# Application of Vulnerability Discovery Models to Major Operating Systems

Omar H. Alhazmi and Yashwant K. Malaiya, *Senior Member, IEEE*

***Abstract*—A number of security vulnerabilities have been reported in the Windows, and Linux operating systems. Both the developers, and users of operating systems have to utilize significant resources to evaluate, and mitigate the risk posed by these vulnerabilities. Vulnerabilities are discovered throughout the life of a software system by both the developers, and external testers. Vulnerability discovery models are needed that describe the vulnerability discovery process for determining readiness for release, future resource allocation for patch development, and evaluating the risk of vulnerability exploitation. Here, we analytically describe six models that have been recently proposed, and evaluate those using actual data for four major operating systems. The applicability of the proposed models, and the significance of the parameters involved are examined. The results show that some of the models tend to capture the discovery process better than others.**

***Index Terms*—Operating systems, security, software reliability growth models, vulnerabilities, vulnerability discovery.**

ACRONYM[1]

| | |
|---|---|
| AIC | Akaike Information Criteria |
| AT | Anderson Thermodynamic Model |
| AML | Alhazmi-Malaiya Logistic Model |
| LM | Linear Model |
| LP | Logarithmic Poisson Model |
| RQ | Rescorla Quadratic Model |
| RE | Rescorla Exponential Model |
| RSS | Residuals Sum of Squares |
| SRGM | Software Reliability Growth Model |
| VDM | Vulnerability Discovery Model |
| VEM | Vulnerability Exploitation Model |

NOTATION

| | |
|---|---|
| $\omega(t)$ | the rate of vulnerabilities discovery at time t |
| $\Omega(t)$ | the cumulative number of vulnerabilities at time t |
| t | time |
| $\lambda$ | rate constant for the RE model |
| $\gamma$, K | parameters of the AT model |
| $\beta_0, \beta_1$ | parameters of the LP model |
| $o_i, e_i$ | observed, and expected ith value |
| C | integration constant for AT, and AML models |
| b, k | parameters of the RQ model |
| T | the number of observations |
| M | the number of parameters in a model |
| v, u | the parameters of the linear model |

Manuscript received October 18, 2006; revised May 7, 2007; accepted May 26, 2007. Associate Editor: N. Ye.

The authors are with the Computer Science Department, Colorado State University, Fort Collins, CO 80523-1873 USA (e-mail: omar@cs.colostate.edu; malaiya@cs.colostate.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TR.2008.916872

[1]The singular and plural of an acronym are always spelled the same.

## I. INTRODUCTION

SECURITY vulnerabilities are a class of software defects that can lead to security violations [24]. A *vulnerability* is defined as "a defect which enables an attacker to bypass security measures" [24]. Thus, the vulnerability discovery process is similar to the process of finding ordinary (non-security related) defects in software during testing. However, there are some remarkable differences. The ordinary defects encountered after release generally do not represent the high degree of risk as vulnerabilities. A significant fraction of the vulnerabilities is found by testers working externally. Some of the finders are experts in commercial security organizations; other finders may be potential "black-hat" individuals who are tempted to use the vulnerabilities discovered for their own gain. Ordinary defects are often not fixed until the next release. However, because the presence of a known, un-remedied vulnerability creates great risks, the system developers need to release patches as soon as possible after a vulnerability is discovered. The presence of known vulnerabilities can represent an extremely high risk for some organizations such as banks, investment & brokerage houses, and web-based merchants.

The software developers, and users need to be able to assess the risk posed by the vulnerabilities, and must invest in effective counter-measures. The risk increases with the delay in developing, and releasing a patch [9], [13]. A developer needs to allocate sufficient resources for continuous vulnerability testing, and patch development to stay ahead of the hackers. The users need to invest in data safeguard mechanisms, intrusion detection, and damage control. This investment must be proportional to the level of risk involved.

Software reliability growth models [16], [20] have been used for characterizing the defect-finding process for ordinary defects. Such models are used to assess the test resources needed to achieve the desired reliability level by the target date, and are needed for evaluating the reliability level achieved. They can also be used to estimate the number of residual defects that are likely to be present. There is a need to develop similar models for quantitative characterization of the security aspects of the software. There are two separate processes to be considered: the first is the vulnerability discovery process, while the second is the exploitation of the individual vulnerabilities discovered. In this paper, we examine modeling the first process. An evaluation of the overall risk should involve a joint consideration of both processes. Obviously, a vulnerability needs to be discovered before it can be exploited. Those who attempt to exploit vulnerabilities can often be amateurs because they can use the hacking scripts available on the Internet, which are developed after a vulnerability has been reported. On the other hand, those who discover new vulnerabilities must have significant technical expertise, because the vulnerabilities often arise as the result of complex interactions of rarely occurring state combinations in the software.

Quantitative modeling of security vulnerabilities has begun only recently. Widespread use of the Internet, including Internet banking, emerged around 1995. Vulnerabilities soon became a subject of considerable concern. A characterization of both the vulnerability discovery process, and the vulnerability exploitation process is needed to assess the security risk. *Vulnerability exploitation models* (VEM) were the first considered. Browne *et al.* [12] have examined the exploitation of some specific vulnerabilities, and have presented a modeling scheme. Beginning in 2002, a few researchers have started investigating the modeling approaches for the vulnerability finding process [2], [8], [22], [23]. We now have enough data, in some cases going back to 1996, to examine & evaluate the proposed *vulnerability discovery models* (VDM). A complete evaluation of the risk will involve both VEM, and VDM.

The first VDM model proposed by Anderson [8] is here termed the Anderson Thermodynamic (AT) model. The second, termed the AML model, is a logistic model proposed by Alhazmi & Malaiya in [2], and investigated in [3]; its prediction capabilities were examined in [6]. Two possible trends were examined by Rescola in [23], which result in a quadratic model (RQ), and an exponential model (RE). In addition, we examine an LP model which is an application of a traditional Logarithmic Poisson software reliability growth model proposed by Musa & Okumoto [21]. Finally, we test a simple linear model (LM) as an approximation of the AML model. All of these can be termed time-based models because they consider calendar time as the main factor. An effort-based model has also been proposed by Alhazmi & Malaiya in [2]. It will not be considered in this paper because it utilizes a different approach that attempts to use test-effort as the main factor rather than calendar time. In this paper, we briefly examine the bases of the proposed models, and evaluate the applicability of these models using actual vulnerability data.

