Vulnerability Discovery Modelling With Vulnerability Severity

Ankur Shukla, Basel Katt and Livinus Obiora Nweke
Department of Information Security and Communication Technology (IIK),
Norwegian University of Science and Technology (NTNU),
Gjϕvik, Norway
ankur.shukla@ntnu.no
basel.katt@ntnu.no
livinus.nweke@ntnu.no

***Abstract*—Web browsers are primary targets of attacks because of their extensive uses and the fact that they interact with sensitive data. Vulnerabilities present in a web browser can pose serious risk to millions of users. Thus, it is pertinent to address these vulnerabilities to provide adequate protection for personally identifiable information. Research done in the past has showed that few vulnerability discovery models (VDMs) highlight the characterization of vulnerability discovery process. In these models, severity which is one of the most crucial properties has not been considered. Vulnerabilities can be categorized into different levels based on their severity. The discovery process of each kind of vulnerabilities is different from the other. Hence, it is essential to incorporate the severity of the vulnerabilities during the modelling of the vulnerability discovery process. This paper proposes a model to assess the vulnerabilities present in the software quantitatively with consideration for the severity of the vulnerabilities. It is possible to apply the proposed model to approximate the number of vulnerabilities along with vulnerability discovery rate, future occurrence of vulnerabilities, risk analysis, etc. Vulnerability data obtained from one of the major web browsers (Google Chrome) is deployed to examine goodness-of-fit and predictive capability of the proposed model. Experimental results justify the fact that the model proposed herein can estimate the required information better than the existing VDMs.**

***Index Terms*—Vulnerabilities, Vulnerability discovery model, Severity, Web browser**

I. Introduction

The recent progresses achieved in information and communication technology have resulted in data security becoming a serious issue for both academia and the industry. Nowadays, web applications have been extensively used in many domains including banking, healthcare, transportation and social networking. A critical vulnerability is capable of giving an attacker full access to the web applications, which could lead to compromising the security properties (confidentiality, integrity and availability) of the applications. Therefore, steps should be taken to ascertain the fact that the security properties of the developed web applications are not breached.

Security vulnerability is caused due to the potential exploitation of software code or system in the form of unauthorized access. This malicious behavior or fraudulent access may be reflected as introduction of viruses, Trojan horses, malware, etc. to the original code. Pfleeger [1] defines vulnerability as a flaw that is inherent in an application which may be abused by an attacker. According to NVD [2], more than 18000 and 14600 vulnerabilities were reported in 2018 and 2017 respectively, which are more than twice the amount of the vulnerabilities reported in 2016, i.e., 6,447. This issue is of great concern to the cybersecurity community and should be addressed as attackers try to exploit these vulnerabilities causing compromise upon the availability, confidentiality or system integrity. Also, security vulnerabilities research received a lot of attention in the last decade, and VDMs have been playing an important role in the study of the trend of vulnerability discovery in the software. Furthermore, VDMs help to determine the readiness of release, allocation of resource for future patch release, followed by assessment of the risk of code vulnerability owing to human exploitation.

Characterization of security vulnerabilities and predicting the unknown future vulnerabilities have been studied for several years. Many authors have considered the importance of characterization of security vulnerability and their exploitation in assessing risk. Some of the authors have made efforts to study the behavior of vulnerability discovery process by using different modelling techniques. The first VDM was developed by Anderson [3], but the model had significant drawbacks. Alhazmi and Malaiya [4] presented a time and effort-based model to examine quantitatively the content of vulnerabilities in a couple of operating systems using the database which accounts for the reported vulnerabilities. Whilst the time-based model involves understanding the correlation between cumulative vulnerabilities and calendar time; the effort-based model uses equivalent effort, which signifies the effort that goes into finding vulnerabilities as a metric. In addition, they introduced another metric referred to as known vulnerability density, which compared the values against the systems and assesses their maturity corresponding to security vulnerabilities. The vulnerability data reported against the varying versions of the main operating systems (Windows and Red Hat), and for IIS HTTP ServersApache and Apache were tested using Alhazmi and Malaiya model [5], [6].

