ISSN 2278-3091
Volume 9, No.4, July – August 2020
International Journal of Advanced Trends in Computer Science and Engineering
Available Online at http://www.warse.org/IJATCSE/static/pdf/file/ijatcse82942020.pdf
https://doi.org/10.30534/ijatcse/2020/82942020

An Improved Prediction Model for Software Vulnerabilities Count using Regression Approach

Mohammad Shamsul Hoque[1,*], Norziana Binti Jamil[2], Nowshad Amin[3]
[1,2]College of Computing and Informatics, Universiti Tenaga Nasional, Selangor
shams_uttara@gmail.com, Norziana@uniten.edu.my
[3]Institute of Sustainable Energy, Universiti Tenaga Nasional, Selangor

ABSTRACT

Successful cyber-attacks are majorly contributed through the exploitation of some of vulnerabilities in software, hardware, system and/or network. Vulnerability forecasting models help us to predict which vulnerabilities that will be highly exploited for future attacks. This study aims at further enhancing the results of recent research findings using intuitive sampling techniques and features selection to develop a regression-based model for predicting highly-exploitable vulnerabilities, by using open source dataset. Vulnerability forecasting models help us to predict the number of vulnerabilities that may occur in the future. Unlike using time series analysis, we propose a vulnerability analytic prediction model based on reframing a time series forecasting problem as a supervised learning problem. Our study has developed a predictive analytic model for three top most vendors having highest number of vulnerabilities published in the National Vulnerability Database (NVD) , namely Microsoft, IBM and Oracle. We have intuitively created feature dataset on 07 days interval for vulnerability count and 04 new features with aggregation.This proposed supervised model producesmore accurate prediction results with RMSE as low as 1.47 and outperforms previous models.

Key words: SupervisedMachine Learning, Information Security, Vulnerability Prediction Model, RapidMiner.

1. INTRODUCTION

Vulnerability shows the weakness of the system and because of this, most of the security breaches are occurred. The reality is that while data volume is growing [1],the vulnerability trend is also growing and accompanying with threat for security breach [2]. In most of the successful cyber-attacks,usually hackers would exploit zero-days vulnerabilities before the security administrator is aware of its existence [3], [4].Therefore, the importance for security focus has dramatically increased and past vulnerability trends and reviews become more valuable to consider in acquisition of new software system [5]. Machine learning is the subfield of computer science and an application of artificial intelligence (AI) that "gives computers the ability to learn without being explicitly programmed" [6]. Data Modelling for predicting vulnerabilities and thereby creating deployable machine learning models is a highly possible tool that can be used as a proxy to predict for the likelihood of highly-exploitable vulnerability. The authors in [7] and [8] showed that it is difficult to address security during software development phase as project managers usually focus on cost and timely delivery of products and therefore vulnerabilities will most probably always exist.

Although research in vulnerability prediction is a popular topic for years but it achieved stronger traction in industry recently [9]. Researchers have used vulnerability databases of various forms to develop models for vulnerabilities disclosure trends. Most of these researchesaimed to find techniques to develop models that can predict the number of vulnerabilities of products in future by using their historical data through regression and time series analysis [10]-[19]. These models suffer in various extents from limitations stemmed from their underlying assumptions, as in [12]. These modelsalso have limitations in various extents to deliver high accuracy to follow trend and seasonality. Since security activities for software and systems are highly resource savvy it is expected that the models can predict vulnerabilities with high accuracy. This yields the need to explore additional models for predicting number of future vulnerabilities. Since software security involves a good amount of resources, a good prediction on the frequency of disclosures for vulnerabilities is deemed expected by the vendors, end users and also businesses such as insurance companies [20]. In this paper, we take the issue of predicting number of future vulnerabilities as a supervised linear and non-linear problem and our work makes the following contributions:

• We introduce an intuitive approach to create a dataset that containsvulnerabilities obtainedfrom the open source NVD database based on 07 days (per week) time interval used for prediction modelling. We choose 07 days in pursuit of creating a balanced dataset with more rows. To the best of our knowledge, this dataset having 07 days' time interval is the first such feature dataset;
• we intuitively created 04 new features from the attributes of dumped database as inputs for predicting future number of vulnerabilities. We compare the capability of our models in predicting the total number of software vulnerabilities in 07-day time intervals for the top 3 vendors as specified in [21],between 1997-2019;
• We show that our models outperforms the most of the

existing models in terms of prediction accuracy and following trends.

The rest of the paper is organized as follows. Section 2 describes the related work. Section 3 describes the dataset and methodology used in the analysis. Section 4 describes the result and analysis andfinally Section 5 describes the conclusions, limitations and future works.

2. RELATED RESEARCH

Although Hsugk [22] developed some testing results concerning electrical switching system involving software, Hudson [22] was the first to describe software failure processes by Markov birth-death process in which introduction and removal of faults were considering a birth and death respectively. A survey on software defect finding processes involving software reliability growth models was done by Lyu and Lyu [22]. Anderson Thermodynamic (AT) was the first time-based vulnerability discovery model proposed [23]. Alhazmi and Malaiyaproposed a time based application of SRGMs to model prediction of vulnerability numbers and later also proposed another logistic regression model (which was termed as AML) that examines its capability on Windows 98 and NT 4.0 to predict the number of undiscovered vulnerabilities [13], [14]. Two trend models using the non-cumulative vulnerability rate data were examined by Rescolaand proposed two time based trend models, a linear model (RL) and an exponential model (RE) to estimate future vulnerabilities[12].Kim proposed a new Weibull distribution based VDM and made comparison with AML model and found their model performed better in many cases [24]. There are also several other studies that worked further based on existing VDMs for various software packages with aim to improve vulnerability discovery rate and prediction for future vulnerability count [25]-[28].

In recent times Movahedi et al. [17] compared the capabilities for predicting the number of future vulnerabilities of nine common vulnerability discovery models (VDMs) with a nonlinear neural network model (NNM) over a prediction period of three years. The common VDMs are NHPP Power-law [17], Gamma-based VDM [29], Weibull-based VDM [23], AML VDM [14], [15], Normal-based VDM[29], Rescorla Exponential (RE) [12], Rescorla Quadratic (RQ) [30], Younis Folded (YF) [31], Linear Model (LM) [32]. They have used NVD dataset with the feedforward NNM with a single hidden-layer NNM for one step ahead forecasting for four well known OSs and four well-known web browsers and assessed the models were assessed by prediction accuracy and prediction bias. The results showed that the NNM, in terms of prediction accuracy, outperformed the VDMs in all the cases and regarding overall magnitude of bias NMM provided the smallest value in against seven common VDMs out of the eight. As for example in case windows, average error (AE) between NMM and best performed common VDM (Gamma) are 0.026 and 0.063 respectively.