Just as static models for defect density and fault exposure ratio can assist in the use of SRGM, the metrics vulnerability density, and vulnerability/defect ratio can be applied to complement, and support VDM. Alhazmi & Malaiya [2] have shown that, for similar systems, the values of these attributes tend to fall within a range. Static metrics can aid to constrain parameter estimation during fitting [4].

This paper examines the proposed models, and presents a comparison using actual data for vulnerabilities in four major operating systems. The statistical goodness of fit test is used to examine how well models track the actual discovery process. In the next section, we discuss the proposed models. In Section III, the data sets used for evaluation, and the methodology are described. In Section IV, the models' adequacies are examined; the section evaluates measures for goodness of fit, and examines the findings. Finally, we present the conclusions, and identify future work.

## II. Vulnerability Discovery Models

Here, we examine the main features of the vulnerability discovery models (VDM) proposed. Although some of the VDM were not formally termed "models," they are appropriate candidates for further examination to see how well they describe the discovery process. For uniformity, we present these models in terms of the cumulative number of vulnerabilities, with time as the independent variable. VDM can describe the rate of vulnerability discovery, denoted by $\omega(t)$, or the cumulative number of vulnerabilities discovered, denoted by $\Omega(t)$. Note that $\Omega(t)$ *is* obtained by integrating $\omega(t)$ with respect to time.

*Anderson Thermodynamic Model (AT):* This model was originally proposed for ordinary defects in [11], and later it was applied to vulnerabilities [8]. Suppose there are $N(t)$ vulnerabilities left after $t$ tests, and let the probability that a test fails be $\omega(t)$. The model assumes that encountering a vulnerability causes it to be removed, and that no bugs are reintroduced. Using an analogy from thermodynamics, Anderson obtains the model

$$\omega(t) = \frac{K}{\gamma t}, \tag{1}$$

where $\gamma$ takes into account the lower failure rate during beta testing by users compared with higher rates during alpha testing. Because we want to compare cumulative models, we integrate (1) to get the model in terms of the cumulative number of vulnerabilities given by the function $\Omega(t)$ as

$$\Omega(t) = \int \left(\frac{K}{\gamma t}\right) dt$$
$$\Omega(t) = \frac{K}{\gamma} \ln(Ct) \tag{2}$$

Note that $\Omega(t)$ is not defined when t $= 0$; hence, we will only consider its applicability when t $\geq 1$. As t grows, $\Omega(t)$ grows logarithmically. Note that this model has a relationship to the well-known Logarithmic Poisson SRGM, and the failure rate bound proposed by Bishop & Bloomfield [10]. Fig. 1 shows a hypothetical plot of the AT model for different values of $k/\gamma$, and $C$.

*Alhazmi-Malaiya Logistic Model (AML):* This model was proposed by Alhazmi & Malaiya [2]. The AML model is based on the observation that the attention given to an operating system increases after its introduction, peaks at some time, and then drops because of the introduction of a newer competing version. The cumulative number of vulnerabilities thus shows an increasing rate at the beginning as the system starts attracting an increasing share of the installed base. After some time, a steady rate of vulnerability finding yields a linear curve. Eventually the vulnerability discovery rate starts dropping due both to reduced attention, and a smaller pool of remaining vulnerabilities.

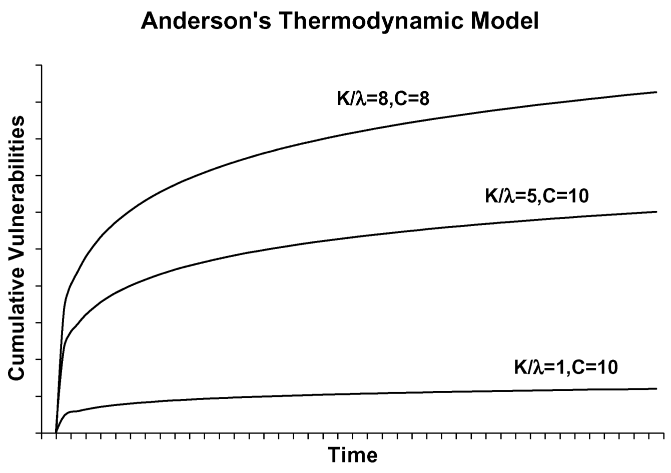


Fig. 1. The Anderson Thermodynamic (AT) model.

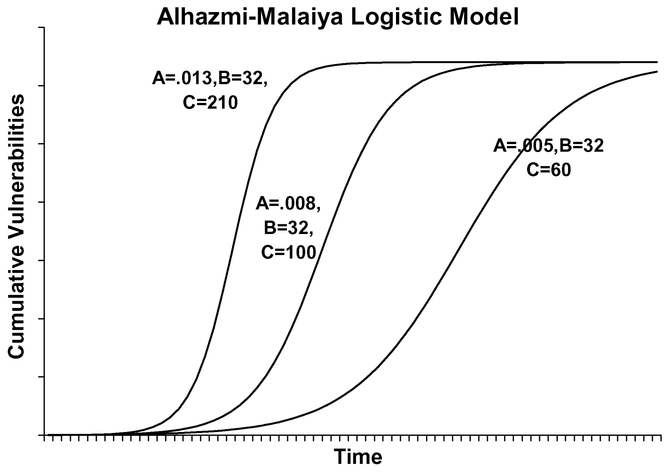


Fig. 2. The Alhazmi-Malaiya Logistic (AML) model.

This model assumes that the rate of change of the cumulative number of vulnerabilities $\Omega$ is governed by two factors, as given in (3). One of these factors 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.

Let us assume that the vulnerability discovery rate is given by the differential equation

$$\frac{d\Omega}{dt} = A\Omega(B - \Omega) \tag{3}$$

where $t = 0$ initially, and $A$ $B$ are empirically determined from the recorded data. By solving (3), we obtain

$$\Omega(t) = \frac{B}{BCe^{-ABt} + 1}. \tag{4}$$

$C$ is introduced while solving (3). It is thus a three-parameter model given by a logistic function. In (4), as $t$ approaches infinity, $y$ approaches $B$. Thus, the parameter $B$ represents the total number of accumulated vulnerabilities that will eventually be found. The model given by (4) will be referred to as the Alhazmi-Malaiya Logistic (AML) model [2].