Rescorla [7] examined the vulnerability discovery rate from empirical data. The process involves associating a reliability growth model on rate of vulnerability discovery and using that to uncover the trend of total number of vulnerabilities. Woo et al. [8] deployed Alhazmi Malaiya Logistic Model [4] to characterize the vulnerabilities of the major browsers, namely: Mozilla, Internet Explorer, and Firefox; and examined the vulnerability discovery shifts. They also applied the vulnerability severity levels to evaluate the utility of the Logistic model. Shar and Tan [9] developed defect predictors against a static code attributes set which depicts input sensitization and validates patterns in code. Hovsepyan et al. [10] exploited raw source code analysis as text for vulnerability prediction. Massacci and Nguyen [11] built an empirical methodology which measured VDMs' performance using quality and predictability as the quantitative metrics. Shar et al. [12] utilized the benefits offered by the current static and dynamic taint analysis methods (which they referred to as hybrid analysis) and proceeded to use prediction models developed using machine learning techniques for web application vulnerability prediction. Jeffrey et al. [13] also studied the effect of the reduction in dimensionality in vulnerability prediction of software models.

Numerous studies have opined that the vulnerability discovery process is similar to the fault recognition process during the testing period of software. Therefore, software reliability growth models are applied in characterizing the vulnerability discovery process in these studies. The authors in [14] proposed a logarithmic Poisson model based on the Musa-Okumoto [15] logarithmic execution time model to ascertain the number of vulnerabilities. Kimura [16] also developed a software vulnerability assessment model based on the non-homogeneous Poisson process to analyze the software vulnerabilities present in the sendmail system. Several authors have also made efforts in vulnerability prediction [11], [17]–[28].

Web browsers are the most favored target of the attackers. Websites use various components from the multiple sources and many of these sources are not authorized by the site owners. Attackers make these websites easy target by distributing the malware through the websites without risking detection. The increasing number of vulnerabilities detected in various web browsers like Google Chrome, Microsoft ChakraCore, Mozilla Firefox, Internet Explorer, Apple Safari etc. is a major concern of the cybersecurity industries and researchers. According to a report published in 2019 [29], vulnerabilities present in the browsers are still on the rise. For web browsers, vulnerabilities reported in 2018 are 20 percent higher than the vulnerabilities reported in 2017. However, Microsoft Edge and Apple Safari are exceptions. As published in this report, Google Chrome and Microsoft Edge are the most favored web browsers, while Microsoft Chakra Core and Apple Safari are less popular among attackers. Therefore, there is a need to develop a systematic methodology to study the behavior of vulnerability discovery process of web browsers. In this paper, a method to model vulnerability in the discovery process with consideration for the vulnerability severity has been proposed. The three categories of vulnerability severity that are use for the study include: low, medium and high. The model proposed has been validated using the vulnerabilities data reported for Google Chrome web browser.

The remaining paper is structured as follows: The detailed description of vulnerability severity and systematic development of model proposed is presented in Section II. The description of vulnerabilities data for Google Chrome and various comparison benchmarks employed to measure the performance of the proposed model are given in Section III. In Section IV, the performance analysis of the proposed model is done and Section V concludes the paper.

II. Model Development

This section elaborates on the severity of vulnerabilities and detailed development of the proposed model are discussed.