Pokhrel proposed vulnerability prediction model based on non-linear approach through time series analysis[18]. They have utilized NVD database for Auto Regressive Moving Average (ARIMA), Artificial Neural Network (ANN), and Support Vector Machine (SVM) settings to develop the prediction model. Three mostly used Operating Systems, such Windows 7, Mac OS X, and Linux Kernel were considered for the experiments. As for a result example, the best model for windows 7 was produced by SVM with SMAPE, MAE and RMSE values of 0.12, 3.15 and 3.58 respectively. ANN is more usable with big data [36], [37] which was not in this case with small data set.

However, since nonlinearity is a common trend in vulnerability disclosure traditional time series based modelling may always have limited capability to get high accuracy in prediction [33]. In our study the number of vulnerability prediction is modelled with intuitively generated features for supervised regression and the prediction accuracy was found better comparing with others such as recent one by [18].

3. METHODOLOGY

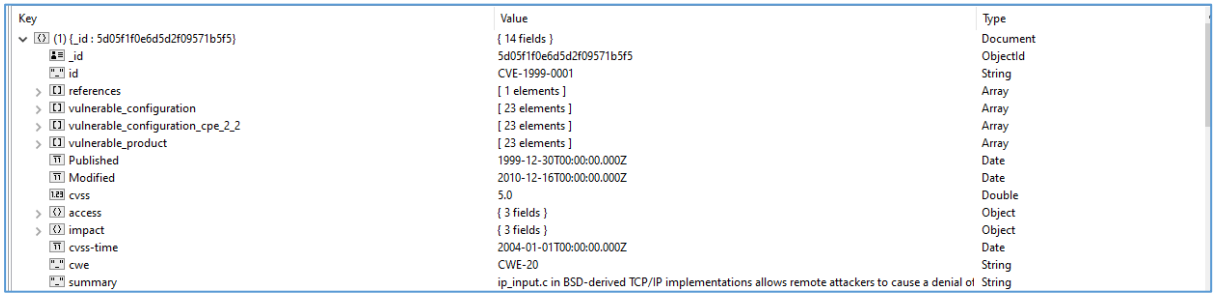
We have extracted the vulnerability data from the National Vulnerability Database [34] and dumped in a local mongodb through the process described in the cve-search tool [35], from 1995 to November 2019. We then created a web scrapper to dump vulnerability data for top 3 vendors form the MITRE's CVE web site [19] and imported in the mongodb for the years from 1997 to 2019. The number of vulnerabilities for Microsoft, Oracle and IBM were 6814, 6115 and 4679 respectively (see Figure 1, snipped from [19]).

Top 50 Vendors By Total Number Of "Distinct" Vulnerabilities

Go to year: 1999 2000 2001 2002 2003 2004 2005 2006 2007 2008 2009 2010 2011 2012 2013 2014 2015 2016 2017 2018 2019 All Time Leaders

	Vendor Name	Number of Products	Number of Vulnerabilities	#Vulnerabilities/#Products
1	Microsoft	529	6814	13
2	Oracle	644	6115	9
3	IBM	1064	4679	4

Figure 1: Top 3 Vendors by Vulnerability Count.



Key	Value	Type
(1) {_id : 5d05f1f0e6d5d2f09571b5f5}	{ 14 fields }	Document
_id	5d05f1f0e6d5d2f09571b5f5	ObjectId
id	CVE-1999-0001	String
references	[1 elements]	Array
vulnerable_configuration	[23 elements]	Array
vulnerable_configuration_cpe_2_2	[23 elements]	Array
vulnerable_product	[23 elements]	Array
Published	1999-12-30T00:00:00.000Z	Date
Modified	2010-12-16T00:00:00.000Z	Date
cvss	5.0	Double
access	{ 3 fields }	Object
impact	{ 3 fields }	Object
cvss-time	2004-01-01T00:00:00.000Z	Date
cwe	CWE-20	String
summary	ip_input.c in BSD-derived TCP/IP implementations allows remote attackers to cause a denial of	String

Figure .2: Tree view of collection "cves", snipped for Studio 3T view

The most important collection in the dumped database is 'cves' and it has attributes with nested documents namely id (CVE_ID), references, vulnerable_conFigureuration, vulnerable_conFigureuration_cpe_2_2, vulnerable_product, Published, Modifies, cvss, access, impact, cvss-time, cew, summay (see Figure 2).

We then intuitively created 4 new attributes in the machine learning modelling for each vulnerability row (CVE_ID/id):

- "conFigure" which is the size of the nested documents in the vulnerable_conFigureuration_cpe_2_2 field.
- "Days_diff" which is the difference between Modified and Published date and converted into number of days.
- "Ref" which is size of nested documents in the references filed.
- "week" which is the difference between the first published date of CVE_ID for each vendor in the database and the respective Published dates of other CVE_IDs and converted the time in days first and into weeks finally. Therefore we can get the number of weeks count since the first entry in the NVD dataset for each CVE_ID or vulnerability entry.

We then imported the scrapped tables for Microsoft, Oracle and IBM as collection in the mongodb in Studio 3T and created new tables for each one by inner join based on CVE_ID field (see Figure 3). Later we created groups on the ''week" field where id field (CVE_ID) was counted and all other fields were summedin the operation by the following code in python (see Figure 4). Thereby we now get count of vulnerabilities (CVE_ID) per respective week in order and we then changed the weeks in a number series starting from 1.

```
15 data=df.groupby(
16     'week', as_index=False
17 ).agg(
18     {
19           'Ref':sum,
20           'cvss': sum,
21           'days_diff': sum,
22           'config':sum,
23           'cve_id': "count"
24
25     }
26 )
```

Figure 3: Code for grouping

```
In [24]: print (data.head())
   week  Ref   cvss  days_diff  config  cve_id
0     0    2  128.0      87418      36      18
1     5    1    5.0       7857       3       1
2     8    2    5.1       8421       5       2
3    12    0   12.5       8358       4       2
4    13    3    5.0       7138       1       1

In [25]:
```

Figure.4: Dataset for ML model (for Microsoft)

We then investigated the relation and trend of the dependent variable,"cve_id", against four generated independent variables by line diagrams (see Figure. 5-10). Each vendor is shown by two line charts (divided at somewhere in middle of total numberof weeks) in zoomed for clear views. It can be observed that the number of vulnerabilities is low and stable at the beginning of time (weeks) but shows spikes and lows as number of weeks grows. Generally the number of vulnerabilities has an increasing trend in course of time but does not show any reliable trend or seasonality which also shows the requirement of modelling with linear and non-linear algorithms. As far as generated features is concerned, we have found that features such as "Ref", "cvss" and "conFigure" but "days_diff" generally follows the trend and spikes and falls with the "cve_id" count with respect to weeks in order. More specifically generated feature "cvss (sum of CVE scores by respective weeks)"follows these characteristics more followed by other features "Ref" and "conFigure". Although feature "days_diff" does not follow the trend well like others we still utilized it in our model to find out its correlation with other features and level data.