The AML model addresses the fact that the vulnerabilities found in an operating system depend on the system's individual usage environment. The saturation phase might not be seen in an OS that has not been present for a sufficiently long time. In addition, if the initial adaptation is quick due to better prior publicity, the early learning phase may not be seen.

An initial estimate of B may be obtained by noting the size of the software, and using the typical vulnerability density values of similar software. Using (2), we can show that the maximum slope occurs at $y = B/2$. It can be shown that the two inflexion points in the derivative of $\Omega$ are $2.63/AB$ time units apart. This fact can be used to guide parameter estimation during fitting [4]. Fig. 2 shows a hypothetical plot of the AML model for different values of $A$, $B$, and $C$.

Alhazmi & Malaiya have also proposed an effort-based model (AMEB) [2]. However, it is not comparable to the other models examined here.

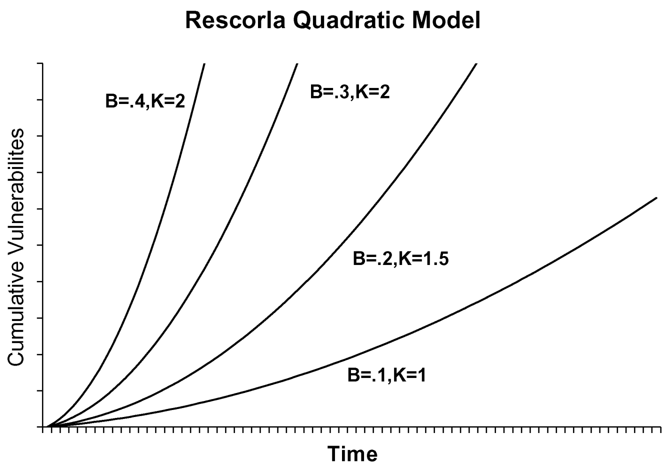


Fig. 3. The Rescorla Quadratic (RQ) model.

*Rescorla Quadratic Model (RQ):* Rescorla has attempted to identify trends in the vulnerability discovery data by applying some statistical tests. We here refer to these tests as the quadratic, and exponential models [23].

Rescorla first examined the possibility of the vulnerability finding rate varying linearly with time, which implies that

$$\omega(t) = bt + k. \tag{5}$$

The cumulative vulnerability discovery model can be derived by integrating (5) to get

$$\Omega(t) = \frac{bt^2}{2} + kt. \tag{6}$$

Here, the integration constant is taken to be zero to allow $\Omega(t)$ to be zero at $t = 0$. In this model, as t grows, $\Omega$ grows quadratically, thus it is called the Rescorla Quadratic model. Fig. 3 shows a hypothetical plot of the RQ model for different values of $b$, and $k$.

*Rescorla Exponential Model (RE):* Rescorla [23] has also used the Goel-Okumoto SRGM [14], which can be given as

$$\omega(t) = N\lambda e^{-\lambda t}. \tag{7}$$

TABLE I
THE OPERATING SYSTEM DATA SETS

| Systems | *Lines of code (millions)* | *OS Type* | *Known Vulnerabilities* | *Vulnerability Density (per Ksloc)* | *Release Date* |
|---|---|---|---|---|---|
| **Windows 95** | 15 | Commercial client | 51 | 0.0034 | Aug 1995 |
| **Windows XP** | 40 | Commercial client | 173 | 0.0043 | Oct 2001 |
| **R H Linux 6.2** | 17 | Open-source | 118 | 0.00694 | Mar 2000 |
| **R H Fedora** | 76 | Open-source server | 154 | 0.00203 | Nov 2003 |

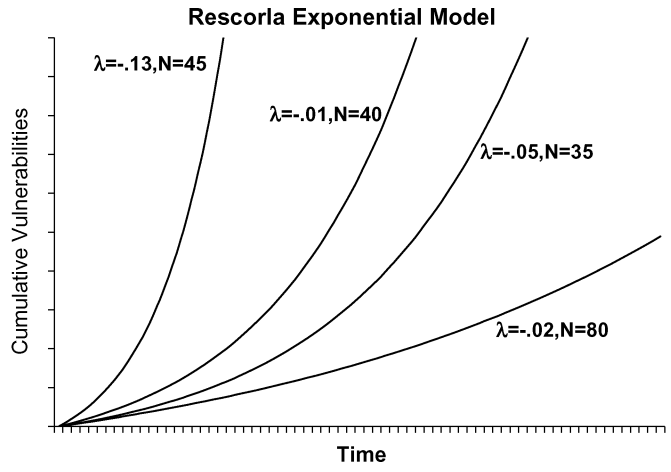


Fig. 4. The Rescorla Exponential (RE) model.

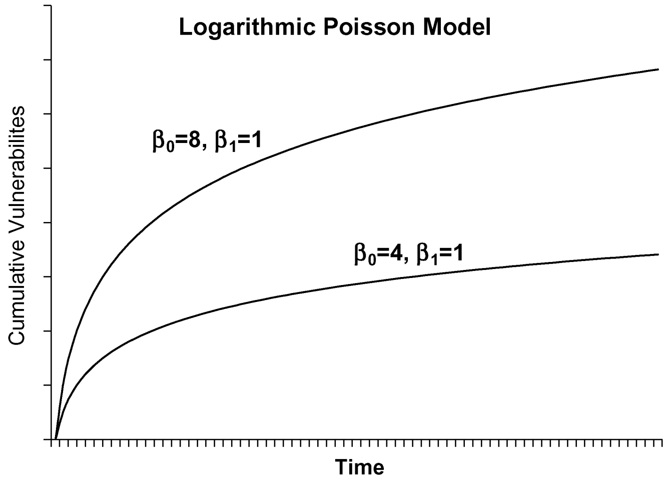


Fig. 5. The Logarithmic Poisson (LP) model.

Again, we integrate (7) to get the expression for the cumulative number of vulnerabilities.

$$\Omega(t) = N(1 - e^{-\lambda t}), \tag{8}$$

where the integration constant has been equated to N to allow the initial value of $\Omega$ to be zero. Note that, as time increases, $\Omega$ approaches N. Fig. 4 gives the plots for the RE model for several values of $\lambda$, and $N$.

