June 2017

Cybersecurity: Probabilistic Behavior of Vulnerability and Life Cycle

Sasith Maduranga Rajasooriya
University of South Florida, sasith@mail.usf.edu

Cybersecurity: Probabilistic Behavior of Vulnerability and Life Cycle.

by

Sasith Maduranga Rajasooriya

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
Department of Mathematics and Statistics
College of Arts and Sciences
University of South Florida

Major Professor: Chris P. Tsokos, Ph.D.
Kandethody Ramachandran, Ph.D.
Dan Shen, Ph.D.
Lu Lu, Ph.D.

Date of Approval:
June 23, 2017

Keywords: Cyber Security, Markov Model, Vulnerability, Risk Factor, Vulnerability Life cycle

Dedication

This doctoral dissertation is dedicated to my father, my mother and my wife.

Acknowledgments

Last three years of my life at USF has been a wonderful academic experience for me. I am grateful that I received this opportunity to study at USF. It is with gratitude that I remind everyone who helped me in numerous ways during the last three years.

My heart full gratitude and respect is paid to my major professor and adviser Dr. Chris Tsokos, the Distinguished University Professor at the Department of Mathematics and Statistics at USF. His guidance, dedicated teaching and motivation has inspired me in many ways. I am grateful for his fruitful advices, exceptional consideration for students and his gifts from his vast knowledge in the subject area.

I am very much grateful to Prof. Kandethody Ramachandran, Prof. Dan Shen and Prof. Lu Lu for their kind assistance and time spent as my dissertation committee members.

I would like to express my thanks to all the faculty members in the Department of Mathematics and Statistics for their dedicated teaching in the courses I have taken. My gratitude goes to the administration of the Department of Mathematics and Statistics for all the helps and resources made available to me as a student.

My thanks go to Knowbe4.Inc at Clearwater for offering me the opportunity of Summer Internship as a Cybersecurity Data Analyst in 2016.

My strength was always my wife behind me all these time. My thanks goes to Pubudu with my love.

It is with pleasure, I remind all my beloved friends, fellow graduate students for all their support and friendship.

Table of Contents

List of Tables

List of Figures

Abstract

Analysis on Vulnerabilities and Vulnerability Life Cycle is at the core of Cybersecurity related studies. Vulnerability Life Cycle discussed by S. Frei and studies by several other scholars have noted the importance of this approach. Application of Statistical Methodologies in Cybersecurity related studies call for a greater deal of new information. Using currently available data from National Vulnerability Database this study develops and presents a set of useful Statistical tools to be applied in Cybersecurity related decision making processes.

In the present study, the concept of Vulnerability Space is defined as a probability space. Relevant theoretical analyses are conducted and observations in the vulnerability space in aspects of events and states are discussed.

Transforming IT related cybersecurity issues into analytical formation so that abstract and conceptual knowledge from Mathematics and Statistics can be applied is a challenge. However, to overcome rising threats from Cyber-attacks such an integration of analytical foundation to understand the issues and develop strategies is essential. In the present study we apply well known Markov approach in a new approach of Vulnerability Life Cycle to develop useful analytical methods to assess the

Risk associated with a vulnerability. We also presents, a new Risk Index integrating the results obtained and details from the Common Vulnerability Scoring System (CVSS).

In addition, a comprehensive study on the Vulnerability Space is presented discussing the likelihood of probable events in the probability sub-spaces of vulnerabilities.

Finally, an Extended Vulnerability Life Cycle model is presented and discussed in relation to States and Events in the Vulnerability Space that lays down a strong foundation for any future vulnerability related analytical research efforts.

1 Introduction

Present chapter is an introduction of the study conducted with the objective of developing a set of Statistical methodologies to be applied in the field of Cybersecurity. The study mainly focuses on contributing Cybersecurity field by addressing several important questions with respect to software Vulnerabilities and Vulnerability Life Cycle.

1.1 Research Problems and their Background

1.1.1 Research Problems

While trying to develop useful Stochastic models on Vulnerabilities, this study tries to address several important analytical problems as follows.

How to develop a complete Vulnerability Life Cycle Model that is applicable both analytically and in real world scenario?

How to observe and analyze the behavior of a vulnerability as a function of time and model such behavior?

How to develop a successful theoretical foundation to analyze any vulnerability in the Cyber space?

Addressing these problems in several aspects, we expect to present our methodologies, new models and their applicability. We shall discuss objectives we achieved and our contributions in relation to these questions in chapters to come.

1.1.2 Background and Literature review

In 2016, National Vulnerability Data Base (NVD)[1], Secunia Vulnerability review [2] and CVE details website recorded 6435 new vulnerabilities. In first five months of 2017, 5953 vulnerabilities are also added. 2016 Annual Vulnerability Report, issues on March 13, 2017 revealed that the absolute number of vulnerabilities detected was 17147. These were found on 2136 different applications from 246 vendors. About 19 percent of the vulnerabilities detected in 2016, had no Patch available at the time of the discloser. 18 percent of 17147 vulnerabilities were rated Highly Critical and, 0.5 as Extremely Critical by the Secunia Report.

On the other hand there were many cyber-attacks in various magnitudes showing how vulnerable the cybersecurity measures with respect to governments and industrial organizations including Banking and Finance sector, Homeland security, Retail commerce etc. These new developments force organizations including governments to consider these developments seriously and re-organize their defending organizations in a dynamic manner.

The core of cyber security related researches is the study, understand and make

efficient processes to eliminate possible damages from Vulnerabilities. A Vulnerability is a flaw in software which can be exploited with a security impact and unauthorized gain.

There are many studies in various aspects by researchers to understand vulnerabilities. Development in Information Technology and related industries including Hardware equipment and software applications during the last decade is an unprecedented land mark in human civilization. However, it is a reasonable observation that, in parallel to the development mentioned above, attention on defending techniques, strategies and deployment of resources were not sufficient.

It is rational to state here that, the applying of Scientific Methodology and integration of the knowledge in natural sciences such as Mathematics and Physics into IT environment for Cybersecurity objectives should be of the priority. Looking at vulnerabilities in Statistical perspective and analyze vulnerability data based on Statistical Methods and Philosophy would play an important role in security development decision making processes. Therefore, it is our objective to look for important contributions done in the area of Vulnerability analysis [3]-[14] by various scholars in the recent years and to put our effort in contributing for further developments.

Understanding the need for extensive and comprehensive research foundation, the US Department of Homeland Security in 2009 issued an in detailed report titled Cybersecurity Roadmap.

In this study, we expect to develop a set of analytical methods and tools using Statistical Methodology to be applied in Cybersecurity related decision making pro-

cesses.