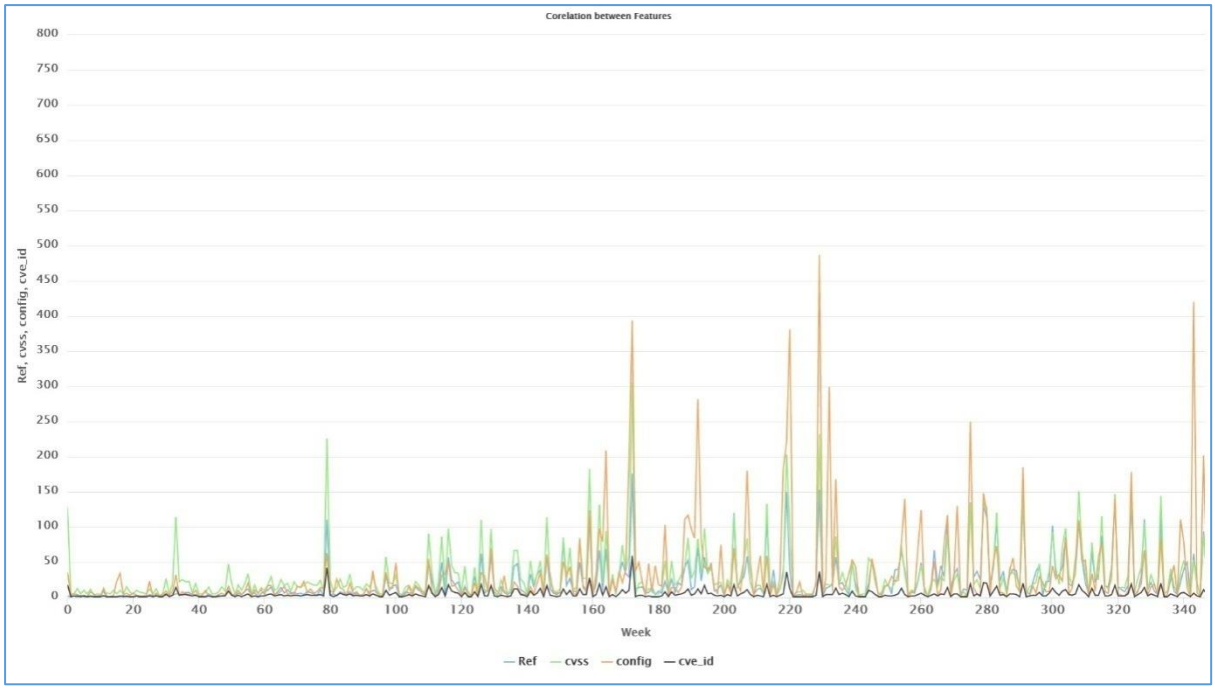


Figure 5: Line plot among cve_id, conFigure, Ref and cvss for Microsoft vulnerability data (zoomed for week no 0-340).

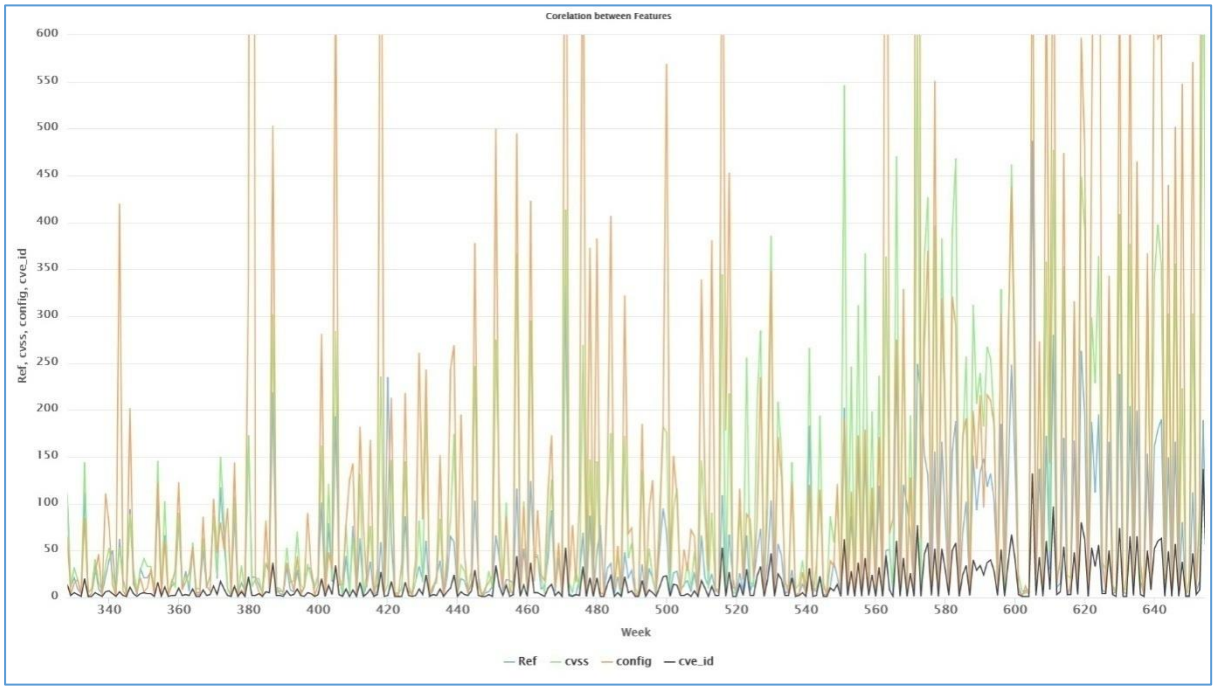


Figure 6: Line plot among cve_id, conFigure, Ref and cvss for Microsoft vulnerability data (zoomed for week no 340-650).