*Logarithmic Poisson Model (LP):* In software reliability engineering, this model is also known as the Musa-Okumoto model [21]. A physical interpretation of the model and its parameters is complex. An interpretation in terms of the variability of the fault exposure ratio is given in [17] as

$$\Omega(t) = \beta_0 \ln(1 + \beta_1 t). \tag{9}$$

It is obvious that at $t = 0$, $\Omega(t) = 0$; $\Omega(t)$ grows indefinitely as the system ages with a logarithmic growth. In spite of the fact that the parameters have a complex interpretation, in many cases this model has been found to be among the better fitting SRGM. Fig. 5 shows various plots of the LP model for different values of $\beta_0$, and $\beta_1$.

*Linear Model (LM):* This model was suggested in [2] as an approximation to the logistic model. It was found to fit well when the learning phase is short, and when the saturation has not been reached, or when vulnerabilities are shared with a successor version of the software. The model is

$$\Omega(t) = vt + u, \tag{10}$$

where v, and u are regression coefficients. At $t = 0$, and $\Omega(t) = 0$, $\Omega(t)$ grows indefinitely as the system ages with linear growth.

## III. METHODOLOGY FOR EVALUATING THE PROPOSED MODELS' APPLICABILITY

Here we discuss how the data were collected & prepared for fitting, and then we describe how the goodness of fit was evaluated.

*The data sources:* Compared to data that have been used for SRGMS in the past, the vulnerability data available have some different characteristics. One of the main differences is that generally no information is available concerning the faults in the SRGM data, whereas for vulnerabilities the databases identify the specific vulnerability. The vulnerability data come from several well-known products, because the data for every operating system, and server, whether commercial, or open-source, are available. The SRGM data come only from some selected projects where the management has permitted disclosure of the data. On the other hand, vulnerability data have some limitations; namely, that they come from a limited number of sources, and the number of vulnerabilities typically represents only a small fraction of the total number of defects.

In our analysis, we have used the data sets for four different operating systems [3], [7], [15], [18], [19], shown in Table I. The vulnerability data of Windows 95 is for a client operating system that has existed for several years. It has gone through nearly a complete life cycle, and its remaining installed base is now very small. The data set for Windows XP represents a relatively new operating system, which may be near the peak of its popularity

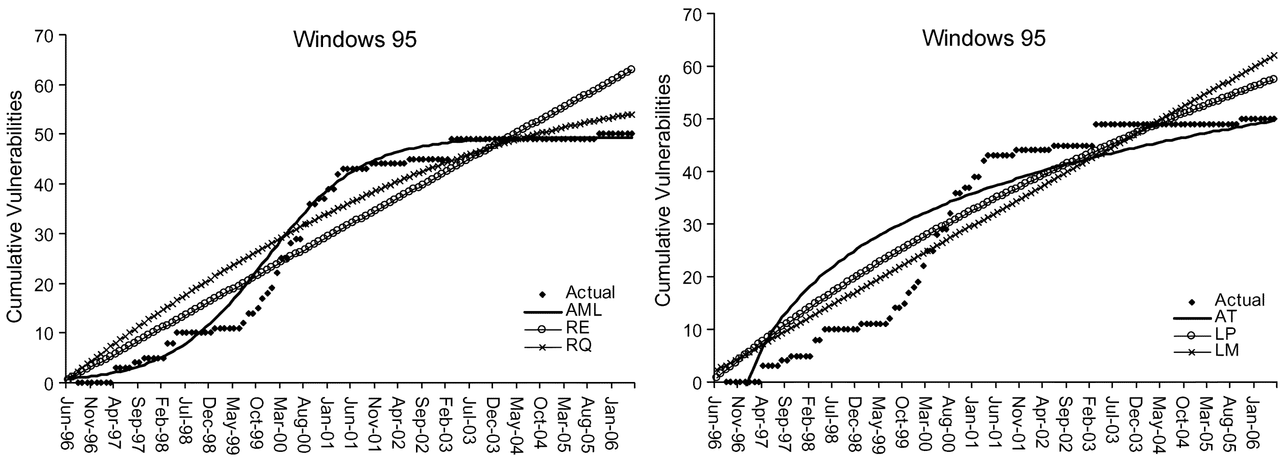


Fig. 6. Fitted VDM with actual Windows 95 data.

TABLE II
WINDOWS 95 GOODNESS OF FIT RESULTS

| Model | Parameters | | | $\chi^2$–test | | RSS | AIC |
|---|---|---|---|---|---|---|---|
| | | | | $\chi^2$ | P-Value | | |
| *AT | K/γ | C | | 226.46 | 0 | 7992.3 | 1082.35 |
| | 18 | 2.36 | | | | | |
| LP | β0 | β1 | | 178.18 | .00036 | 3974.4 | 998.52 |
| | 69.00863716 | 1.08E-02 | | | | | |
| LM | u | v | | 202.44 | .000003 | 5313.9 | 1033.37 |
| | 1.690 | 0.5032016 | | | | | |
| RE | λ | N | | 181.47 | .0002 | 5359.95 | 1034.41 |
| | .000094 | 5629.743 | | | | | |
| RQ | S | K | | 169.96 | .0015 | 3429.55 | 980.82 |
| | -0.00249 | 0.749276 | | | | | |
| AML | A | B | C | 46.93 | 1 | 422.9 | 731.65 |
| | 0.002 | 49.37396 | 1.358819937 | | | | |

*The Chi-square test was applied to the positive values of the AT, and LM models.

in 2007. We have also included data for Linux Red Hat 6.2, which represents an open-source operating system. Some of the key attributes of the four systems are given in Table I. The vulnerability density is the number of discovered vulnerabilities per thousand source lines of code (Ksloc). Windows XP is much larger than Windows 95; however, its vulnerability density is comparable. It is likely that a significant number of thus far undiscovered vulnerabilities are present in Windows XP. The higher number of vulnerabilities in Red Hat Linux 6.2 may be because a larger fraction of its code is devoted to access control. Fedora is a relatively new version of Linux.

Vulnerability data need to be manually extracted from databases. Our major sources of data are the MITRE Corporation website [19], and the National Vulnerabilities Database (NVD) Metabase [15]. NVD is an easily searchable database with the option of downloading an ACCESS database.