Alhazmi, O. H. and Malaiya, Y. K. [3] in 2005 analyzed the vulnerability discovery process Modeling the Vulnerability Discovery Process. For the same analysis Malaiya and others used Weibull Distribution and proposed statistical approach as Vulnerability Discovery Model [14] and proposed models for Major Operating Systems [10]. In 2010, Joh, H. and Malaiya presented a framework for Software Risk Evaluation using Vulnerability Life Cycle and CVSS metrices.

One of the major and very important focus of cybersecurity study has been the study of Vulnerability Life Cycle. S. Frei in his doctoral dissertation discussed the concept of Vulnerability Life Cycle in several aspects [4]. His analysis on the Vulnerability Life Cycle was useful in understanding vulnerability behavior in different stages. S.Abraham and S.Nair, used Absorbing Markov model to develop a stochastic model for Security Quantification [7].

Vulnerabilities that is actively exploited by attackers before they are made known to the public and hence does not have a patch at the time of discloser are called Zero Day Vulnerabilities. Leyla Bilge and Tudor Dumitras presnted their Empirical Study on real world Zero Day attacks in 2012 [12]. Analysis on Advance Cyber attack Modelling by Jajodia, S. and Noel, S. discussed Attack Graphs, Attack Matrices and Attack Predictions and demonstrated a new approach for visualization and prediction of multi-step attack graphs [15], [16]. Study of Attack graphs and attack graph developing models is also a main aspect of Cybersecurity related studies. Mehta [17] and others in 2006 proposed a ranking Method for attack graphs in a computer

science perspective.

Attackers in general are referred for Black Hat hackers. But, there are White hat hackers also who are hacking into systems with good intentions to observe weaknesses and inform relevant parties. White Hat hackers could be internal employees of the organizations or free-lance security professionals. M. Zhao, J. Grossklags, and K. Chen have conducted an interesting Exploratory Study of White Hat Behaviors in a Web Vulnerability Disclosure Program in 2014 [18].

There are many such important contributions and efforts made in understanding vulnerabilities and attack behaviors. However, it should be noted that development of applicable and analytically sound Stochastic Analyses and processes seems to be crucial in Cybersecurity related studies.

Cyber security studies can lead in two main directions which are related in many ways. The first one is to analyze the weakness. The weakness to be analyzed is clearly the Vulnerability. Second direction is to analyze the human behavior in relation to cyber security. That is mainly the analyzing of Attacker and Attackers behavior in attack processes and cyber space. While considering both aspects, this study mainly focus on the first direction. That is the analyses on Vulnerability.

1.2 Introduction to Vulnerability Space

In chapter two, we introduce the concept of Vulnerability Space as a probability space. Chapter 2 lays down the foundation of deeper analysis for vulnerabilities and

Cybersecurity at large. We define Vulnerability Space taking the triple of Sample Space, Set of events and the Function of probability measure. Considering the entire vulnerability space based on the behavior of vulnerabilities, we observe and list a set of probable events in the vulnerability space. In the same chapter, we discuss the National Vulnerability Data base and Common Vulnerability Scoring System (CVSS) [5]. A basic introduction to the Vulnerability Life Cycle and States in the Vulnerability Space will be discussed. Contributions from the chapter are summarized at the end.

1.3 Stochastic Modelling of Vulnerability Life Cycle and Security Risk Evaluation

In chapter 3, we presents a developed model of Vulnerability Life Cycle graphically and then apply the Markov process that allows us to develop an analytical formation of a particular vulnerability. With this analytical matrix from, we apply the Morkovian [7], [20] iteration process to obtain probability of a particular vulnerability being exploited as a function of time. Methodology we used and results we obtained with examples will be discussed in details. In the same chapter, we will also develop and presents a new index of the Risk associated with the vulnerability.

Finally, in chapter 3, we develop a set of parametric models to predict the probability of vulnerability being exploited as a function of time. With these models, users can skip the analytical process of Markov approach and save time and effort yet have the similar level of accuracy. Developed models are tested and proven for their successfulness. Contributions from the chapter are summarized at the end.

1.4 Non Linear Stochastic Models for Predicting the Exploitability

Chapter 4 further analyze the exploitability and improve the modelling techniques and quality of developments in the previous chapter. In chapter 4, entire vulnerability date base is analyzed and resulted parameter estimates from properly considering the vulnerability data [6] base with over 75000 vulnerabilities are used instead of statistics from small samples as used in chapter 3. Set of better parameter estimates are therefore in use for the Markov approach in developing the Transition Probability Matrix in this chapter. With this sound improvement, we develop a set of successful Non-Linear Stochastic Models for predicting the Exploitability Probability for any vulnerability in three vulnerability levels, Low, Medium and High. Contributions from the chapter are summarized at the end.

1.5 A Comprehensive Analysis on Vulnerability Space

In chapter 5, a comprehensive analysis on the Vulnerability space that we defined in chapter 2 is presented. We use a new approach of Vulnerability Analysis with Venn diagrams. Chapter 5 identifies and presents almost all the probable events that could occur in the Vulnerability space. Likelihood and conditional probabilities are defined in relation to all these sates. This approach is proposed as a suitable foundation for any kind of cybersecurity analysis [13] and study in Statistical and Mathematical perspectives.

Some of the Danger Zones in the vulnerability space are also discussed in brief

in chapter 5. Using a set of Venn diagrams presenting and discussing one after the other, this chapter analyzes the relationship between States of Vulnerability space and Events that generates those states. Contributions from the chapter are summarized at the end.

1.6 Introduction to the Complete Vulnerability Life Cycle

Chapter 6 presents a new approach of a Complete Vulnerability Life Cycle [4]. This comprehensive approach considers all the states for a vulnerability. This model is a complex model for analytical purposes. However, it is very useful and applicable based on the behavior of a vulnerability or set of vulnerabilities of interest. This Vulnerability Life Cycle have many states and discusses all their relationships and behavior with respect to time. Finally, an analytical presentation of Markov approach for this complete Vulnerability Life Cycle is discussed.

2 Introduction to Vulnerability Space

2.1 Introduction

Study of Vulnerabilities [2] in numerous aspects have been the core of the scientific efforts in Cybersecurity and related disciplines. During the last decade, several important contributions towards vulnerability related studies have been done. In searching for better cyber security strategies, correctly identifying the Life Cycle [3], [4] of Vulnerabilities and their behavior throughout the life time is very important. However, when considering the recent rapid increase in the number of vulnerabilities and cyber-attacks which were never expected in such an abundance and magnitude, it is crucial to re-consider our understanding on the vulnerabilities and their behavior with respect to the time. Even though there are many important contributions in the modeling of the concept of Vulnerability Life Cycle, those models are not comprehensive enough to explain most of the real world aspects regarding vulnerabilities. Several important states of vulnerabilities are yet to be discussed and included in relevant analyses. As an example, Zero Day Vulnerabilities [12], [13] are not very well explained in many such models developed even though it is well known that zero day vulnerabilities represent a major threat to Cybersecurity.