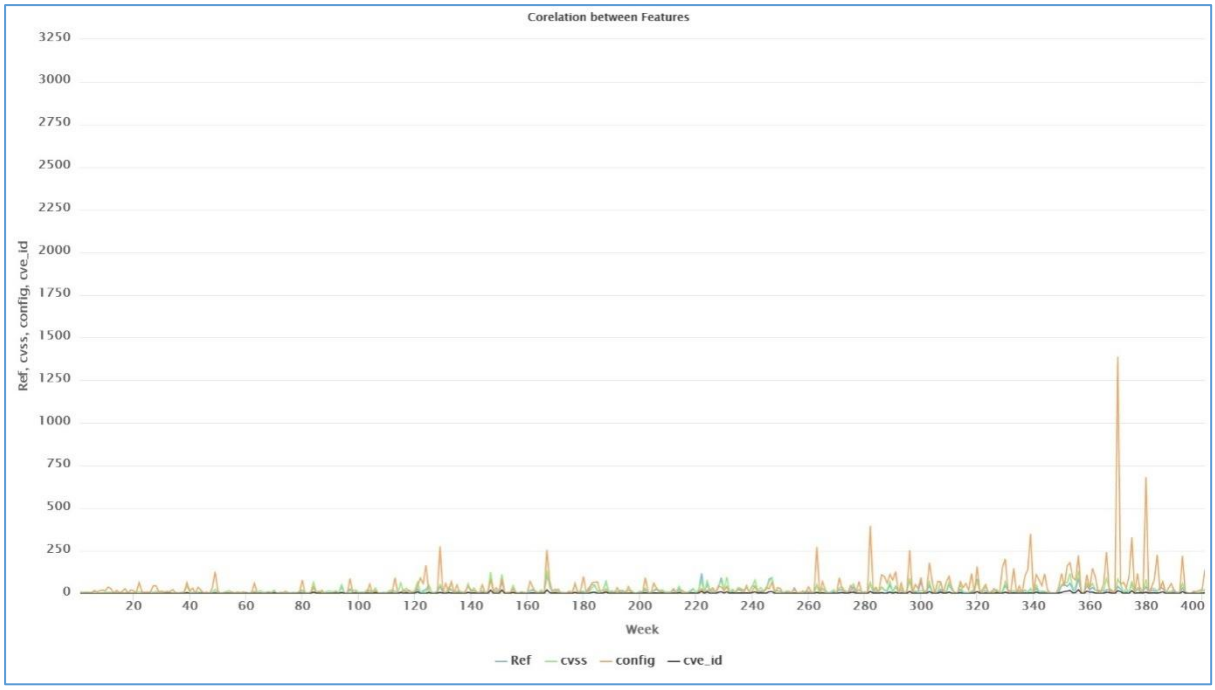


Figure 7: Line plot among cve_id, conFigure, Ref and cvss for Oracle vulnerability data (zoomed for week no 0-400).

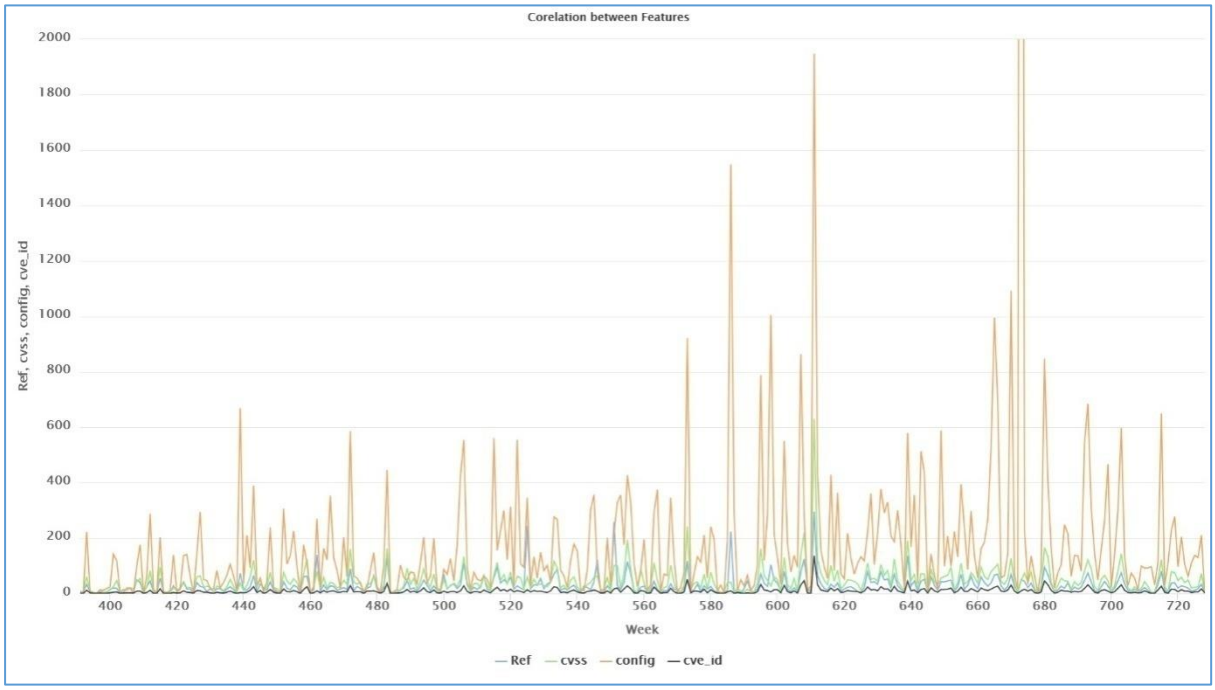


Figure 8: Line plot among cve_id, conFigure, Ref and cvss for Oracle vulnerability data (zoomed for week no 400-700).

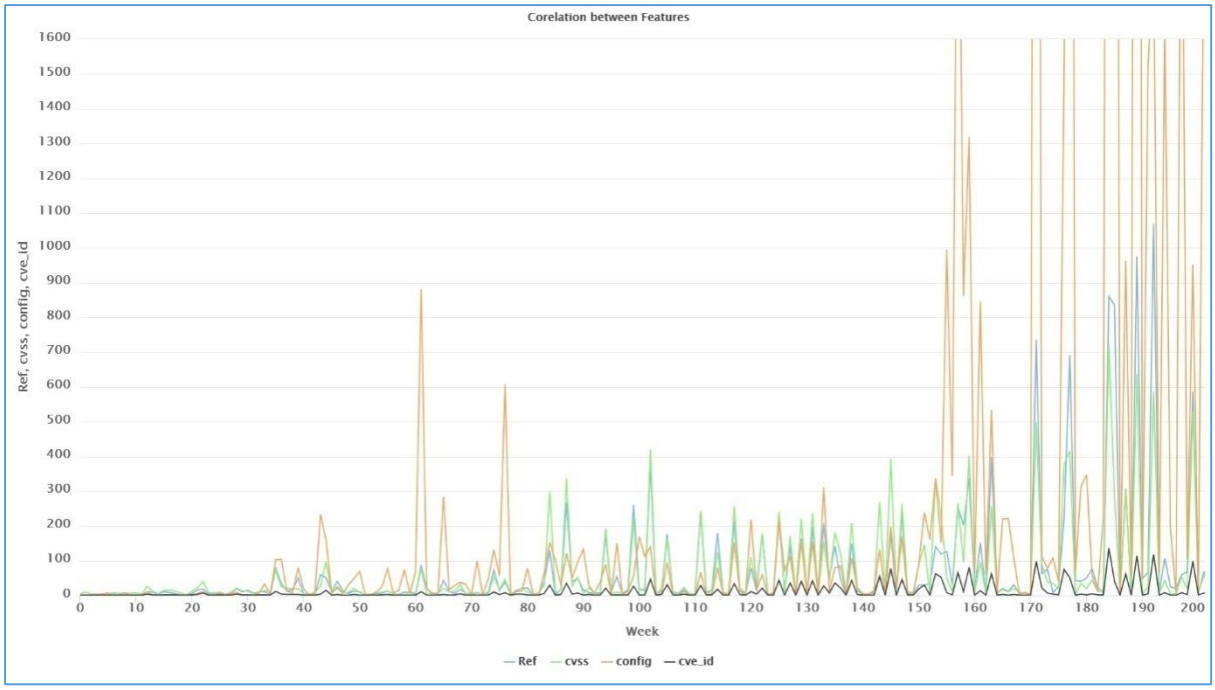


Figure 9: Line plot among cve_id, conFigure, Ref and cvss for IBM vulnerability data (zoomed for week no 0-200).