*Evaluation of goodness of fit:* We will apply two goodness of fit tests. The first is the chi-square goodness of fit test. The chi-square $(\chi^2)$ statistic is calculated as

$$\chi^2 = \sum_{i=1}^{n} \frac{(o_i - e_i)^2}{e_i}. \tag{11}$$

For the fit to be acceptable, the chi-square statistic should be less than the critical value for a given alpha level, and degrees of freedom. The P-value represents the probability that a value of the $\chi^2$ statistic at least as high as the value calculated by the above formula could have occurred by chance. We use an alpha level of 5%; i.e., if the P-value of the chi-square test is below 0.05, then the fit will be rejected. A P-value closer to 1 indicates a better fit. The P-value is calculated by using the number of degrees of freedom of the data set, and chi-square distribution.

For model adequacy testing, *Akaike Information Criteria (AIC)* is also frequently used, and is considered an unbiased measure [1]. The AIC methodology attempts to find the minimal model that correctly explains the data. $AIC$ is formally defined as

$$AIC = (-2 \times log\ likelihood) + 2M.$$

When the residuals are approximately s-normally distributed, which is the case here, an equivalent way to compute AIC is

$$\mathrm{AIC} = T\, ln(\mathrm{RSS/T}) + 2M. \tag{12}$$

We use the formulation of AIC given by (11).

## IV. FITTING DATA TO PROPOSED MODELS

The results of fitting the models to the data are presented graphically in the plots given in Figs. 6–9, which show the fitted plots together with actual cumulative data. For convenience in

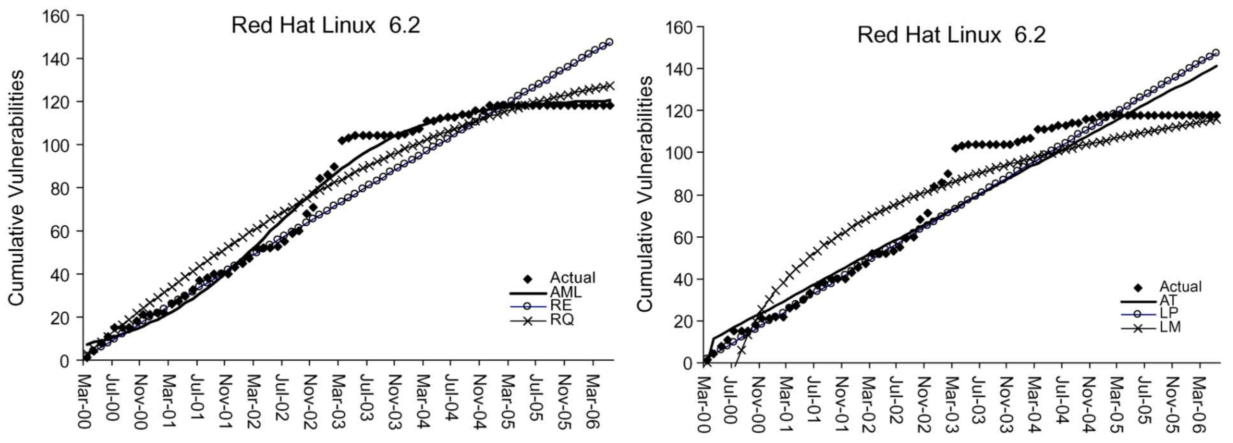


Fig. 7. Fitted VDM with actual Red Hat Linux 6.2 data.

TABLE III
FITTING RESULTS FOR RED HAT LINUX 6.2

| Model | Parameters | | | $\chi^2$–test | | RSS | AIC |
|---|---|---|---|---|---|---|---|
| | | | | $\chi^2$ | P-Value | | |
| *AT | K/γ | C | | 199.35 | 0 | 19469.39 | 744.75 |
| | 40.7745502 | 9.438709877 | | | | | |
| *LM | u | v | | 144.087 | 0 | 10635.6 | 699.397 |
| | 9.591 | 1.7812802 | | | | | |
| LP | $\beta_0$ | $\beta_1$ | | 131.02 | 0 | 12073.63 | 708.91 |
| | 6999.939 | 2.84E-04 | | | | | |
| RE | λ | N | | 130.24 | 0 | 12009.071 | 708.51 |
| | 0.000354 | 5628.812 | | | | | |
| RQ | S | K | | 81.35 | 0.261 | 5389.545 | 648.42 |
| | -0.01504 | 2.823606 | | | | | |
| AML | A | B | C | 35.52 | 1 | 1589.1 | 558.82 |
| | 0.000855 | 121.235 | 0.139727427 | | | | |

*The Chi-square test was applied to the positive values of the AT, and LM models.

viewing, the plots for each operating system are shown in two separate figures given side by side. In addition, the parameter values obtained during the fit, and the corresponding measures of goodness of fit for the four operating systems' sets, are given in Tables II–V. First, we discuss the results for each dataset; we then present observations for each of the models.

*Observations on Datasets:* The Windows 95 data (Fig. 6) has a distinct "s"-shape because vulnerability detection reached saturation in 2003. As we would expect, the AML model fits quite well. The fitted RE, RQ, LP, and LM models give plots that look linear, and thus show considerable divergence at the end. The fitted AT model gives negative values at the beginning, and significantly diverges from the actual data except near the end, which is why its chi-square value is significantly higher than the other models. The AML model has the lowest AIC of 731.65, followed by RQ with 980.82, and then by other models ranging from 998.52 to 1082.35.

The Linux Red Hat 6.2 data (see Fig. 7) shows a milder "s"-shape. This allows AML, and RQ to fit the data. The AML model shows a significant fit with $\mathrm{P-value} = 1$. The RQ model was able to fit the data with a P-value of 0.261 (see Table III), because the data have just entered the saturation phase, and thus, unlike other models, RQ was able to bend with the saturation. On the AIC scale, AML gave the best AIC test score of 558.82, followed by RQ at 648.42, while the other models yielded higher numbers.