A. Severity of Security Vulnerability

Security vulnerabilities can be indicated by a vulnerability value which denotes the severity of risk or loss because of the vulnerability. For example, password file and Microsoft Word are used to store the information on the computer systems, but the vulnerability related to the password file typically has high severity due to importance of password. The severity in vulnerability depends on numerous factors ranging from impact on the integrity, confidentiality or availability of data, along with which particular attack vector is used, the complexity of the attack, the required privileges, or any other interaction with the user. National Vulnerability Database (NVD) [2] assigns qualitative vulnerability severity rating which helps responders to prioritize responses and resources according to threat. NVD use Common Vulnerability Scoring System (CVSS) to assign score to vulnerability. National Infrastructure Advisory Council (NIAC), introduced first version of CVSS in February 2005 with the goal being designed to provide open and universal standard severity ratings of software vulnerabilities. The current version of CVSS, i.e., CVSS 3.0 was launched in June 2015. NVD use two version of CVSS (CVSS v2.0 and CVSS v3.0) standards. It assigns qualitative severity rankings based on the base score range of the vulnerabilities. Severity ranking with base score range is given in Table I for two versions of CVSS.

TABLE I
CVSS Severity Ratings [2]

CVSS v2.0 Ratings		CVSS v3.0 Ratings	
Severity	Base Score Range	Severity	Base Score Range
		None	0.0
Low	0.0-3.9	Low	0.1-3.9
Medium	4.0-6.9	Medium	4.0-6.9
High	7.0-10.0	High	7.0-8.9
		Critical	9.0-10.0

B. Related Models

In the past, some authors have developed VDMs to characterise the vulnerability discovery rate. Some of the most relevant contributions are as follows.

Alhazmi and Malaiya [4] developed a time-based model which is based on the fact that the change in the rate of cumulative number of vulnerabilities depends on two factors and it is formulated as follows:

$$\frac{dV}{dt} = aV(b-V), \tag{1}$$

where V refers to the cumulative number of vulnerabilities, a and b are considered to be the empirical constant which depend on available data. Solving the above equation, the total fault content function can be represented as follows:

$$V = \frac{b}{bc\exp(-abt)+1} \tag{2}$$

Rescorla [7] examined the trend of the vulnerability discovery process by conducting some statistical tests. In first test, he utilized the vulnerability discovery rate as a linear function of time as follows:

$$v(t) = bt + k \tag{3}$$

where b and k are the constants. He suggested to fit the above linear curve to the curve of vulnerability discovery and then, the total number of vulnerabilities may be find easily by integrating it as follows:

$$V(t) = \frac{bt^2}{2} + kt \tag{4}$$

Similarly, he [7] considered the vulnerability discovery rate as exponential function of time in his second test that was based on model framed by Goel-Okumoto (G-O) [30]. He represented the rate of vulnerability discovery as

$$v(t) = N\lambda e^{-\lambda t}, \tag{5}$$

where N represents the overall number of vulnerabilities and λ represents the discovery rate constant. Cumulative number of vulnerabilities can be obtained by integrating the above equation

$$V(t) = N(1-e^{-\lambda t}), \tag{6}$$

where $V(t)$ is the cumulative number of vulnerabilities and $V(0) = 0$. $v(t)$ is also expressed as $v(t) = \frac{dV(t)}{dt}$.

Alhazmi and Malaiya [14] developed a logarithmic Poisson model based on the Musa-Okumoto [15] logarithmic execution time model to estimate the number of vulnerabilities which is as follows:

$$V(t) = b_0 \ln(1 + b_1 t), \tag{7}$$

where b_0 and b_1 are the constants and $V(0) = 0$.

C. Proposed Model: Vulnerability Discovery Model with Vulnerability Severity

1) Formulation of the Proposed Model: In the proposed model, vulnerabilities discovery process is assume to follow a non-homogeneous Poisson process similar to the model developed by Goel-Okumoto [30] and it is assume that the rate of vulnerability discovery process is proportional to the number of undiscovered vulnerabilities at time t. The vulnerability discovery rate is represented as follows:

$$\frac{dV(t)}{dt} = \lambda\left(N - V(t)\right), \tag{8}$$

where $V(t)$ is the cumulative number of vulnerabilities discovered, N is the overall number of product's vulnerabilities and λ is a discovery rate constant. Solving the equation (8), with $V(0) = 0$ we have

$$V(t) = N(1-e^{-\lambda t}), \tag{9}$$

As discussed in the previous section, a product may have vulnerability with different severity. In the past, numerous VDMs have been recommended to study the vulnerability discovery process, however no effort has been made to incorporate the severity of vulnerabilities in modelling the vulnerability discovery process. Considering this fact, a VDM is proposed in this section which incorporated the vulnerability severity. In this model, vulnerability severity is categorized in three parts similar to the CVSS v2.0 standards, as follows:

1) Low
2) Medium, and
3) High.