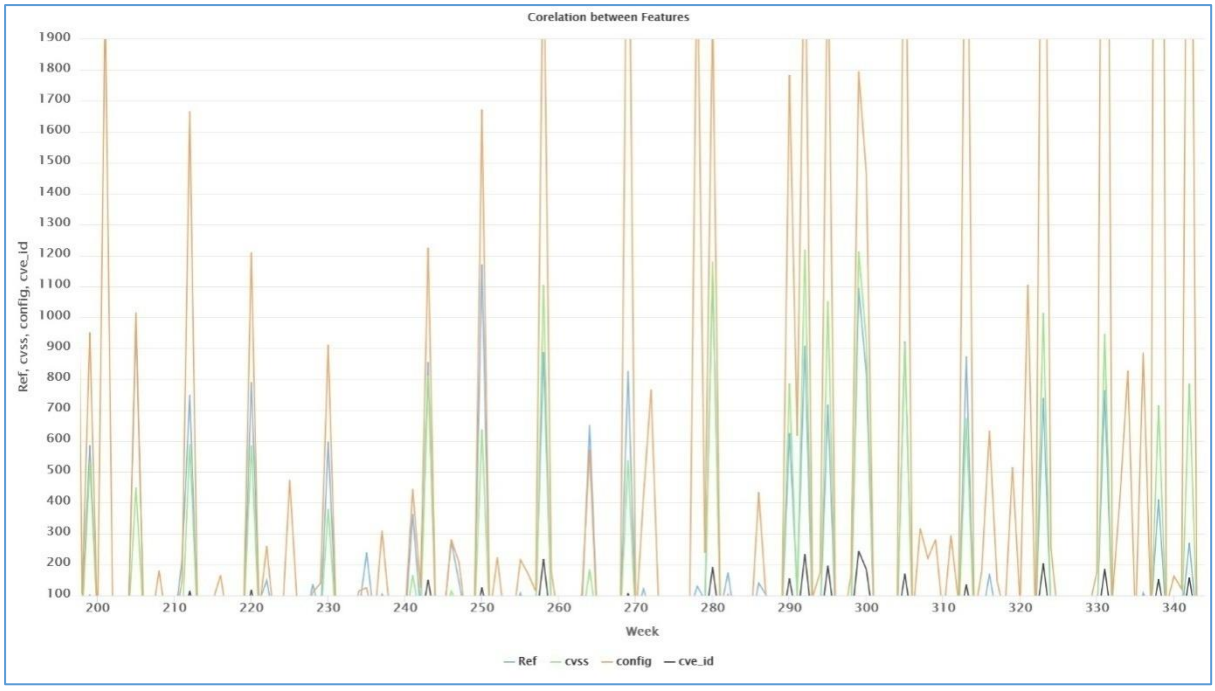


Figure 10: Line plot among cve_id, conFigure, Ref and cvss for IBM vulnerability data (zoomed for week no 700-340).

We have utilized MS Excel, Studio 3T trail edition for working with mongodb, RapidMiner Studio educational edition and Anaconda (Spyder) for this research work. Supervised Linear and non-linear machine learning algorithms such as Generalized Linear Model, Deep Learning, Decision Tree, Random Forest, Gradient Boosted Trees and Support Vector Machine were utilized for this regression prediction problem. We utilized "root_mean_squared_error" (RMSE) which penalizes large errors more and "squared_error" (MAE) for performance measurement of the models. The data was split by 60% and 40% by number of respective weeks for each vendors as training and test datasets respectively.

4. RESULTS AND ANALYSIS

Table 1-3 shows the model performances for Microsoft, Oracle and IBM vulnerabilities predictions respectively.

Random Forest showed the best performance with RMSE and MAE as low as 1.47 and 2.38 respectively for Microsoft dataset whereas results from other models are also very promising.For Oracle dataset Generalized Linear Model performed as the best model with RMSE and MAE value for 1.483208262 and 2.218468231 respectively. Lastly Generalized Linear Model with RMSE and MAE values for 4.67876 and 26.19466308 respectively showed best performance for IBM dataset. A bit poorer performance for IBM dataset in comparison with Microsoft and Oracle dataset mostly due to having less number of data rows for modelling.

Table 1: Performance of Models for Microsoft dataset.

Model	Root Mean Squared Error	Squared Error (MAE)
Generalized Linear Model	2.889268086	9.204168435
Deep Learning	2.265379391	8.096902506
Decision Tree	3.485938542	19.97803745
Random Forest	1.472581146	2.387238205
Gradient Boosted Trees	2.03465883	4.386616251
Support Vector Machine	1.907887381	4.83589023

Table 2: Performance of Models for Oracle dataset.

Model	Root Mean Squared Error	Squared Error (MAE)
Generalized Linear Model	1.483208262	2.218468231
Deep Learning	1.515252726	2.64331362
Decision Tree	2.116960694	4.862272003
Random Forest	2.493036872	7.441452768
Gradient Boosted Trees	2.471268936	6.729051211
Support Vector Machine	3.337078162	16.08689261

Table 3: Performance of Models for IBM dataset.

Model	Root Mean Squared Error	Squared Error (MAE)
Generalized Linear Model	4.67876	26.19466308
Deep Learning	5.224022	38.77076466
Decision Tree	9.416095	114.4626701
Random Forest	8.256984	81.88893276
Gradient Boosted Trees	11.46529	157.1603427
Support Vector Machine	3.140617	11.45106563

Tables 4-12 show the contribution by weights for the attributes/features for modelling results. These tables also have nested tables that show contribution weights for features for overall modelling result, contribution weights for features for best performing model and correlation between features respectively for each vendor dataset. These resulting clearly again show that “cvss” is the most important contributing weight followed by “Ref” and “conFigure” whereas feature “days_diff” having very minimal or no contribution. We have also observed these outcome in the line diagrams described before. Table 13 shows the best performing models for each vendor dataset. Table 15 shows a comparison of results between our proposed models and a recent model [18]. As can be seen our models achieved better accuracy. Figureure 11-13 shows the line charts for predictions on test dataset for vendor Microsoft, Oracle and IBM respectively. Table 14 shows the regression coefficients for GLM for Oracle (snipped from RapidMiner result view).

Table 4: Overall weights of features form all model

Attribute	Weight
cvss	0.995947
Ref	0.187547
conFigure	0.065441
days_diff	0

(Microsoft).

Table 5: Overall weights of features form best performing RF model.