Therefore, it is extremely important that we have a proper and comprehensive analytical model for the vulnerability life cycle that would present all the probable states of any vulnerability. However, before developing a comprehensive Vulnerability Life Cycle Model it is mandatory that we understand and list all the possibilities that a vulnerability could face. In other words, we need to identify all possible states of a vulnerability. To achieve this objective, in this study, we introduce the concept of Vulnerability Space, the probability space where all possible incidents that would occur are included. We discuss Vulnerability Space using a Venn Diagram Approach with all possible situations of vulnerabilities in the cyber world.

Once we explain the Vulnerability Space, we propose a comprehensive vulnerability life cycle model that considers all the probable events of any kind of vulnerability. Once we have such a model, we can then analyze the available data and observe the behavior of vulnerability states. In this study, we also analyze such important observation that we identify as Danger Zones in the Cybersecurity. There are cybersecurity related practices that we need to review seriously. Some practices in security efforts might actually make it worse the network computer systems. Initiations taken aiming at securing sometimes would actually result in more disastrous outcomes. Cybersecurity is right to be said as a Warfield. It is a combat between attackers and the defenders. Attackers are getting more powerful and using highly sophisticated exploit strategies as recent records revealed. Better understanding and analytical strategies on vulnerabilities will provide the defending personals and security systems much more formidable.

2.2 Vulnerability Space

2.2.1 Definition

We define the Vulnerability Space as the entire set of vulnerabilities that exist at any state at a given time. In other words, it is the universal set of all software vulnerabilities that are not being dead (patched) at a particular time. A vulnerability is known to go through several states from its birth to the death. Most commonly known states that vulnerabilities would go through are Discovery, Disclosure, Exploitation and Patch release.

In probability theory, Probability Space is defined as a measure space with total measure one. Probability space consists with three parts. Named, Sample space (Ω), σ -algebra of a subset of the sample space (F) and function in F taking values (probabilities) in [0,1] (P). Accordingly, we have to illustrate the relevance of this definition of probability space in the context of Vulnerability Space that we expect to define here. That is, Vulnerability Space should be defined consisting the triple (Ω, F, P) so that it can be considered as a Probability Space.

2.2.2 Sample Space of the Vulnerability Space (Ω_V)

Definition

Universal set of all the vulnerabilities in any possible state that is in the universal space of Cyber-space and related software systems is the sample space of the probability

space of vulnerabilities. In this context it should be noted that any vulnerability constitutes a random variable.

Therefore the sample space of vulnerabilities (Ω_V) contains the all possible outcomes in the Vulnerability Space that we discuss in details below. In one observing angle, with respect to vulnerabilities, we refer to these outcomes as states. It is quite impossible to list all the outcomes in most of the real world phenomena considering a probability spaces. However, observing the nature of various kinds of existing and discovered vulnerabilities and their behavior, we identify four main outcomes and their interactions. Those four main outcomes (states) are given below.

1. **Discovery**

   Discovery is the event of earliest observation or identification of a software vulnerability. A vulnerability could be discovered by an attacker, defender or any other observer such as a software developer, system administrator etc. Whoever the person or whatever the job the discoverer carries out, we define the discovered party based on the intention. If the person discovers the vulnerability at first, has the intention to exploit the vulnerability, we consider it was discovered by an attacker. If the vulnerability was discovered by a person with the intention of defending the software integrity and security, or intend to help system administrators, software developers and vendors, then we consider the vulnerability was discovered by a defender.

2. **Disclosure**

   Disclosure is the earliest public discloser of a vulnerability. After discovery, a vulnerability could be disclosed to public by vendors with the intention of alarming to protect the users systems. Disclosure of vulnerability is very critical activity with many complex outcomes.

3. **Exploitation**

   Exploitation is the act that, an attacker who creates an exploit (a software tool that is developed to manipulate the vulnerability and execute the exploitation act) using it successfully.

4. **Patch release**

   Patch release is the act of releasing (making available) a software patch for the vulnerability. Once the patch is released, users can install the patch so that the vulnerability is fixed completely. Once the patch is installed in a system, the vulnerability is said to be treated in that system. But, releasing of a patch not necessarily constitute the death or the end of a vulnerability. For a vulnerability to be considered as Dead, released patches should be installed in all the systems that the software is installed. Even though it is theoretically correct to say so, practically, we may consider a vulnerability is dead, as long as the patches are installed by almost all the users so that the probability of a major threats causing that vulnerability is negligible.

2.2.3 Random Process of Generating Vulnerabilities

To better understand these states and their behavior, the concept of Vulnerability life cycle has been used by many researchers. This study also uses the concept of vulnerability Life cycle throughout all the next chapters in various aspects. An introduction to vulnerability Life cycle will be given in section 2.4 in this chapter. However, at this point it should be noted that, by using the term Life Cycle we do not imply a process of re-production of vulnerabilities that continues as a circle. The term Vulnerability Life Cycle, only means the series of stages through which a vulnerability passes from the beginning of its life (existence) until its death.

Vulnerabilities exist in computer software systems. When we use the term Computer Software System, we include the whole system including Hardware, Software and the power supply. However, the interface that vulnerabilities are defined is mainly the software. A software developer, or a team uses programming languages and other tools in developing a software. Such software would be installed on a hardware system or a hardware system with a system software. Programs, Networking protocols and relevant hardware will make the connectivity among such software.

Since software is human made and developed using codes there is no guarantee that they are devoid of flaws. Innumerable such code flaws could exist unknown to anyone. But, our discussion here starts with an observer. The observer could be an Attacker who intends to find a software flaw and exploit it or a system administrator (defender) who intends to protect the system being hacked (exploited). Defining of the "Birth" of a vulnerability is uncertain. Conventionally, we can accept the exis-

tence of vulnerabilities prior to the observation of such a flaw. Then, the "Birth" of any vulnerability is should be the creation of the software itself. Practically, if we take this classical approach, the "Birth" of a vulnerability could be considered as the moment(date) that the software is released for users. In such a classical approach where we accept the existence of software vulnerabilities independent of the observer, likelihood of the other events should be defined based on that approach too. As an example, inferences regarding events such as discovering a vulnerability that was unknown previously should be defined based on the assumption that there are unknown innumerable number of software vulnerabilities are there. However, such an approach again in our pointof view possess an inherent doubts. In such classical approach where we accept the "absolute existence" of vulnerabilities (independent of the observer) could make it almost impossible to justify our statistical inferences.