The data for Windows XP (Fig. 8) shows a very linear trend, allowing RQ to fit quite well with the P-value of 0.971, leaving AML with a P-value of only 0.147 (see Table IV). This shows a less significant fit than RQ because the data for Windows XP does not yet have a strong "s"-shape. Windows XP is a relatively new operating system, and the saturation phase had not been reached yet; consequently, the AML model does not yield the best fit. Other models were unable to provide a significant fit. On the AIC scale, RQ has shown a better AIC of only 492.10 compared to 511.47 of AML; other models have yielded higher AIC values ranging from 661.80 to 745.20. By this measure, also, the other models provided a poor fit.

In Red Hat Fedora (see Fig. 9), AML fit with an AIC of 0.426 (see Table V), while other models were unable to fit the data. This shows that Red Hat Fedora has begun to assume the "s"-shape, but it is still in an early stage, allowing the AML model to fit; furthermore, it can be expected that as Red Hat Fedora data start to mature further, the P-value will eventually increase. Additionally, AML has an AIC of 347.68, while the other models' AIC ranged from 388.08 to 449.90.

*Comparative Performance of the Models:* Here we examine the performance of each individual model.

The Anderson Thermodynamic (AT) model did not fit any of the data; it exhibited the highest AIC scores, and lowest P-values. For the Windows 95 data, its AIC of 1082.35 is 48 points away from the nearest model. For Linux 6.2, it yields

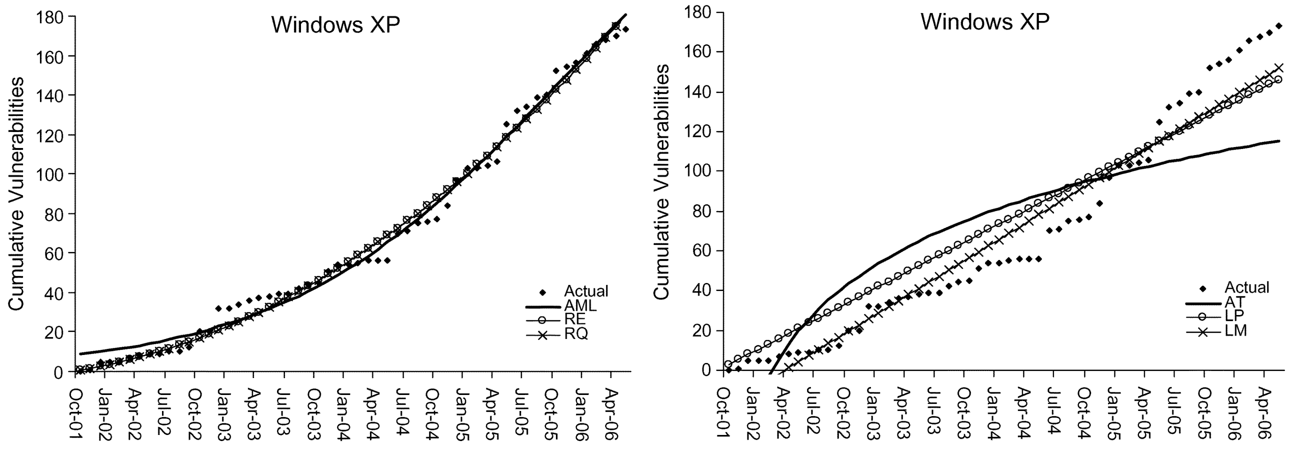


Fig. 8. Fitted VDM with actual Windows XP data.

TABLE IV
FITTING RESULTS FOR WINDOWS XP

| Model | Parameters | | | $\chi^2$–test | | RSS | AIC |
|---|---|---|---|---|---|---|---|
| | | | | $\chi^2$ | P-Value | | |
| *AT | K/γ | C | | 524.59 | 0 | 54178.7 | 745.20 |
| | 49.251 | 9.186 | | | | | |
| LM | u | v | | 127.18 | 0.006916 | 8928.76 | 622.67 |
| | -20.412 | 3.0779474 | | | | | |
| LP | $\beta_0$ | $\beta_1$ | | 243.04 | 0 | 16018.76 | 662.34 |
| | 11757.872 | 0.0002234 | | | | | |
| RE | λ | N | | 241.42 | 0 | 15890.74 | 661.80 |
| | 6.9301E-05 | 37790.660 | | | | | |
| RQ | S | K | | 36.86 | 0.971 | 1655.98 | 492.10 |
| | 0.044 | 0.734 | | | | | |
| AML | A | B | C | 65.99 | 0.147 | 1691.58 | 511.47 |
| | 0.0003 | 288.7025 | 0.1206 | | | | |

*The Chi-square test was applied to the positive values of the AT, and LM models.

an AIC of 744.75, which is 36 points higher than the nearest model. For Windows XP, and Red Hat Fedora, AIC scores were also significantly higher than the nearest model.

The Rescorla Quadratic (RQ) model was able to fit two out of four of the datasets under consideration. However, the RQ was not able to fit either the Windows 95, or Red Hat Fedora datasets (see Figs. 6, and 9) due to the strong “s”-shaped trend of the data, although the model yielded an arc-shaped curve with the ability to show saturation at the end. Thus, for both datasets, it scored poorly in the chi-square test with a P-value close to zero, whereas it should have been at least 0.05 to be acceptable. For Windows XP data, it achieved a P-value of 0.971, which is a very good fit. It fit the Linux 6.2 data with a P-value of 0.261.

The fit was poor for the chi-square test with P-values within 0 to 0.0002, which is significantly less than the acceptable P-value of 0.05. The RE scores were among the highest, except when compared with AT.

The Linear Model (LM) was unable to fit any of the datasets. However, its AIC score was better than RE, and AT most of the time. Nonetheless, the Alhazmi-Malaiya Logistic Model (AML) was the only model that fit all four datasets. The fit was especially superior for the Windows 95, and Linux Red Hat 6.2 datasets. However, the fit was not as good for Red Hat Fedora, and Windows XP, where the P-values were 0.426, and 0.147, respectively. Moreover, the AML model yielded significantly better AIC scores for all datasets, except for Windows XP where the AML model was second after the RQ model.

