Network Security: Nonlinear Arbitrary Models for Predicting Vulnerabilities.

Mr.K.Sriram
3rd Year, Department of CSE
Adhi College of Engineering and Technology
Kanchipuram, Tamilnadu
sriramkesavan98@gmail.com

Aim : To predict Vulnerabilities that can be patched by using Model.

***Abstract*— Obtaining complete information regarding discovered vulnerabilities looks extremely difficult. Yet, developing statistical models requires a great deal of such complete information about the vulnerabilities. I introduced the use of Markovian approach to estimate the probability of a particular vulnerability being at a particular "state" of the vulnerability life cycle. In this study, we further develop our models, use available data sources in a probabilistic foundation to enhance the reliability and also introduce some useful new modeling strategies for vulnerability risk estimation. Finally, we present a new set of Non-Linear Statistical Models that can be used in estimating the probability of being exploited as a function of time. Our study is based on the typical security system and vulnerability data that are available. However, our methodology and system structure can be applied to a specific security system by any software engineer and using their own vulnerabilities to obtain their probability of being exploited as a function of time. This information is very important to a company's security system in its strategic plan to monitor and improve its process for not being exploited.**

Keywords*— Vulnerability Lifecycle, Arbitrary Modeling, Security Risk Factor, Markov Process, Risk Evaluation

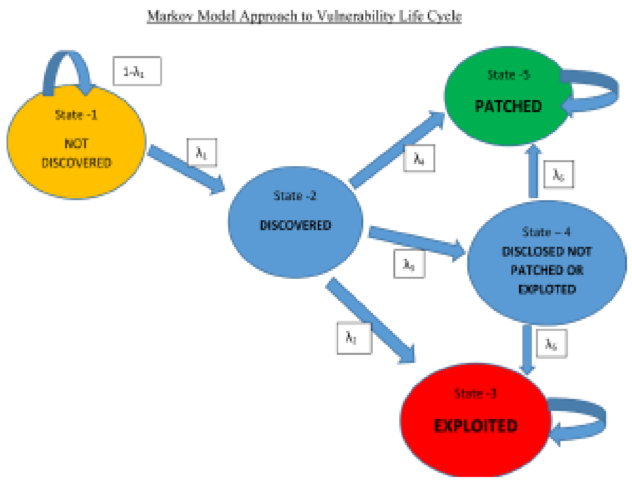
Introduction

"Risk" is an unavoidable phenomenon in the Cyber world. Information systems ranging from very small and personal level apps to massive corporate and government applications and system platforms are facing the threat from Cyber-attacks [1] in various dimensions.

The objective of this study is to propose and present a rational set of methods to identify the probabilities for each different state in the vulnerability life cycle [2] [4] [5] and use this information to develop three different statistical models to evaluate the **"Risk Factor"** [2] [5] of a particular vulnerability at time "t". In our recent study **"Stochastic Modelling of Vulnerability Life Cycle and Security Risk Evaluation" (Journal of Information Security, 7, 269-279)** [2], we introduced the strategy of using Markov processes to obtain the "transition probability matrix" of all the states of a particular vulnerability as a function of time. We iterated the Markov process and determined that it reached the "steady state" with probabilities of reaching the "absorbing states" [1] [2].

1.1. Vulnerability Life Cycle Analysis Method

In our previous study [2], we introduced the use of Markov chain process to develop the transition probability matrix including all the important states of Vulnerability Life Cycle. The Vulnerability Life Cycle Graph that we discussed is presented below by "Figure 1". When we draw a Life Cycle Graph for a given vulnerability, it has several nodes which represent the stages of the Vulnerability Life Cycle. Earlier we assigned logical probabilities for a hacker to reach each state by examining the properties of a specific vulnerability.