It is assume that vulnerability discovery rate of a category of vulnerabilities is proportional to the number of undiscovered vulnerabilities of that category at time t. For example, the rate of discovery of vulnerabilities with low severity is proportional undiscovered vulnerabilities of the product with low severity at time t. Therefore, the vulnerability discovery rate of i^{th} type of vulnerabilities is given by the differential equation, as follows

$$\frac{dV_i(t)}{dt} = \lambda_i\{Np_i - V_i(t)\} \tag{10}$$

where $V_i(t)$ is the cumulative number of i^{th} type vulnerabilities discovered, N is the total number of product's vulnerabilities to be eventually reported, λ_i is a constant discovery rate of i^{th} type of vulnerabilities, and p_i is the proportion of i^{th} type of vulnerabilities. where $i = 1,\ 2,\ 3$. The above equations satisfy the initial condition $V_i(0) = 0$.

2) Solution of the Proposed Model: Cumulative number of vulnerabilities of each category of the vulnerabilities can be represented by the equation obtained by solving the Eqn. (10) with $V_i(0) = 0$, as follows:

1) Vulnerabilities with low severity (for $i = 1$)
Cumulative number of vulnerabilities of this category of the vulnerabilities can be represented as

$$V_1(t) = Np_1(1-e^{-\lambda_1 t}), \tag{11}$$

2) Vulnerabilities with medium severity (for $i = 2$)
Cumulative number of vulnerabilities of this category of the vulnerabilities can be represented as

$$V_2(t) = Np_2(1-e^{-\lambda_2 t}), \tag{12}$$

3) Vulnerabilities with high severity (for $i = 3$)
Cumulative number of vulnerabilities of this category of the vulnerabilities can be represented as

$$V_3(t) = Np_3(1-e^{-\lambda_3 t}), \tag{13}$$

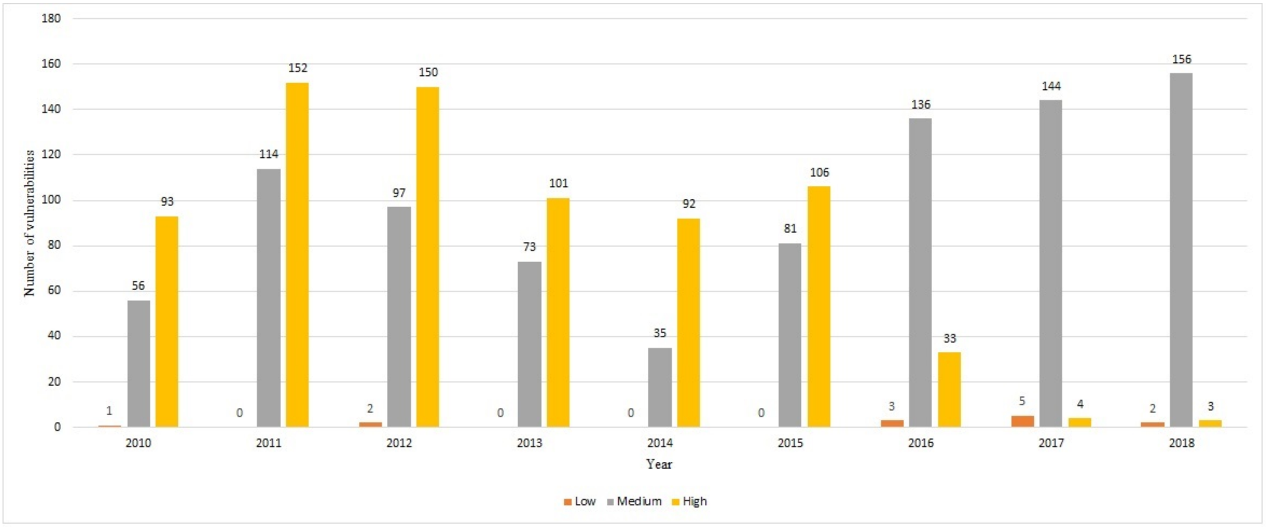

Fig. 1. Number of vulnerabilities with different severity reported during January 2010 to December 2018.