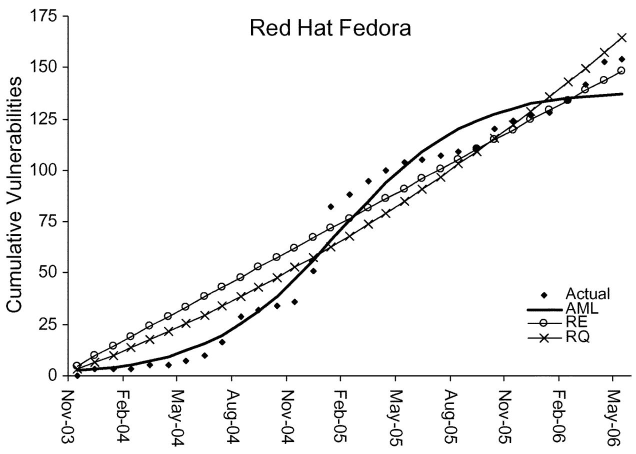
The other four models failed the goodness of fit test for Windows 95 because they generated unacceptably high chi-square values, and consequently low P-values below 0.02, although the Windows XP vulnerability discovery rate is expected to decline eventually. The results show that AML is the most consistent model for the four data sets.

The Logarithmic Poisson (LP) model was unable to fit all four examined datasets, with p-values ranging from 0 to 0.00036.

We note here that LP, RE, and even RQ can perform better if only partial data are used before the onset of saturation, when we expect the data to be somewhat linear, and hence models such as LP, RE, RQ, and LM can easily fit the data [5].

## V. CONCLUSIONS, AND FUTURE WORK

Several vulnerability discovery models were examined using AIC, and chi-square tests. The evaluation found that the AML model is generally best for the longer term, performing better for systems such as Windows 95, Red Hat Linux 6.2, and Red Hat Fedora. Because it captures the “s”-shape pattern in the data, it has better fit as determined by using AIC, and the chi-square test. RQ is a very good fit for Windows XP, a system that has not yet shown signs of saturation. This can be attributed to the fact that the model can fit trends that are largely linear with



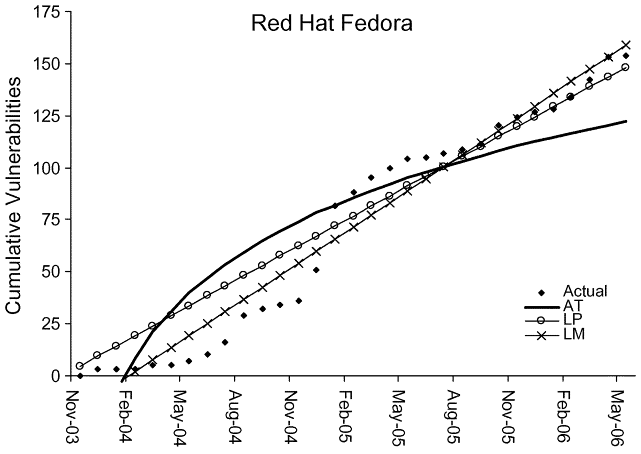


Fig. 9. Fitted VDM with actual Fedora data.

TABLE V
FITTING RESULTS FOR RED HAT FEDORA

| **Model** | **Parameters** | | | **$\chi^2$–test** | | **RSS** | **AIC** |
|---|---|---|---|---|---|---|---|
| | | | | **$\chi^2$** | **P-Value** | | |
| ***AT** | $K/\gamma$ | C | | 234.036 | 0 | 20108.948 | 449.90 |
| | 55.496 | 16.214 | | | | | |
| ***LM** | u | v | | 66.423 | 0.00002 | 3592.179 | 372.39 |
| | -21.477 | 5.8120967 | | | | | |
| **LP** | $\beta_0$ | $\beta_1$ | | 178.986 | 0 | 7065.290 | 402.83 |
| | 9188.010 | 0.0005244 | | | | | |
| **RE** | $\lambda$ | N | | 178.051 | 0 | 7013.404 | 402.50 |
| | 0.000127 | 37790.59 | | | | | |
| **RQ** | S | K | | 96.84 | 0 | 5090.857 | 388.08 |
| | 0.070 | 3.131 | | | | | |
| **AML** | A | B | C | 32.195 | 0.426 | 1984.343 | 347.69 |
| | 0.00201 | 139.045 | 0.535 | | | | |

*The Chi-square test was applied to the positive values of the AT, and LM models.

slight saturation. LM, LP, and RE were not able to fit any of the datasets because of the lack of a semi-linear trend. The AT model considered did not perform well in general; it has shown significant diversion from the datasets. Three of the models, RQ, RE, and LP, appear to do a good job of following the shorter-term trends when the cumulative vulnerabilities show a linear trend. The AML model often provides the best fit because it can follow the "s"-shaped trend that is often observed.

We also note that the AML model uses three parameters, while the other five models are two-parameter models. However, AIC takes the number of parameters into account, and thus provides a fair basis for comparison.

Among the five models, the parameters of the RE, and AML models have some simple interpretations. One of the parameters in both is related to the total number of vulnerabilities present in the software. If the expected range of vulnerability density values can be estimated based on past experience, a preliminary estimate of the total number of vulnerabilities may be empirically obtained [3], [4]. However, empirical estimation of other parameters requires further investigation.

Finding vulnerabilities in a widely installed system should be more rewarding to internal testers, as well as to external experts, and hackers. Thus, the testing effort changes with changes in the market share of software systems. This variability of effort has been addressed in [2] by developing an equivalent effort model. The model addresses changes in the usage environment, which affects the discovery process by considering the cumulative time spent by the workstations using the software systems. Although the equivalent effort model fit very well, the model requires data that can be very hard to collect. The AML attempts to model the effort variation implicitly.

Vulnerability discovery models assume that each specific release of an operating system is independent, and can be separately modeled. In practice, a significant sharing of the code occurs between successive releases. Thus, a vulnerability detected in a particular version may also exist in previous versions. This will cause vulnerabilities to be found in a version which has been largely replaced by a later version. Further research is needed to model the impact of such shared code. All the models considered here are continuous models. It is possible to define discrete models that may offer some advantages in some cases. Further research is needed to develop, and evaluate discrete models.

In a recent work [4], the prediction capabilities of two of the VDM models were examined, and the results showed a good prediction capability that improves as more data become available. The prediction capability testing showed that putting some constraints on the model's parameters based on previous observations significantly improves the prediction capability. Metrics such as vulnerability density, and vulnerability/defect ratios [3] can be used to check the projections made using VDM during early phases when the available data are insufficient, or when a VDM is known to have some specific limitations. Examination of the prediction capabilities of the other models is still needed to give a broader comparison with the other vulnerability discovery models.