2.0 Methodology

2.1. Methodology of Assigning Initial Probabilities

Probability	State Represented
λ_1	Discovered
λ_2	Exploited before patched or disclosed
λ_3	Disclosed but not yet patched

	or exploited
λ_4	Patched before disclosed
λ_5	Exploited after disclosed
λ_6	Patched after disclosed

2.2.Common Vulnerability Scoring System (CVSS) and Common Vulnerabilities and Exposures (CVE)

It is important to discuss here the usage of Common Vulnerability Scoring System (CVSS) [8] and CVE Details [9], as we gather data from those resources. Common Vulnerability Scoring System (CVSS) is the commonly used and freely available standard for assessing the magnitude of Information System Vulnerabilities. CVSS gives a score for each vulnerability scaling from 0 to 10 based on several factors. National Vulnerability Database (NVD) [3] provides CVSS score and updates continuously as new vulnerabilities are discovered. CVSS score is calculated using three main matrices named, Base Matric, Temporal Metric and Environmental Metric. However, NVD data base provides us with the Base Metric Scores for the Vulnerability only because the Temporal and Environmental Scores are varied on other factors related to the organization that uses the computer system. The Base score for more than 75,000 different vulnerabilities are calculated using 6 different Matrices. It is managed by the Forum of Incident Response and Security Teams.

3.TEST CASES.

Before a test case is carried out the existing bug reports are exported the ReBug Scan Model and predefined set of payloads are set in the model for intense scan to produce the λ value absolutely. The existing bug reports are fed in the model by using the payload text file (Refer image below). The existing bug are tested by the end points. Some unknown issues are also fed to find new bugs/ vulnerabilities. If a bug is fed with the CV Score of $\lambda_= 9$, the model is iterated for the series of continuous testing to satisfy the condition of $\lambda_= 0$.

Refer below table:

λ = 0-4	Low Severity bug
λ = 5-7	Medium Severity bug
λ = 8-10	High Severity bug

core
install
payload
ReBug_Scan

Now the model will scan for the bug/ vulnerabilities of specific vulnerability score. Until a condition of $\lambda = 0$ must be satisfied. This test is performed for infinite loop until the condition is satisfied. Below image represents a pictorial model.

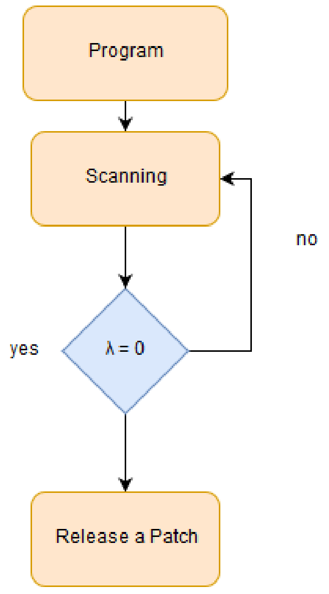


Fig :Model of ReBug Scan

4.PATCHING METHOD

The test cases are used to find the bugs that are already patched on the existing models. Now it is retested and checked if its reproduced again. If the bug is reproduced again it is sent to the security team to manually patch the bugs. If a bug is patched the models is modified according to the recent update on the patch. The only drawback of this model is to update the package again to again to prevent the model getting rescanned again and again. The bugs can only be patched by a manual tester.

Bugs patched by this model

- Twitter parameters tampering on user unsubscribe.
- Google improper authentication on cookies. Reported by myself.
- Zomato reflected XSS bug on user dashboard.

4.1. TECHNOLOGIES USED

Python is used as the basic scanning tool with the existing model named Gloom which is an open source model published on Github. With this model the endpoints can be tested and vulnerabilities can be exposed. And by using the ReBug scan model to check the λ model and if the condition satisfies. When the condition is satisfied the model proceeds for next end points of the URL. Python is the basic language used in this model and API of several search engines are used such as Shodan. [https://account.shodan.io]. Several models are also included and built which also makes it as a bug search model.

5. *CONCLUSION*

In this study, we continue to improve the models we build up in our previous study [2]. We have improved the calculation methods of initial probabilities and created the Transition Probability Matrix in using of the Markovian process that we introduced in our previous studies. We used CVSS data presented in CVE details website and calculated initial probabilities for discovering and exploiting a vulnerability based on the records on last 17 years data. Finally, we created two sets of three models for predicting the risk of a particular vulnerability being exploited as a function of time. The models we presented are proven to have an excellent fit with the Markovian process probabilities. Therefore, we can replace the Markovian process using these models since these models enable us to get rid of analytical requirement to execute the Markovian iteration process of identifying the steady states of being exploited or being patched for each vulnerability.