There is another approach that we can take here defining the "Birth" of a vulnerability. In this new approach "unobserved existence" of flaws in software is not our consideration. Therefore, we call a vulnerability is Born at the very first time that someone observed that particular flaw in the software. Thus, the sample space or universal set of the vulnerability space consists with the vulnerabilities that has been "observed" by any human being at a particular time.

However, this kind of radical approach must be reviewed thoroughly for its consequences and theoretical strength by Mathematicians and Statisticians. In this study we only take the classical approach defining the "Birth" of a vulnerability at the time of the development or the release of the software itself. But, it is our expec-

tation to conduct our future research efforts considering this new approach of defining of vulnerability dependent to the "observer".

It should also be noted here that the approach we take in defining the "Birth" does not affect in understanding the events that could occur in the "Vulnerability Space" substantially. That is, all the events that would be defined in the new "observational approach" could be defined in the classical approach also, but with a different interpretation on the likelihood.

In next two chapters we use a basic model of Vulnerability Life Cycle to develop and introduce a set of useful statistical models in Cybersecurity. In chapter five we further analyze the Vulnerability Life Cycle and presents a comprehensive analysis.

2.3 Common Vulnerability Scoring System (CVSS)

Our study of vulnerability space and its events are based on the available data. THe most commonly used and best available data source we have is the National Vulnerability Database which is based on the Common Vulnerability Scoring System (CVSS)[5], [6]. Therefore, in this section we will discuss about this data base and the information we get from the database.

Common Vulnerability Scoring System (CVSS) is a free and open industry standard for assessing the severity of computer system security vulnerabilities. It is under the custodianship of the Forum of Incident Response and Security Teams (FIRST) [1]. CVSS is composed of three metric groups, Base, Temporal, and Environmental, each consisting of a set of metrics. It attempts to establish a measure

of how much concern a vulnerability warrants, compared to other vulnerabilities, so efforts can be prioritized. The scores are based on a series of measurements (called metrics) based on expert assessment. The scores range from 0 to 10. Vulnerabilities with a base score in the range 7.0-10.0 are High, those in the range 4.0-6.9 as Medium, and 0-3.9 as Low. Figure 2.1 and 2.2 below give a schematic presentation of the Common Vulnerability Scoring System (CVSS) which is the basis of the metric calculation model and the temporal and environmental matrices calculation model, respectively.

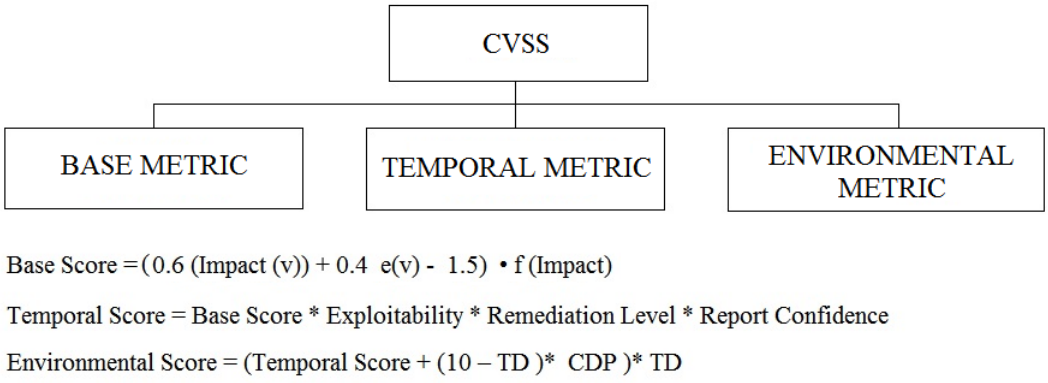

Figure 2.1: Common Vulnerability Scoring System

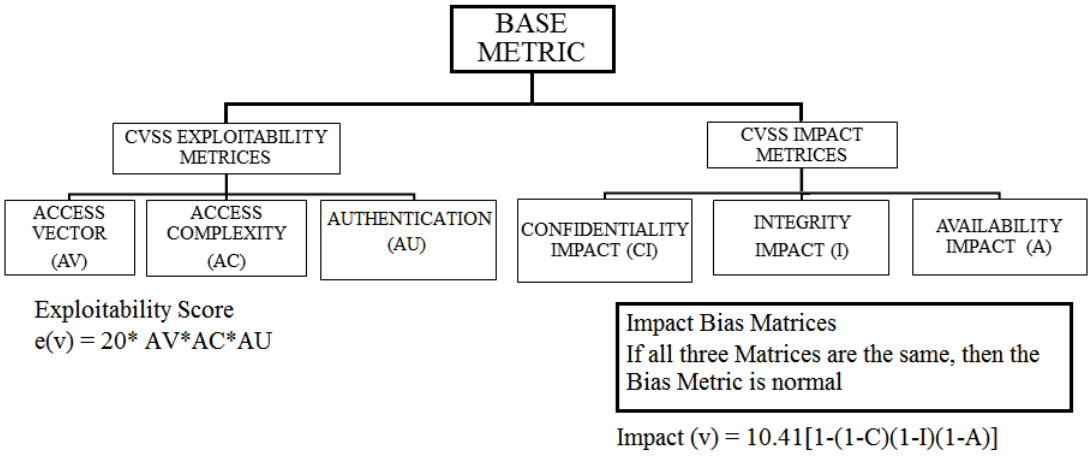

Figure 2.2: Common Vulnerability Scoring System- Base Metric Calculation Model

2.3.1 Base Metric

Base Metric is derive from two sub metrics,Exploitability metric and Impact metric.Calculation methodology is given in Appendix A and B.

$$BaseScore = (0.6 * Impact + 0.4 * e(v) - 1.5) * f(Impact),$$

$$e(v) = 20 * AV * AC * AU, Impact(v) = 10.41(1 - (1 - C)(1 - I)(1 - A)),$$

$$f(Impact) = \begin{cases} 0, & \text{if } Impact(v) = 0 \\ 1.176, & \text{Otherwise} \end{cases}$$

Access Vector (AV)

This measures whether a vulnerability is exploitable locally or remotely. Local: The vulnerability is only exploitable locally Remote: The vulnerability is exploitable

remotely (and possibly locally as well)

Access Complexity (AC)

This measures the complexity of attack required to exploit the vulnerability once an attacker has access to the target system. High: Specialized access conditions exist such as specific window of time (a race condition), specific circumstance (non-default configurations) or victim interaction such as tainted e-mail attachment. Low: Specialized access conditions or extenuating circumstances do not exist. In other words, it is always exploitable. This is the most common case

Authentication (AW)

This measures whether or not an attacker needs to be authenticated to the target system in order to exploit the vulnerability. Required: Authentication is required to access and exploit the vulnerability. Not Required: Authentication is not required to access or exploit the vulnerability.