Attribute	Weight
cvss	0.825866
Ref	0.190402
conFigure	0.146933
days_diff	0.05845

Table6: Correlation between attributes (Microsoft dataset).

Attribut...	config	cve_id	cvss	days_diff	Ref
config	1	0.750	0.747	0.264	0.598
cve_id	0.750	1	0.984	0.432	0.874
cvss	0.747	0.984	1	0.474	0.875
days_diff	0.264	0.432	0.474	1	0.505
Ref	0.598	0.874	0.875	0.505	1

Table 7: Overall weights of features form all model

Attribute	Weight
cvss	0.97806
Ref	0.77963
conFigure	0.407988
days_diff	0.219114

(Oracle).

Table 8: Overall weights of features form best performing Generalized Linear model.

Attribute	Weight
cvss	0.999117873
Ref	0.100735028
conFigure	0.03580796
days_diff	0

Table 9: Correlation between attributes (Oracle dataset).

Attribut...	config	cve_id	cvss	days_diff	Ref
config	1	0.408	0.389	0.009	0.383
cve_id	0.408	1	0.978	0.219	0.780
cvss	0.389	0.978	1	0.317	0.800
days_diff	0.009	0.219	0.317	1	0.385
Ref	0.383	0.780	0.800	0.385	1

Table 10: Overall weights of features form all model

Attribute	Weight
cvss	0.993039
Ref	0.894051
conFigure	0.676475
days_diff	0.512685

(Oracle).

Table 11: Overall weights of features form best performing SVM model.

Attribute	Weight
cvss	0.991643163
Ref	0.086739088
conFigure	0.069472242
days_diff	0

Table 12: Correlation between attributes (IBM dataset).

Attribut...	config	cve_id	cvss	days_diff	Ref
config	1	0.676	0.679	0.378	0.716
cve_id	0.676	1	0.993	0.513	0.894
cvss	0.679	0.993	1	0.553	0.906
days_diff	0.378	0.513	0.553	1	0.596
Ref	0.716	0.894	0.906	0.596	1

.

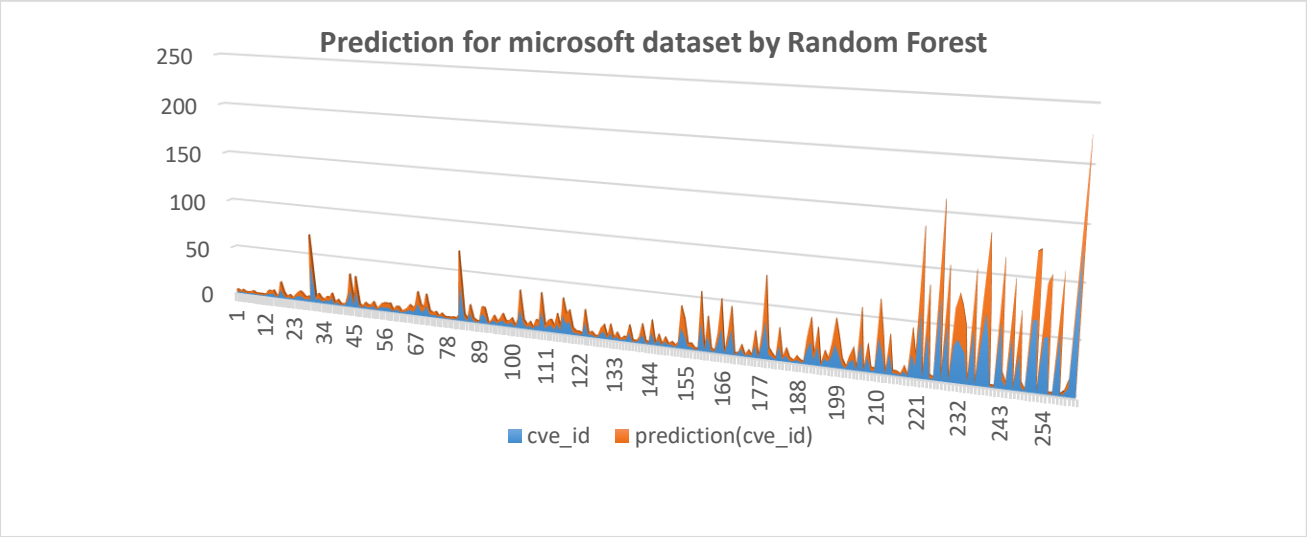


Figure 11: Prediction on test dataset (Microsoft, original=blue, prediction=orange).

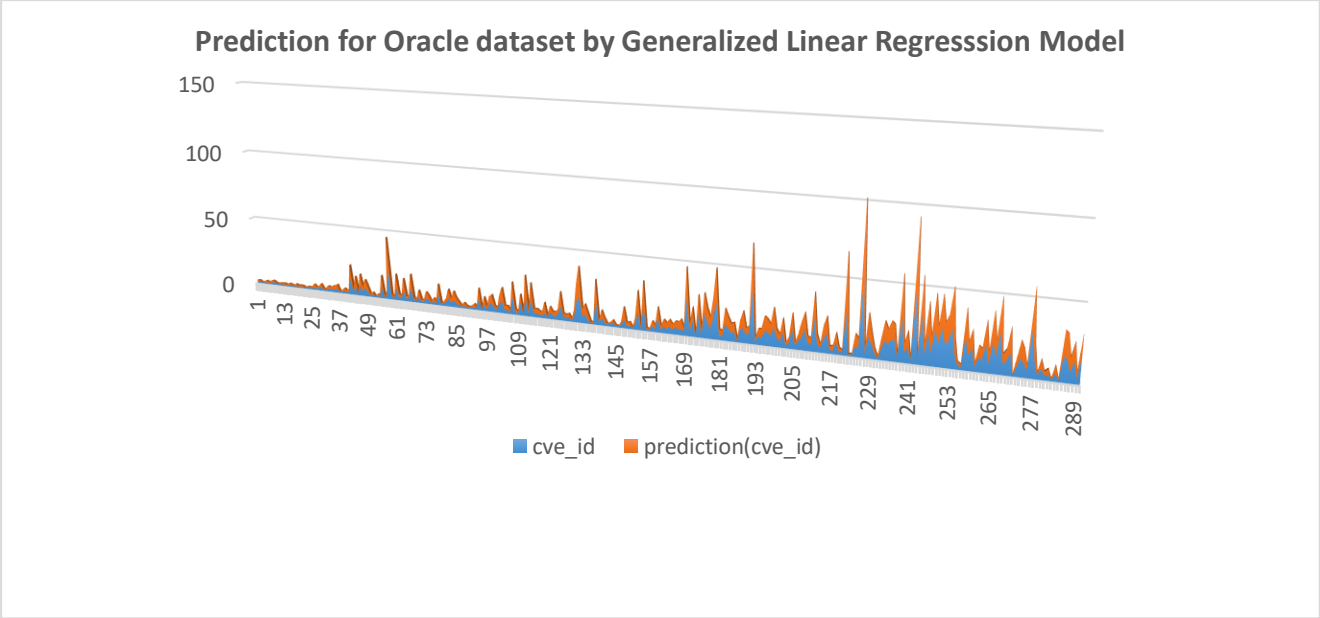


Figure 12: prediction on test dataset (Oracle, original=blue, prediction=orange).