References

[1] Kaluarachchi, P.K., Tsokos, C.P. and Rajasooriya, S.M. (2016) Cybersecurity: A Statistical Predictive Model for the Expected Path Length. Journal of Information Security,7,112-128. https://doi.org/10.4236/jis.2016.73008

[2] Rajasooriya, S.M., Tsokos, C.P. and Kaluarachchi, P.K. (2016) Stochastic Modelling of Vulnerability Life Cycle and Security Risk Evaluation. Journal of information Security,7,269-279. https://doi.org/10.4236/jis.2016.74022

[3] NVD. National Vulnerability Database. http://nvd.nist.gov/

[4] Frei, S. (2009) Security Econometrics: The Dynamics of (IN) Security. PhD Dissertation, ETH, Zurich.

[5] Joh, H. and Malaiya, Y.K. (2010) A Framework for Software Security Risk Evaluation Using the Vulnerability Lifecycle and CVSS Metrics. Proceedings of the International Workshop on Risk and Trust in Extended Enterprises, November 2010, 430-434.

[6] Kijsanayothin, P. (2010) Network Security Modeling with Intelligent and Complexity Analysis. PhD Dissertation, Texas Tech University, Lubbock, TX.

[7] Alhazmi, O.H., Malaiya, Y.K. and Ray, I. (2007) Measuring, Analyzing and Predicting Security Vulnerabilities in Software Systems. Computers & Security,26,219-228. https://doi.org/10.1016/j.cose.2006.10.002

[8] Schiffman, M. Common Vulnerability Scoring System (CVSS). http://www.first.org/cvss/

[9] CVE Details. http://www.cvedetails.com/

[10] Secunia Vulnerability Review 2015: Key Figures and Facts from a Global Information Security Perspective. March 2015 https://secunia.com/?action=fetch&filename=secunia_vulnerability_review_2015.pdf

[11] Alhazmi, O.H. and Malaiya, Y.K. (2008) Application of Vulnerability Discovery Models to Major Operating Systems. IEEE Transactions on Reliability, 57, 14-22. https://doi.org/10.1109/TR.2008.916872

[12] Alhazmi, O.H. and Malaiya, Y.K. (2005) Modeling the Vulnerability Discovery Process. Proceedings of 16th International Symposium on Software Reliability Engineering, Chicago, 8-11 November 2005, 129-138. https://doi.org/10.1109/ISSRE.2005.30

[13] Noel, S., Jacobs, M., Kalapa, P. and Jajodia, S. (2005) Multiple Coordinated Views for Network Attack Graphs. VIZSEC'05: Proceedings of the IEEE Workshops on Visualization for Computer Security, Minneapolis, MN, 26 October 2005, 99-106. https://doi.org/10.1109/vizsec.2005.1532071

[14] Mehta, V., Bartzis, C., Zhu, H., Clarke, E.M. and Wing, J.M. (2006) Ranking Attack Graphs. In: Zamboni, D. and Krügel, C., Eds., Recent Advances in Intrusion Detection, Vol. 4219 of Lecture Notes in Computer Science, Springer, Berlin, 127-144.

[15] Lawler, G.F. (2006) Introduction to Stochastic Processes. 2nd Edition, Chapman and Hall/CRC, Taylor and Francis Group, London, New York.

[16] Abraham, S. and Nair, S. (2014) Cyber Security Analytics: A Stochastic Model for Security Quantification Using Absorbing Markov Chains. Journal of Communications, 9, 899-907. https://doi.org/10.12720/jcm.9.12.899-907

[17] Jajodia, S. and Noel, S. (2005) Advanced Cyber Attack Modeling, Analysis, and Visualization. 14th USENIX Security Symposium, Technical Report 2010, George Mason University, Fairfax, VA.

[18] Wang, L., Singhal, A. and Jajodia, S. (2007) Measuring Overall Security of Network Configurations Using Attack Graphs. In: Barker, S. and Ahn, G.J., Eds., Data and Applications Security XXI. DBSec 2007. Lecture Notes in Computer Science, Vol. 4602, Springer, Berlin, Heidelberg, 98-112. https://doi.org/10.1007/978-3-540-73538-0_9

[19] Wang, L., Islam, T., Long, T., Singhal, A. and Jajodia, S. (2008) An Attack GraphBased Probabilistic Security Metric. DAS 2008, LNCS 5094, 283-296.