi. A taxonomy of computer program security flaws. *ACM Computer Survey*, 26(3):211–254, 1994.

[29] Susan C. Lee and Lauren B. Davis. Learning from experience: Operating system vulnerability trends. *IT Professional*, 05(1):17–24, January 2003.

[30] Antoinette Leung, Zhuang Yan, and Simon Fong. On designing a flexible e-payment system with fraud detection capability. *2004 IEEE International Conference on E-Commerce Technology*, pages 236–243, 2004.

[31] Bev Littlewood, Sarah Brocklehurst, Norman E. Fenton, P. Mellor, Stella Page, David Wright, J. Dobson, J. McDermid, and Dieter Gollmann. Towards operational measures of computer security. *Journal of Computer Security*, 2(2-3):211–230, 1993.

[32] Bharat B. Madan, Katerina Goseva-Popstojanova, Kalyanaraman Vaidyanathan, and Kishor S. Trivedi. A method for modeling and quantifying the security attributes of intrusion tolerant systems. *Perform. Eval.*, 56(1-4):167–186, 2004.

[33] Noriyuki Matsuda, Tsukasa Hirashima, Toyohiro Nomoto, Hirokazu Taki, and Jun'ichi Toyoda. Context-sensitive filtering for the web. *Web Intelligence and Agent Systems*, September 2003.

[34] Trend Micro. Vulnerability exploits break records. Technical report, Trend Micro, 2005.

[35] David Moore, Colleen Shannon, and Kimberly C. Claffy. Code-Red: a case study on the spread and victims of an internet worm. *Internet Measurement Workshop*, pages 273–284, 2002.

[36] Alexander Moshchuk, Tanya Bragin, Steven D. Gribble, and Henry M. Levy. A crawler-based study of spyware on the web. *The 13th Annual Network and Distributed System Security Symposium*, 2006.

[37] J. D. Musa. *Software Reliability Engineering*. McGraw-Hill, 1999.

[38] Ryan Naraine. Zero-day Firefox exploit sends Mozilla scrambling. *http://www.eweek.com/article2/0,1759,1814056,00.asp*, May 2005.

[39] Netcraft. *http://news.netcraft.com/*, 2006.

[40] Real World Impact of IE Flaw. Real world impact of ie flaw. April 2006.

[41] C. P. Pfleeger. *Security in Computing*. Prentice-Hall, 1997.

[42] Eric Rescorla. Security holes... who cares? *In Proceedings of the 12th USENIX Security Symposium*, pages 75–90, 2003.

[43] Eric Rescorla. Is finding security holes a good idea? *IEEE Security and Privacy*, 03(1):14–19, 2005.

[44] Symantec Vulnerability Research. *http://www.symantec.com/enterprise/research/index.jsp*, 2006.

[45] Jarrett Rosenberg. Some misconceptions about lines of code. *In Proceedings of the 4th IEEE International Software Metrics Symposium*, November 1997.

[46] M. Sahinoglu. Quantitative risk assessment for dependent vulnerabilities. *In Proceedings of Reliability and Maintainability Symposium.*

[47] E. E. Schultz and D. S. Brownand T. A. Longstaff. Responding to computer security incidents. Technical report, Lawrence Livemore National Laboratory, July 1990.

[48] E. Eugene Schultz. Internet Explorer security: is there any hope? *Network Security*, pages 6–10, January 2005.

[49] Scurityfocus. *http://www.securityfocus.com/*, 2006.

[50] R. Seacord. *Secure Coding in C and C++*. Addison Wisely, 2005.

[51] R. C. Seacord and A. D. Householder. A structured approach to classifying vulnerabilities. Technical Report CMU/SEI-2005-TN-003, Carnegie Mellon, 2005.

[52] Secunia. *http://secunia.com/*, April 2006.

[53] SLOCCount. *http://www.dwheeler.com/sloccount/*, 2005.

[54] Symantec. Symantec Internet security threat report: Trends for July 05–December 05. Technical report, Symantec, 2006.

[55] Common Vulnerabilities and Exposures. *http://www.cve.mitre.org/*, 2006.

[56] Sung-Whan Woo, Omar H. Alhazmi, and Yashwant K. Malaiya. An analysis of the vulnerability discovery process in web browsers. *to appear in the SEA 2006 the 10th IASTED International Conference on Software Engineering and Applications*, November 2006.

[57] Sung-Whan Woo, Omar H. Alhazmi, and Yashwant K. Malaiya. Assessing vulnerabilities in Apache and IIS http servers. *In Proceedings the 2nd IEEE International Symposium on Dependable Autonomic and Secure Computing (DASC'06)*, September 2006.

[58] CERT Advisory CA-2001-26 Nimda Worm. *http://www.cert.org/advisories/CA-2001-26.html*, September 2001.