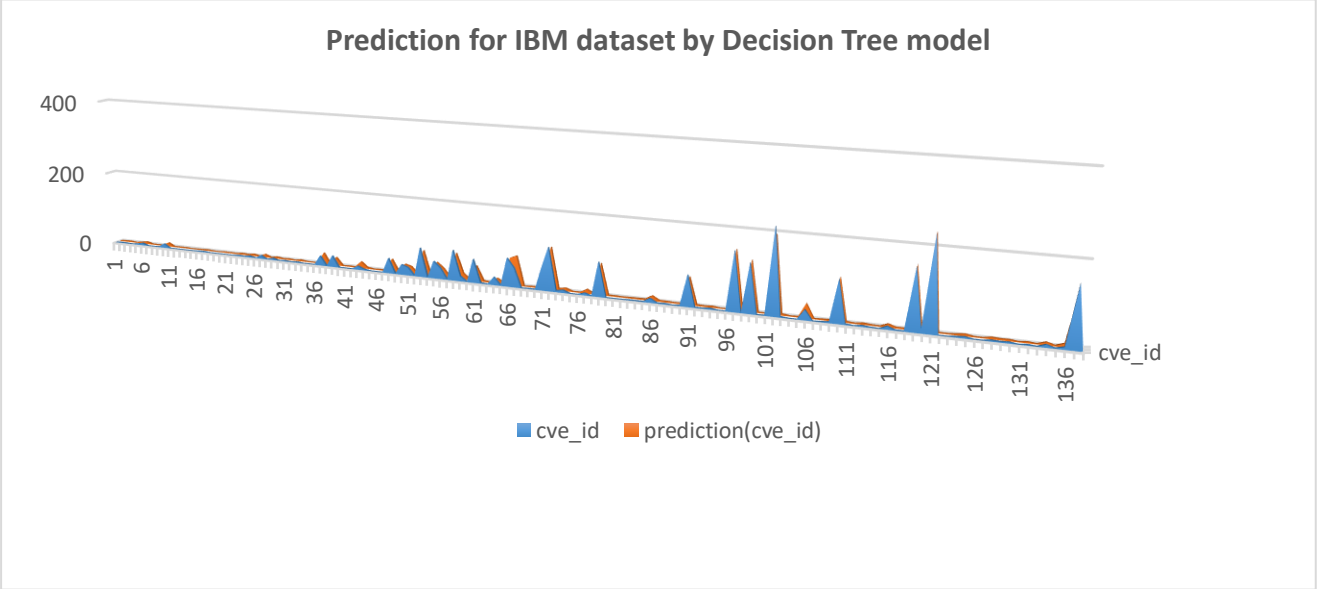


Figure 13: Prediction on test dataset (IBM, original=blue, prediction=orange).

Table 13: Selection of best Models.

Vendor	Best Model	RMSE	Squared Error (MAE)
Microsoft	Random Forest	1.472581146	2.387238205
Oracle	Generalized Linear Model	1.483208262	2.218468231
IBM	Support Vector Machine	3.140617	11.45106563

The purpose of multiple regression is to predict a single variable from one or more independent variables. The prediction of Y is accomplished by the following equation, as in

$$Y'_i = b_0 + b_1X_{1i} + b_2X_{2i} + \ldots + b_kX_{ki} \quad (1)$$

The "b" values are called regression weights or regression coefficient and are computed in a way that minimizes the sum of squared deviations, as in

$$\sum_{i=1}^{N}(Y_i - Y'_i)^2 \quad (2)$$

b_0 (interceptor) is the value of Y when all of the independent variables (X_{1i} through X_{ki}) are equal to zero. Each regression coefficient represents the change in Y relative to a one unit change in the respective independent variable. We have then developed linear regression equation for our best model for Oracle since Generalized Linear Model performed best for this vendor.

Therefore regression equation for vendor Oracle would be as follows:

Y (vulnerability count) = -0.096+0.288(cvss) + 0.002(Ref).

Table 14: Regression coefficients for GLM (Oracle dataset).

Attribute	Coefficient	Std. Coefficient
Ref	0.002	0.055
cvss	0.208	8.310
days_diff	-0.000	-0.632
config	0.000	0.077
Intercept	-0.096	6.249

Table 15: comparison of results between Models proposed by [18] and ours.

Models proposed by [18]			Our proposed Models		
Vendor	Best Model	RMSE	Vendor	Best Model	RMSE
MAC OS X	ARIMA	19.64	Microsoft	Random Forest	1.47
Windows 7	SVM	3.58	Oracle	Generalized Linear Model	1.48
Linux Kernel	SVM	3.99	IBM	Support Vector Machine	3.14

5. CONCLUSION, LIMITATIONS AND FUTURE WORKS

We have utilized the time series algorithm to predict future vulnerabilities of the top 3 vendors namely Microsoft, Oracle and IBM, whose vulnerabilities data were taken from [19] and developed 6 supervised linear and non-linear machine learning models such as Generalized Linear Model, Deep Learning, Decision Tree, Random Forest, Gradient Boosted Trees and Support Vector Machine to predict number of vulnerabilities. In doing so we have intuitively created feature dataset on 7 days interval for vulnerability count and 4 new features with aggregation. To the best of our knowledge this work is the first to utilize such feature dataset to develop a VDM. Our model achieved better RMSE results than other models such as the recent one in [18]. The developers, the user community, and individual organization can utilize our model as their respective requirements.

Although we believe that this study makes anumber of contributions, it has some limitations. Such as we have utilized public data from NVD and have not considered unreported or undisclosed data. Another limitation with this public dataset is that the recorded published of the vulnerabilities are not always as per their real detection date as may vulnerabilities usually are detected or utilized earlier by many interested users than they are officially recorded by NVD.

We believe that there is much to do for future research that may include enhancing accuracy further by incorporating new independent features, for example vulnerability severity. Also vulnerability data from other open or commercial databases can be utilized to overcome dataset limitations. Another promising future may involve utilizing non-linear Recurrent Neural Network (RNN) as time series prediction models which can handle modelling dependencies better between two points in a time series.

ACKNOWLEDGEMENT

This research is supported by TNB Seed Fund 2019 with a project code U-TC-RD-19-09 and ICT Ministry, Bangladesh.

REFERENCES

1. B.Manoj, V.K.Sasikanth, V.Subbarao, Jyothi Prakash, Analysis of Data Science with the use of Big Data, International Journal of Advanced Trends in Computer Science and Engineering Volume 7, No.6, November - December 2018. https://doi.org/10.30534/ijatcse/2018/02762018
2. HP Identifies Top Enterprise Security Threats 2014, available at:https://www8.hp.com/us/en/hp-news/press-release.html?id=1571359. Accessed on December, 2009.
3. S. Frei, M. May, U. Fiedler, and B. Plattner, Large-scale Vulnerability Analysis, ser. LSAD ’06. ACM, 2006.