3) Mean Value Function of the Proposed SRGM: Since the reported cumulative number of vulnerabilities in the product is the sum of vulnerabilities with all type of severity. Therefore, mathematically it can be represented as

$$V(t) = \sum_{i} V_i(t), \tag{14}$$

where $i = 1,\ 2,\ 3.$

Proposition 1: As mentioned in the above equations, $p_1,\ p_2$, and p_3 are the proportionality of the vulnerabilities with low, medium and high vulnerabilities and N is the expected number of vulnerabilities to be reported. Then $N = N(p_1 + p_2 + p_3)$ where $p_1 + p_2 + p_3 = 1$ or $p_3 = 1 - p_1 - p_2$.

This proposition will also be helpful in parameter estimation of the proposed model.

III. Data Set and Comparison Criteria

In this section, a detailed description of the data set and different criteria used for comparing the proposed model and existing related models are presented.

A. Data Set

In this paper, vulnerability data set of one of the most popular browser (Google Chrome) is used. This data set was obtained from the CVE Details [31] which is a Common Vulnerabilities and Exposures (CVE) security vulnerability database/information source. This data set has been collected between January 2010 to June 2019. During the period of 114 months, total 1812 vulnerabilities are reported in Google Chrome. In the reported vulnerabilities, total 18 vulnerabilities are reported with low severity, 1056 vulnerabilities are reported with medium severity and 738 vulnerabilities are reported with high severity. The graphical representation of year wise vulnerabilities reported with different severity is shown in Fig. 1.

B. Comparison Criteria

The following set of measures are considered in comparing the proposed model with some related models:

1) Mean Square Error (MSE)
Let $V(t_i);\ i = 1, 2, ...n$ and V_i are the estimated and actual number of vulnerabilities respectively, then MSE can be measured as

$$\text{MSE} = \frac{1}{(n-k)} \sum_{i=1}^{n} (V(t_i) - V_i)^2, \tag{15}$$

where the total number of observations is denoted by n, and the number of unknown parameters is denoted by k.

2) Bias
The term Bias is explained as the sum of the deviations between estimated vulnerabilities and actual vulnerabilities given as

$$\text{Bias} = \frac{1}{n} \sum_{i=1}^{n} (V(t_i) - V_i) \tag{16}$$

3) Variation
Variation is the measure of estimation error, and it can be defined as follows:

$$\text{Variation} = \sqrt{\frac{1}{n-1} \sum_{i=1}^{n} (V(t_i) - V_i - \text{Bias})^2}. \tag{17}$$

4) R-Square (R^2)
R^2 is the ratio of the sum of square (R^2) derived from the trend VDM to the actual vulnerabilities. It can be expressed as follow:

$$R^2 = \frac{residualSS}{correctedSS} \tag{18}$$

5) Predictive Sum of Square Error (PSSE)

TABLE II
ESTIMATED PARAMETERS OF THE VDMS.

Model	Estimated parameters						
	a	b	c				
Alhazmi-Malaiya Logistic Model [4]	0.00004776	1317.007	0.008				
	N	λ					
Rescorla Exponential Model [7]	4189.074	0.005					
	b_0	b_1					
Musa-Okumoto Model [15]	3645.623	0.005					
	N	p_1	p_2	p_3	λ_1	λ_2	λ_3
Proposed Model	3856.734	0.231	0.209	0.560	0.003	0.003	0.007

TABLE III
COMPARISON BETWEEN EXPONENTIAL VDM WITH CHANGE POINT AND WITHOUT CHANGE POINT.