Both the developers, and the user community can use vulnerability discovery models. Developers can assess the product readiness by projecting upcoming vulnerability discovery trends. Developers need to allocate security maintenance resources to detect vulnerabilities, preferably before others do, and to release security patches as soon as possible. The users also need to assess the risk due to vulnerabilities before patches are applied. A patch may need to be tested for stability before it is applied, as discussed by Brykczynski *et al.* [13], and Beattie *et al.* [9]. An organization's effective security policy requires both time, and resources; and vulnerability discovery models can be used to guide such policies quantitatively.

## References

[1] H. Akaike, Prediction and Entropy NTIS, Springfield, VA, MRC Technical Summary Report #2397, 1982.

[2] O. H. Alhazmi and Y. K. Malaiya, "Quantitative vulnerability assessment of systems software," in *Proc. Annual Reliability and Maintainability Symposium*, January 2005, pp. 615–620.

[3] O. H. Alhazmi, Y. K. Malaiya, and I. Ray, "Measuring, analyzing and predicting security vulnerabilities in software systems," *Computers and Security Journal*, vol. 26, no. 3, pp. 219–228, May 2007.

[4] O. H. Alhazmi and Y. K. Malaiya, "Prediction capability of vulnerability discovery models," in *Proc. Reliability and Maintainability Symposium*, January 2006, pp. 86–91.

[5] O. H. Alhazmi and Y. K. Malaiya, "Modeling the vulnerability discovery process," in *Proceedings of 16th IEEE International Symposium on Software Reliability Engineering (ISSRE'05)*, 2005, pp. 129–138.

[6] O. H. Alhazmi and Y. K. Malaiya, "Measuring and enhancing prediction capabilities of vulnerabilities discovery models for Apache and IIS HTTP servers," in *Proc. 17th IEEE International Symposium on Software Reliability Engineering (ISSRE'06)*, 2006, pp. 343–352.

[7] J. Amor-Iglesias, J. González-Barahona, G. Robles-Martínez, and I. Herráiz-Tabernero, "Libre software as a field of study," *The European Journal for the Informatics Professional*, vol. VI, no. 3, pp. 13–16, June 2005.

[8] R. J. Anderson, "Security in open versus closed systems—The dance of Boltzmann, Coase and Moore," in *Open Source Software: Economics, Law and Policy*. Toulouse, France: , June 20–21, 2002.

[9] S. Beattie, S. Arnold, C. Cowan, P. Wagle, and C. Wright, "Timing the application of security patches for optimal uptime," in *Proc. LISA XVI*, November 2002, pp. 233–242.

[10] P. G. Bishop and R. E. Bloomfield, "A conservative theory for long-term reliability growth prediction," *IEEE Trans. Reliability*, vol. 45, no. 4, pp. 550–560, Dec. 1996.

[11] R. M. Brady, R. J. Anderson, and R. C. Ball, Murphy's law, the fitness of evolving species, and the limits of software reliability Cambridge University Computer Laboratory, Technical Report No. 471, September 1999 [Online]. Available: http://www.cl.cam.ac.uk/ftp/users/rja14/babtr.pdf

[12] H. K. Browne, W. A. Arbaugh, J. McHugh, and W. L. Fithen, "A trend analysis of exploitation," in *Proc. IEEE Symposium on Security and Privacy, 2001*, May 2001, pp. 214–229.

[13] B. Brykczynski and R. A. Small, "Reducing internet-based intrusions: Effective security patch management," *IEEE Software*, vol. 20, no. 1, pp. 50–57, Jan./Feb. 2003.

[14] A. L. Goel and K. Okumoto, "Time-dependent error detection rate model for software and other performance measures," *IEEE Trans. on Reliability*, vol. R-28, no. 3, pp. 206–211, August 1979.

[15] NVD Metabase July 2006 [Online]. Available: http://nvd.nist.gov

[16] *"Handbook of Software Reliability Engineering,"* M. R. Lyu, Ed., McGraw-Hill, 1995.

[17] Y. K. Malaiya, A. von Mayrhauser, and P. K. Srimani, "An examination of fault exposure ratio," *IEEE Trans. Software Engineering*, pp. 1087–1094, Nov. 1993.

[18] G. McGraw, "From the ground up: The DIMACS software security workshop," *IEEE Security & Privacy*, vol. 1, no. 2, pp. 59–66, March/April 2003.

[19] The MITRE Corporation, February 2005 [Online]. Available: www.mitre.org

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

[21] J. D. Musa and K. Okumoto, "A logarithmic Poisson execution time model for software reliability measurement," in *Proc. 7th International Conference on Software Engineering*, Orlando, FL, 1984, pp. 230–238.

[22] A. Ozment and S. E. Schechter, "Milk or wine: Does software security improve with age?," in *The 15th USENIX Security Symposium*, Vancouver, BC, July 31–August 4 2006.

[23] E. Rescola, "Is finding security holes a good idea?," *Security and Privacy*, pp. 14–19, Jan./Feb. 2005.

[24] E. E. Schultz, Jr., D. S. Brown, and T. A. Longstaff, *Responding to Computer Security Incidents*. : Lawrence Livermore National Laboratory, July 23, 1990 [Online]. Available: ftp://ftp.cert.dfn.de/pub/docs/csir/ihg.ps.gz

**Omar H. Alhazmi** received the Ph.D. from Colorado State University in 2006, and his Master's degree in computer science from Villanova University in 2001. He has published eleven papers on quantitative vulnerability discovery modeling.

**Yashwant K. Malaiya** is a Professor in the Computer Science Department at Colorado State University. He received MS in Physics from Sagar University, MScTech in Electronics from BITS Pilani, and PhD in Electrical Engineering from Utah State University. He has published widely in the areas of fault modeling, software and hardware reliability, testing and testable design, and quantitative security risk evaluation. He has also been a consultant to industry. He was the General Chair of 1993, and 2003 IEEE International Symposium on Software Reliability Engineering ISSRE. He co-edited the IEEECS Technology Series book *Software Reliability Models, Theoretical Developments, Evaluation and Applications*. He is a recipient of the IEEE Third Millennium Medal, and the Computer Society Golden Core award. He is a senior member of the IEEE.