4. S. Frei, D. Schatzmann, B. Plattner, and B. Trammell, "Modelling the Security Ecosystem- The Dynamics of (In)Security." in WEIS, 2009.
5. OumaymaOueslati, Ahmed Ibrahim S. Khalil, Habib Ounelli, Sentiment Analysis for Helpful Reviews Prediction, International Journal of Advanced Trends in Computer Science and Engineering, Volume 7, No.3, May- June 2018.
6. MACHINE LEARNING, available on 2019 at: http://www.contrib.andrew.cmu.edu/~mndarwis/ML.html.
7. Mourad A, Laverdiere MA, Debbabi M. An aspect-oriented approach for the systematic security hardening of code. ComputSecur 2008a; 27(3):101e14.
8. Mourad A, Laverdiere MA, Debbabi M. A high-level aspect- oriented-based framework for software security hardening. Inform Secur J A Glob Perspect 2008b; 17(2):56e74.
9. Zhuobing Han, Xiaohong Li, Zhenchang Xing, Hongtao Liu, and Zhiyong Feng. Learning to Predict Severity of Software Vulnerability Using Only Vulnerability Description. In ICSME, pages 125–136. IEEE, September 2017.
10. Okamura, H., Tokuzane, M., Dohi, T., Optimal Security Patch Release Timing under Non homogeneous Vulnerability-Discovery Processes, 2009. pp. 120–128.
11. Lyu, M.R. (Ed.), Handbook of software reliability engineering. IEEE Computer Society Press; McGraw Hill, Los Alamitos, Calif. : New York. 1996.
12. Ozment, J.A., 2007. Vulnerability discovery & software security. University of Cambridge Retrieved from (available on 2019).
13. Rescorla, Eric, Security holes... Who cares? August. Presented at the USENIX Security, 2003.
14. Rescorla, E., Is finding security holes a good idea? IEEE Security and Privacy Magazine, 2005, 3 (1), 14–19. https://doi.org/10.1109/MSP.2005.17.
15. Alhazmi, O.H., Malaiya, Y.K., Modeling the vulnerability discovery process. In: 16th IEEE International Symposium on Software Reliability Engineering (ISSRE'05). 2005a, 10 pp–138.
16. Alhazmi, O.H., Malaiya, Y.K., Quantitative vulnerability assessment of systems software. In: Proceedings of Annual Reliability and Maintainability Symposium. IEEE, 2005b. pp. 615–620.
17. Roumani, Y., Nwankpa, J.K., Roumani, Y.F., Time series modeling of vulnerabilities. Computers & Security, 2015, 51, 32–40.
18. Pokhrel, N.R., Rodrigo, H., Tsokos, C.P, Cybersecurity: Time Series Predictive Modeling of Vulnerabilities of Desktop Operating System Using Linear and NonLinear Approach. Journal of Information Security, 2017, 08 (04), 362–382. https://doi. org/10.4236/jis.2017.84023.
19. Movahedi, Y., Cukier, M., Andongabo, A., Gashi, I., Cluster-based Vulnerability Assessment Applied to Operating Systems. In: Presented at the 13th European Dependable Computing Conference. Geneva, Switzerland, 2017.
20. Li, P., Shaw, M., Herbsleb, J., Selecting a defect prediction model for maintenance resource planning and software insurance. EDSER-5 Affiliated with ICSE, 2003. P32–p37.
21. Top 50 Vendors by Total Number of "Distinct" Vulnerabilities, available on 2019 at: https://www.cvedetails.com/top-50-vendors.php.
22. M. Xie, Software Reliability Modelling, World Scientific Publishing Co. Pte. Ltd, Singapore, 1991, ch 1, pp10-11. 22.
23. R. J. Anderson, "Security in Opens versus Closed Systems—the Dance of Boltzmann, Coase and Moore," Open Source Software: Economics, Law and Policy, Toulouse, France, June 20-21, 2002.
24. Kim, J., Malaiya, Y.K., Ray, I., 2007. Vulnerability Discovery in Multi-Version Software Systems. In: 10th IEEE High Assurance Systems Engineering Symposium, 2007. HASE '07, pp. 141–148.
25. Woo, S., Alhazmi, O., Malaiya, Y., 2006. Assessing Vulnerabilities in Apache and IIS HTTP Servers, pp. 103–110.
26. Alhazmi, O.H., Malaiya, Y.K., Application of Vulnerability Discovery Models to Major Operating Systems. IEEE Transactions on Reliability, 2008. 57 (1), 14–22..
27. Massacci, F., Nguyen, V.H., An Empirical Methodology to Evaluate Vulnerability Discovery Models. IEEE Transactions on Software Engineering, 2014. 40 (12), 1147–1162.
28. Shrivastava, A.K., Sharma, R., Kapur, P.K., Vulnerability discovery model for a software system using stochastic differential equation. In: International Conference on Futuristic Trends on Computational Analysis and Knowledge Management (ABLAZE), 2015. pp. 199–205.
29. Joh, H., Malaiya, Y.K., Modeling Skewness in Vulnerability Discovery: Modeling Skewness in Vulnerability Discovery. Quality and Reliability Engineering International, 2014. 30 (8), 1445–1459.
30. Rescorla, E., Is finding security holes a good idea? IEEE Security and and Privacy Magazine 3 (1), 14–19. Magazine 3 (1), 14–19.
31. Younis, A.A., Joh, H., Malaiya, Y., Modeling Learningless Vulnerability Discovery using a Folded Distribution. In: Proceedings of the International Conference on Security and Management (SAM), 2011. pp. 617–623.
32. Alhazmi, O.H., Malaiya, Y.K., Measuring and Enhancing Prediction Capabilities of Vulnerability Discovery Models for Apache and IIS HTTP Servers. In: 17th International Symposium on Software Reliability Engineering, 2006. pp. 343–352.
33. Wang, L., Zeng, Y., Chen, T., Back propagation neural network with adaptive differential evolution algorithm for time series forecasting. Expert Systems with Applications, 2005. 42 (2), 855863.
34. NATIONAL VULNERABILITY DATABASE, available at: https://nvd.nist.gov/.

35. CVE-Search, available and accessed on 2019 at: https://github.com/cve-search/cve-search.
36. Louis, YeowHaurTeng and Kuok King Kuok, Development of Whale Optimization Neural Network for Daily Water Level Forecasting. International Journal of Advanced Trends in Computer Science and Engineering, Volume 8, No.3, May - June 2019.
37. Danish Ahamad, MD Mobin Akhtar and ShabiAlam Hameed, A Review and Analysis of Big Data and MapReduce. International Journal of Advanced Trends in Computer Science and Engineering, Volume8, No.1, January-February 2019.