Model	Comparison Criteria				
	MSE	|Bias|	Variation	R^2	PSSE
Alhazmi-Malaiya Logistic Model	3.7517e+03	18.1063	57.7214	0.9395	2.0287e+06
Rescorla Exponential Model	6.4849e+03	71.4549	35.2525	0.9748	4.3097e+05
Musa-Okumoto Model	2.3908e+03	23.0402	63.0391	0.9915	1.2954e+05
Proposed Model	1.6606e+03	24.9062	30.5834	0.9950	6.0817e+04

PSSE is defined as:

$$\text{PSSE} = \sum_{i=1}^{n} \left(V_i - \hat{V}_i\right)^2 \tag{19}$$

where V_i denotes the observed vulnerabilities, and $\hat{V}_i$ denotes the predicted vulnerabilities.

IV. PERFORMANCE ANALYSIS

In this section, details of the performance analysis for the proposed model with respect to related models which were discussed in Section II.C are described. For this, two criteria: goodness of fit to the data set and its predictive capability are used. 75% of the data is used to test the goodness of fit and the remaining data is used to examine the prediction capability of the proposed model. The parameters value of the VDMs are obtained using least square estimation technique with the help of non-linear regression module of "SPSS" software [32]. Estimated parameters of VDMs are given in Table II. Comparison criteria, MSE , Bias, Variation and R^2 are used to test the goodness of fit, and PSSE is used to test the prediction capability.

After estimation of parameters, the proposed model and other VDMs are fitted to the vulnerability data reported in Google Chrome and hence the values of different comparison criteria are calculated. As given in Table III, the proposed model gives the best MSE and R^2 value as the value of MSE for the proposed model is lower, and the value of R^2 is higher in comparison to the existing models. However, Alhazmi-Malaiya Model gives a better bias value in comparison to the proposed model but the proposed model give the best value of the other comparison criteria. Fig. 2 shows the graphical representation of fitted plots and the actual curve of vulnerabilities reported in Google Chrome. As shown in this figure, fitted plot and actual vulnerabilities are very closed. The comparison of vulnerabilities determined by the proposed model and existing VDMs is shown graphically in Fig. 3. As

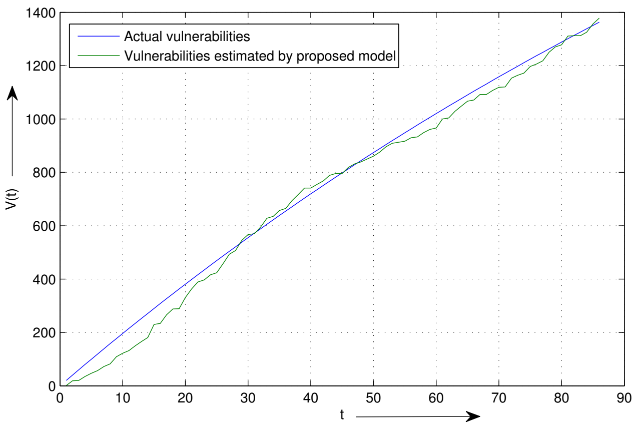

Fig. 2. Actual vulnerabilities and vulnerabilities estimated by the proposed model.

shown in this figure, vulnerabilities obtained by the proposed model are relatively closer to the reported vulnerabilities as compared to the existing VDMs. From these results, it is clear that the proposed model fits best in score to the vulnerability of the data set. Moreover, PSSE estimated by the proposed model is also lower in comparison to the existing VDMs. This implies that the prediction capability of our proposed model is also superior.

Hence, the above results show that our proposed model performs better for both goodness of fit and forecasting.