Confidentiality Impact (C)

Confidentiality Impact measures the impact on Confidentiality of a successful exploit of the vulnerability on the target system. None: No impact on confidentiality. Partial: There is consider able informational disclosure. Complete: A total compromise of critical system information.

Integrity impact (I)

Integrity Impact measures the impact on Integrity of a successful exploit of the vulnerability on the target system. None: No impact on integrity. Partial: Considerable breach in integrity. Complete: A total compromise of system integrity.

Availability impact (A)

Availability Impact measures the impact on Availability of a successful exploit of the vulnerability on the target system. None: No impact on availability Partial: Considerable lag in or in eruptions in resource availability Complete: Total shutdown of the affected resource

2.3.2 Temporal Metric

The Temporal metrics measure the current state of exploit techniques or code availability, the existence of any patches or workarounds, or the confidence that one has in the description of a vulnerability.

2.3.3 Environmental Metric

These metrics enable the analyst to customize the CVSS score depending on the importance of the affected IT asset to a users organization, measured in terms of complementary/alternative security controls in place, Confidentiality, Integrity, and

Availability. The metrics are the modified equivalent of base metrics and are assigned metrics value based on the component placement in organization infrastructure.

Collateral Damage Potential (CDP)

This metric measures the potential for loss of life or physical assets through damage or theft of property or equipment. The metric may also measure economic loss of productivity or revenue. The possible values for this metric are listed in Appendix A. Naturally, the greater the damage potential, the higher the vulnerability score.

Target Distribution (TD)

This metric measures the proportion of vulnerable systems. It is meant as an environment-specific indicator in order to approximate the percentage of systems that could be affected by the vulnerability. The possible values for this metric are listed in Appendix A. The greater the proportion of vulnerable systems, the higher the score.

2.4 Vulnerabilities Life cycle

The Life Cycle of a Vulnerability [3] can be introduced with different stages that a vulnerability passes through. We shall discuss specific stages that are commonly identified in a given situation. As we identified in the section 2.2 "states" in the Vulnerability Space are corresponding to the these "stages" of Vulnerability Life Cycle. That is, collection of all the vulnerabilities in a particular "state (or stage)" of their life cycles constitute the likelihood of the same state in the vulnerability space.

Commonly identified stages are Birth (Pre-discovery Stage), Discovery, Disclosure, Availability for Patching and Availability for Exploiting that are directly relevant to the "main events" we identified in the vulnerability space.

2.4.1 Birth (Pre-Discovery)

The birth of a vulnerability occurs at the development of a software, mostly due to a weakness or a mistake in coding of the software. At this stage the vulnerability is not yet discovered or exploited. In a well-developed software package where its reliability has been identified, one can identify the probability of the birth of the problem.

2.4.2 Discovery

Vulnerability is said to be discovered once someone identifies the flaw in the software. It is possible that the vulnerability is discovered by the system developers themselves, skilled legitimate users or by the attackers also. If the vulnerability is discovered internally or by white hackers, (who are making breaking attempts on a system to identify the flaws and vulnerabilities with good intentions of helping them to be patched so that the system security is strengthened) it will be notified to be fixed as soon as possible. But, if a black hacker discovers a vulnerability it is possible that he or she will try to exploit it, or sell in the black market or distribute it among hackers to be exploited.

It should be noted here that while vulnerabilities could actually exist prior to

the discovery, until it is discovered, it is not a potential security risk. "Time of the discovery" is the earliest time that a vulnerability is identified. In a vulnerability life cycle the "time of discovery" is an important and critical event. Exact discovery time might not be published or disclosed to the public due to the other risks that could be associated with a vulnerability. However, in general after the "disclosure" of a vulnerability, public may know the time of discovery subject to security risk review.

We would like to mention here that in developing our statistical model, we consider only "pre-exploit discovery". There are rare chances that a discovery of a vulnerability could occur after it is actually exploited. As an example, an attacker could run an exploit attempt aiming to exploit a particular vulnerability. But, the exploit actually breaks into the system through another unidentified or undiscovered vulnerability instead of expected vulnerability at that time. Such rare occurrences are not taken into account in our our present study.

2.4.3 Disclosure

Once a vulnerability is discovered, it is subject to be disclosed. Disclosure could take place in different ways based on the system design, authentication and who discovered it. However, "disclosure" in widely accepted form in the information security means the event that a particular vulnerability is made known to public through relevant and appropriate channels. Definition for the disclosure of vulnerability is however presented differently by different individuals.

In general, public disclosure of a vulnerability is based on several principles.

The availability of access to the vulnerability information for the public is one such important principle. Another such important principle is validity of information. Validity of information principle is to ensure the users ability to use those information, assess the risk and take security measures. Also, the independence of information channels is also considered to be important to avoid any bias and interferences from organizational bodies including the vendor.

2.4.4 Scripting (Exploiting) and Exploit Availability

A Vulnerability enters to the stage of exploit availability from the earliest time that an exploit program of code is available. Once the exploits are available even low skilled crackers (or in other words a black hat hacker) could be capable of exploiting the vulnerability. As we mentioned earlier, there are some occurrences that the exploit could happen even before the vulnerability is discovered. However in the present study we consider the modeling of Vulnerability Life Cycles with exploit availability occurs only after the discovery.

2.4.5 Patch Availability and Death: (Patched)

Patch is a software solution that the vendor or developer release to provide necessary protection from possible exploits of the vulnerability. Patch will act against possible exploit codes or attacking attempts for a vulnerability and protect the system and ensure the integrity. The vulnerability dies when one applies a security patch to all

the vulnerable systems.

When a White Hat Researcher discovers a vulnerability, the next transition is likely to be the internal disclosure leading to patch development. On the other hand, if a Black Hat Hacker discovers a vulnerability, the next transition could be an exploit or internal disclosure to his underground community. Some active black hats might develop scripts that exploit the vulnerability.

2.5 Events of Vulnerability Space

In this section, we discuss a set of identified events occurs in the vulnerability space based on the main "states" in the Vulnerability Life Cycle that we introduced earlier. The first event is obviously the the random process that generates the vulnerabilities. In other words, the "Birth" of a vulnerability. Then we have the event of discovering a vulnerability. That is the event of first human observation of a software flaw. For the simplicity lets consider the the vulnerability space consisting all the discovered vulnerabilities at a particular time.

Once an attacker or a defender discovers a vulnerability there are several important incidents that are probable to occur. Those important incidents are called events in the vulnerability space. Occurrence of these probable events are complex than we would in general imagine. For better understanding of these probable events, lets discuss the events that are probable in the vulnerability space. We identify 12 distinct important events that can occur after the first event that a vulnerability is discovered in the space of the vulnerabilities.