V. CONCLUSION

In this paper, a VDM is presented to assess the total number of vulnerabilities present in the web browser quantitatively considering vulnerability severity, that can be employed in estimating the number of vulnerabilities, discovery rate of vulnerabilities, future occurrence of vulnerabilities, risk analysis, etc. Vulnerabilities can be categorized into different types based on

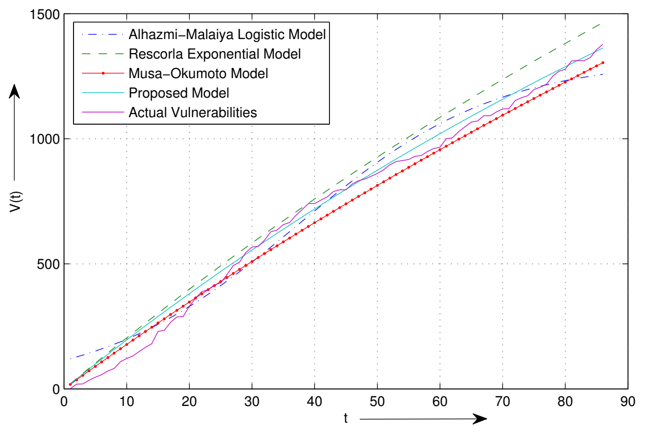

Fig. 3. Estimated vulnerabilities by the proposed model and existing models

severity. We categorized the vulnerabilities based on severity into three categories, namely: low, medium and high and these categories are also considered in the model development. To measure the performance of our model, vulnerability data from the most popular web browser, that is Google Chrome was used. Experimental outcomes demonstrate that our model performs better than related models. Therefore, our model can be helpful for academia and the industry with regards to risk assessment and decision-making when allocating resources for patches releases, etc.

Acknowledgment

This work was carried out during the tenure of an ERCIM Alain Bensoussan Fellowship Programme.

References

[1] C. P. Pfleeger and S. L. Pfleeger, *Security in computing*. Prentice Hall Professional Technical Reference, 2002.

[2] Nvd.Nist.Gov, "National vulnerability database," May 2019. [Online]. Available: https://nvd.nist.gov/vuln/full-listing

[3] R. Anderson, "Security in open versus closed systems-the dance of boltzmann," in *Coase and Moore Conference on Open Source Software Economics, Toulouse, France*, 2002.

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

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

[6] O. H. Alhazmi and Y. K. Malaiya, "Measuring and enhancing prediction capabilities of vulnerability discovery models for apache and iis http servers," in *2006 17th International Symposium on Software Reliability Engineering*. IEEE, 2006, pp. 343–352.

[7] E. Rescorla, "Is finding security holes a good idea?" *IEEE Security & Privacy*, vol. 3, no. 1, pp. 14–19, 2005.

[8] S.-W. Woo, O. H. Alhazmi, and Y. K. Malaiya, "An analysis of the vulnerability discovery process in web browsers," *Proc. of 10th IASTED SEA*, vol. 6, pp. 13–15, 2006.

[9] L. K. Shar and H. B. K. Tan, "Predicting common web application vulnerabilities from input validation and sanitization code patterns," in *2012 Proceedings of the 27th IEEE/ACM International Conference on Automated Software Engineering*. IEEE, 2012, pp. 310–313.

[10] A. Hovsepyan, R. Scandariato, W. Joosen, and J. Walden, "Software vulnerability prediction using text analysis techniques," in *Proceedings of the 4th international workshop on Security measurements and metrics*. ACM, 2012, pp. 7–10.

[11] F. Massacci and V. H. Nguyen, "An empirical methodology to evaluate vulnerability discovery models," *IEEE Transactions on Software Engineering*, vol. 40, no. 12, pp. 1147–1162, 2014.

[12] L. K. Shar, L. C. Briand, and H. B. K. Tan, "Web application vulnerability prediction using hybrid program analysis and machine learning," *IEEE Transactions on dependable and secure computing*, vol. 12, no. 6, pp. 688–707, 2014.

[13] J. Stuckman, J. Walden, and R. Scandariato, "The effect of dimensionality reduction on software vulnerability prediction models," *IEEE Transactions on Reliability*, vol. 66, no. 1, pp. 17–37, 2016.

[14] O. H. Alhazmi and Y. K. Malaiya, "Application of vulnerability discovery models to major operating systems," *IEEE Transactions on Reliability*, vol. 57, no. 1, pp. 14–22, 2008.

[15] J. D. Musa and K. Okumoto, "A logarithmic poisson execution time model for software reliability measurement," in *Proceedings of the 7th international conference on Software engineering*. Citeseer, 1984, pp. 230–238.

[16] M. Kimura, "Software vulnerability: definition, modelling, and practical evaluation for e-mail transfer software," *International journal of pressure vessels and piping*, vol. 83, no. 4, pp. 256–261, 2006.

[17] L. K. Shar and H. B. K. Tan, "Predicting sql injection and cross site scripting vulnerabilities through mining input sanitization patterns," *Information and Software Technology*, vol. 55, no. 10, pp. 1767–1780, 2013.

[18] H. Joh and Y. K. Malaiya, "Modeling skewness in vulnerability discovery models in major operating systems," *Red*, vol. 2, no. 5, p. 0, 2010.

[19] S. Rahimi and M. Zargham, "Vulnerability scrying method for software vulnerability discovery prediction without a vulnerability database," *IEEE Transactions on Reliability*, vol. 62, no. 2, pp. 395–407, 2013.

[20] J. Yang, D. Ryu, and J. Baik, "Improving vulnerability prediction accuracy with secure coding standard violation measures," in *2016 International Conference on Big Data and Smart Computing (BigComp)*. IEEE, 2016, pp. 115–122.

[21] M. K. Gupta, M. C. Govil, and G. Singh, "Predicting cross-site scripting (xss) security vulnerabilities in web applications," in *2015 12th International Joint Conference on Computer Science and Software Engineering (JCSSE)*. IEEE, 2015, pp. 162–167.

[22] D. Last, "Using historical software vulnerability data to forecast future vulnerabilities," in *2015 Resilience Week (RWS)*. IEEE, 2015, pp. 1–7.

[23] A. Shrivastava, R. Sharma, and P. Kapur, "Vulnerability discovery model for a software system using stochastic differential equation," in *2015 International Conference on Futuristic Trends on Computational Analysis and Knowledge Management (ABLAZE)*. IEEE, 2015, pp. 199–205.

[24] H. Alves, B. Fonseca, and N. Antunes, "Experimenting machine learning techniques to predict vulnerabilities," in *2016 Seventh Latin-American Symposium on Dependable Computing (LADC)*. IEEE, 2016, pp. 151–156.

[25] M. Jimenez, M. Papadakis, and Y. Le Traon, "Vulnerability prediction models: A case study on the linux kernel," in *2016 IEEE 16th International Working Conference on Source Code Analysis and Manipulation (SCAM)*. IEEE, 2016, pp. 1–10.

[26] H. Joh and Y. K. Malaiya, "Periodicity in software vulnerability discovery, patching and exploitation," *International Journal of Information Security*, vol. 16, no. 6, pp. 673–690, 2017.

[27] K. Z. Sultana, "Towards a software vulnerability prediction model using traceable code patterns and software metrics," in *Proceedings of the 32nd IEEE/ACM International Conference on Automated Software Engineering*. IEEE Press, 2017, pp. 1022–1025.

[28] X. Wang, R. Ma, B. Li, D. Tian, and X. Wang, "E-wbm: An effort-based vulnerability discovery model," *IEEE Access*, vol. 7, pp. 44 276–44 292, 2019.

[29] Lp.Skyboxsecurity.Com, "Skyboxsecurity," May 2019. [Online]. Available: https://lp.skyboxsecurity.com/rs/440-MPQ-510/images

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

[31] Cvedetails.Com, "Cve details," May 2019. [Online]. Available: https://www.cvedetails.com/product/15031

[32] Ibm.Com, "Ibm spss software," May 2019. [Online]. Available: https://www.ibm.com/analytics/spss-statistics-software