2.5.1 List of probable events in the subspace of discovered vulnerabilities

1. Event that a discovered vulnerability is disclosed (D) to the public before it is exploited (E) or patch is released (P), $(D \cap (E \cup P)^c)$.

2. Event that a discovered vulnerability is disclosed (D) to the public after a patch is released (P) before it is exploited (E), $(D \cap (E^{'} \cap P))$.

3. Event that a discovered vulnerability is disclosed (D) to the public after it is exploited (E) but before the patch is released (P), $(D \cap (E \cap p^c))$.

4. Event that a discovered vulnerability is disclosed (D) to the public after it is both exploited (E) and released with a patch (P) developed, $(D \cap (E \cap P))$.

5. Event that a discovered vulnerability is released with a patch (P) before it is disclosed (D) or exploited (E), $(P \cap (D \cup E)^c)$.

6. Event that a discovered vulnerability is released with a patch (P) before it is exploited (E) but after it is disclosed (D), $(P \cap (D \cap E^c))$.

7. Event that a discovered vulnerability is released with a patch (P) before it is disclosed (D) but after it is exploited (E), $(P \cap (E \cap D^c))$.

8. Event that a discovered vulnerability is released with a patch (P) after it is disclosed (D) and exploited (E), $(P \cap (E \cap D))$.

9. Event that a discovered vulnerability is exploited (E) before it is disclosed (D) or patch is released (P), $(E \cap (D \cup P)^c)$.

10. Event that a discovered vulnerability is exploited (E) before it is disclosed (D) but after patch is released (P), $(E \cap (D^{'} \cap P))$.

11. Event that a discovered vulnerability is exploited (E) before the patch is released (P) but after disclosed (D), $(E \cap (D \cap P^{c}))$.

12. Event that a discovered vulnerability is exploited (E) after it is disclosed (D) and patch is released (P), $(E \cap (D \cap P))$.

All events mentioned above make a vulnerability to move from one state into another particular state in the Vulnerability Space. This is indeed a parallel observation of a vulnerability in its life cycle [10],[14]. A move of a vulnerability from one state to another state in its life cycle contribute to a variation in the "Likelihood" of the corresponding state in the Vulnerability Space.

Some of these events are indeed very rare and probability of such an event to occur would be very small. But, being less probable does not make the event unimportant in cybersecurity field by no means. One such rare event could lead to a major flaw in a network system which might make way for an attack and incur a huge financial and data loss. Therefore, it is crucial to understand all these event and causes that drives these events.

In chapters five and six we will further discuss and analyze Vulnerability Space and relevant events occur in the this space in a probabilistic point of view.

2.6 Contributions

In this chapter, we introduce and defined "Vulnerability Space" as a probability space. This definition constitutes the frame work and lay down the foundation of the study of vulnerabilities in mathematical and statistical points of view. We further identified the main "States" and "Events" of the Vulnerability life cycle which simultaneously creates probable "states" in the Vulnerability Space.

3 Stochastic Modeling of Vulnerability Life Cycle and Security Risk Evaluation.

3.1 Introduction

In this chapter we propose a method using Markov chain [7], [8] to understand the Vulnerability Life Cycle [9] and analyze it to observe the Security Risk behavior [11]. Any identified vulnerability, is hazardous to a security system and makes the system susceptible to be exploited until it is well patched. Therefore, we believe it is very important to know how to deal with a vulnerability behavior throughout its different stages. "Vulnerability Life Cycle"[3], [4] would certainly help us to better understand the vulnerability and its behavior in a security system with respect to time. There are a number of ways to present the life cycle of a particular vulnerability. However, all these different introductions have several important stages in common. The level of the risk associated with different stages of a vulnerability should be different indeed and need to be estimated.

However, measuring of such a "risk factor"[10], [11] and obtaining a probabilistic estimate is certainly a challenge given the lack of data resources. If we have a method developed to measure the risk level associated with a particular vulnerability

at a certain time or stage, it will help the defenders and organizations to act accordingly with well-defined priorities. Then the users and organizations can make sure adequate attention, resources and security intellects are employed to address such a risk and proper fixing steps are taken before it is exploited. One of the main objectives we have is to obtain a statistical model that can give us the probability of a vulnerability being exploited or patched at a given time. In this study we use the well-known theory of Markov Chain Process [20] to develop such a model.

3.2 Vulnerability and Vulnerability Life Cycle

In this section we will further explain basic concepts of Vulnerability, Vulnerability Life Cycle that we discussed in chapter 2 and related technical terms to make it easier to understand later sections.

Microsoft Security Response Center (MSRC) defines the term Vulnerability as follows.

"A security vulnerability is a weakness in a product that could allow an attacker to compromise the integrity, availability, or confidentiality of that product."

We understand that, a vulnerability could be derived by investigating the various weaknesses of an implemented security system. With a weakness in a custom design software, a vulnerability can come to effect in authentication protocols, software reliability and system process, Hardware management and Networking among others.

3.2.1 Stages of Vulnerability Life Cycle

The Life Cycle of a Vulnerability [3], [21] can be introduced with different stages that a vulnerability passes through. We shall discuss specific stages that are commonly identified in a given situation. Commonly identified stages are involved with the events such as the Birth (Pre-discovery Stage), Discovery, Disclosure, Availability for Patching and Availability for Exploiting [4].

Figure 1, illustrates the life cycle of vulnerability showing key stages to be discussed.

3.2.2 Birth (Pre-Discovery)

The birth of a vulnerability occurs at the development of a software, mostly due to a weakness or a mistake in coding of the software. At this stage the vulnerability is not yet discovered or exploited. In a well-developed software package where its reliability has been identified, one could be able to estimate the probability of the birth of a vulnerability.

3.2.3 Discovery

Vulnerability is said to be discovered once someone identifies the flaw in the software. It is possible that the vulnerability is discovered by the system developers themselves, skilled legitimate users or by the attackers also. If the vulnerability is discovered internally or by white hackers, (who are making breaking attempts on a system to identify

the flaws and vulnerabilities with good intentions of helping them to be patched so that the system security is strengthened) it will be notified to be fixed as soon as possible. But, if a black hacker discovers a vulnerability it is possible that he or she will try to exploit it, or sell in the black market or distribute it among hackers to be exploited.

It should be noted here that while vulnerabilities could actually exist prior to the discovery, until it is discovered, it is not a potential security risk. "Time of the discovery" is the earliest time that a vulnerability is identified. In a vulnerability life cycle the "time of discovery" is an important and critical event. Exact discovery time might not be published or disclosed to the public due to the other risks that could be associated with a vulnerability. However, in general after the "disclosure" of a vulnerability, public may know the time of discovery subject to security risk review.

We would like to mention here that in developing our statistical model, we consider only "pre-exploit discovery". There are rare chances that a discovery of a vulnerability could occur after it is actually exploited. As an example, an attacker could run an exploit attempt aiming for a particular vulnerability but, the exploit instead break the intended system through another unidentified or undiscovered vulnerability at that time. While intending to address and incorporate such rare occurrences in our future research, in the present study we will consider vulnerabilities that we discovered before being exploited.

Disclosure

Once a vulnerability is discovered, it is subject to be disclosed. Disclosure could take place in different ways based on the system design, authentication and who discovered it. However, "disclosure" in widely accepted form in the information security means the event that a particular vulnerability is made known to public through relevant and appropriate channels. Definition for the disclosure of vulnerability is however presented differently by different individuals.

In general, public disclosure of a vulnerability is based on several principles. The "availability of access" to the vulnerability information for the public is one such important principle. Another such important principle is "validity of information". Validity of information principle is to ensure the user's ability to use those information, assess the risk and take security measures. Also, the "independence of information channels" is also considered to be important to avoid any bias and interference from organizational bodies including the vendor.

3.2.4 Scripting (Exploiting) and Exploit Availability

A Vulnerability enters to the stage of "exploit availability" from the earliest time that an exploit program of code is available. Once the exploits are available even low skilled crackers (or in other words a black hat hacker) could be capable of exploiting the vulnerability. As we mentioned earlier, there are some occurrences that the exploit could happen even before the vulnerability is discovered. However in the present study

we consider the modeling of Vulnerability Life Cycles with exploit availability occurs only after the discovery.

3.2.5 Patch Availability and Death: (Patched)

Patch is a software solution that the vendor or developer release to provide necessary protection from possible exploits of the vulnerability. Patch will act against possible exploit codes or attacking attempts for a vulnerability and protect the system and ensure the integrity. The vulnerability dies when one applies a security patch to all the vulnerable systems.

When a White Hat Researcher [18] discovers a vulnerability, the next transition is likely to be the internal disclosure leading to patch development. On the other hand, if a Black Hat Hacker discovers a vulnerability, the next transition could be an exploit or internal disclosure to his underground community [27]. Some active black hats might develop scripts that exploit the vulnerability. Figure 3.1, below illustrates the process of the above discussion.

3.3 Methodology

3.3.1 Markov Chain and Transition Probabilities

A discrete type stochastic process [20] $X = \{X_N, N \geq 0\}$ is called a Markov chain [20], [24] if for any sequence $\{X_0, X_1, ..., X_N\}$ of states, the next state depends only on

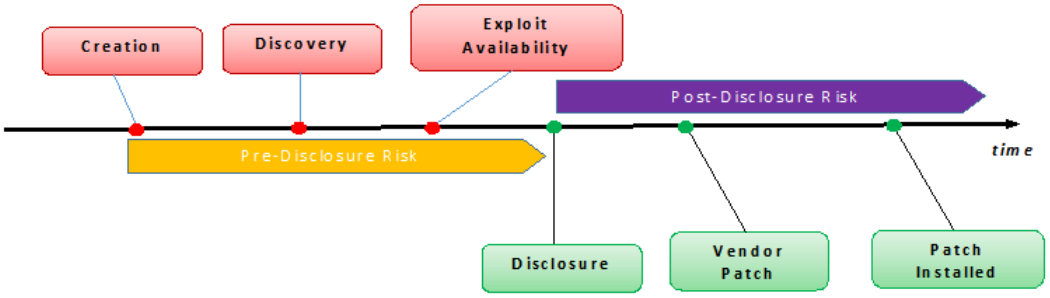

Figure 3.1: The Life Cycle of Vulnerability [4]

the current state and not on the sequence of events that preceded it, which is called the Markov property. Mathematically we can write this as follows.

$$P(X_N = j|X_0 = i_0, X_1 = i_1, ..., X_{N-2} = i_{N-2}, X_{N-1} = i) = P(X_N = j|X_{N-1} = i) \tag{3.3.1}$$

We will also make the assumption that the transition probabilities $P(X_N = j|X_0 = i_0, X_1 = i_1, ..., X_{N-2} = i_{N-2}, X_{N-1} = i)$ do not depend on time. This is called time homogeneity. The transition probabilities $(P_{i,j})$ for Markov chain can be defined as follows.

$$(P_{i,j}) = P(X_N = j|X_{N-1} = i) \tag{3.3.2}$$

The transition matrix P of the Markov chain is the NxN matrix whose (i, j) entry

$P_{i,j}$ satisfied the following properties.

$$0 \leq P_{i,j} \leq 1, 1 \leq i, j \leq N$$

$$\sum_{j=1}^{N} P_{i,j} = 1, 1 \leq i, j \leq N$$

Any matrix satisfying the above two equations is the transition matrix for a Markov chain. To simulate a Markov chain, we need its stochastic matrix P and an initial probability distribution π_0.

3.3.2 Transient States

Let P be the transition matrix [8],[15] & [16] for Markov chain X_n. A state i is called transient state if with probability 1 the chain visits i only a finite number of times. Let Q be the sub matrix of P which includes only the rows and columns for the transient states. The transition matrix for an absorbing Markov chain [24], [25] has the following canonical form.

$$P = \begin{bmatrix} Q & R \\ 0 & I \end{bmatrix} \tag{3.3.3}$$

Here P is the transition matrix, Q is the matrix of transient states, R is the matrix of absorbing states and I is the identity matrix.

The matrix P represents the transition probability matrix of the absorbing Markov chain. In an absorbing Markov chain the probability that the chain will be

absorbed is always 1. Hence, we have,

$$Q^n \to \infty, n \to \infty.$$

Thus, is it implies that all the eigenvalues of Q have absolute values strictly less than 1. Hence, I-Q is an invertible matrix [20] and there is no problem in defining the matrix

$$M = (I - Q)^{-1} = I + Q + Q^2 + Q^3 + ...$$

This matrix is called the fundamental matrix of P. Let i be a transient state and consider Y_i, the total number of visits to i. Then we can show that the expected number of visits to i starting at j is given by M_{ij}, the (i, j) entry of the matrix M. Therefore, if we want to compute the expected number of steps until the chain enters a recurrent class, assuming starting at state j, we need only sum M_{ij} over all transient states i.

3.4 Vulnerability Life Cycle Analysis Method

In this section we discuss the application of the methodology presented in the previous section. That is the application of the Markov process into the vulnerability life cycle graph [11], [25] and developing of the "Transition Probability Matrix" [20], [24].

3.4.1 Vulnerability Life Cycle Graph

The core component of the Vulnerability Life Cycle Analysis method we propose here is the Life Cycle Graph [17], [22], [23] & [25]. When we draw a Life Cycle Graph for a given vulnerability it has several nodes which represent the Vulnerability Life Cycle stages. We can assign a possible probability to reach each state by examining the properties of a specific vulnerability. Also, a Life Cycle Graph has two absorbing states [25] that are named "Patched state" and "Exploited state". Therefore, this allows us to model the Life Cycle Graph as an absorbing Markov chain.

The Markov Model Approach to Vulnerability Life Cycle we develop is given in the Figure 3.2, below. In this figure, we present a Markov approach of Vulnerability Life Cycle with five states. It should be noted that the states three and five are absorbing states of this Life Cycle Graph as there are no out flaws from those states.

We define, λ_i = the probability of transferring state i to state j.

In actual situations the probability of discovering a vulnerability can be assumed very small. Therefore, for λ_1 we can assign a small value. Then we assigned probabilities to $\lambda_2, \lambda_3, \lambda_4, \lambda_5$, accordingly.

Using these transition probabilities we can derive the absorbing transition probability matrix for a Vulnerability Life Cycle, which follows the properties defined under Markov Chain Transformation Probability Method.

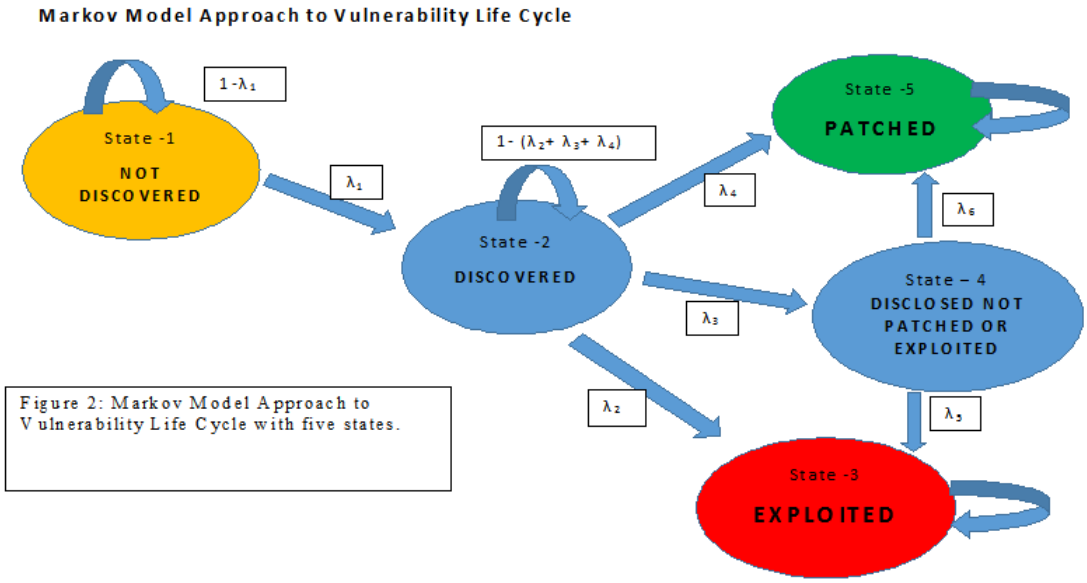

Figure 3.2: Markov Model Approach to Vulnerability Life Cycle

3.4.2 Transition Matrix for Vulnerability Life Cycle

Using the methodology on the vulnerability life cycle graph we can now write the transition probability matrix for vulnerability life cycle as follows.

$$P = \begin{bmatrix} 1-\lambda_1 & \lambda_1 & 0 & 0 & 0 \\ 0 & 1-(\lambda_2+\lambda_3+\lambda_4) & \lambda_2 & \lambda_3 & \lambda_4 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.4.4}$$

Where,

$P_t(t)$- Probability that the system is in state i at time$t$.

For $t = 0$ we have

$P_1(0)$= 1, Probability that the system is in State 1 at the beginning ($t = 0$).

$$P_2(0) = 0, P_3(0) = 0, P_4(0) = 0, P_5(0) = 0.$$

Therefore, the initial probability can be given as $\begin{bmatrix} 1 & 0 & 0 & 0 & 0 \end{bmatrix}$, that is, the probabilities of each state of the Vulnerability Life Cycle initially. It is clear that, the "State 1" (Not Discovered) with probability of one represents that at the initial time (for $t = 0$), the Vulnerability is not yet been discovered and therefore the probabilities for all others stages are zero.

We can assign some reasonable values to λ_i' s and create the transformation matrix P as follows. As an example, if we consider a time intervals of days, for probabilities of each stages to a specific vulnerability can be derived using the Markov process as follows.

For $t = 0$, we have

$$P^{(0)} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \end{bmatrix},$$

For $t = 1$, results in

$$P^{(1)} = P^{(0)}P,$$

For $t = 2$, we can write

$$P^{(2)} = P^{(0)}P^{(2)},$$

And thus, for $= n$, we have

$$P^{(n)} = P^{(0)}P^{(n)}.$$

Using this method, we can find the pattern of probability that is changing with time and is related to each "state" and then to work on finding the statistical model that can fit the vulnerability life cycle.

For $\lambda_1 = 0.1$, $\lambda_2 = 0.2$, $\lambda_3 = 0.3$, $\lambda_4 = 0.4$, $\lambda_5 = 0.4$, $\lambda_6 = 0.6$ transition probability matrix can be written as follows:

$$P = \begin{bmatrix} 0.9 & 0.1 & 0 & 0 & 0 \\ 0 & 0.1 & .2 & 0.3 & 0.4 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0.4 & 0 & 0.6 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}. \tag{3.4.5}$$

As we execute this algorithm, the stationarity was reached (considering to 4 decimal digits) at $t = 107$, that is at $t = 107$, we can find the minimum number of steps so that the vulnerability reaches its absorbing states [29] and the resulting vector of probabilities for each of the states is obtained as follows. As the row vector presents, the transition probabilities are completely absorbed into the two absorbing states which gives the probability of the vulnerability that is being exploited and the probability of the vulnerability will be patched. All other states have reached the probability of zero. That is,

$$P^{(n)} = P^{(0)}P^{(n)} = \begin{bmatrix} 0 & 0 & 0.3556 & 0 & 0.6444 \end{bmatrix}$$

The following figures illustrates the behavior of the probabilities as a function of time with respect to the different states. For states one, three, four and five taking initial probabilities as mentioned above, the behavior as a function of time is graphed. For states one and three the probability of "Not-discovered" and "Disclosed not patched" respectively, decreases with respect to time and approach zero eventually.

Figure 3.3, 3.4, 3.5 and 3.6 below presents the behavior of the probability of each state based on the initial probabilities we assigned. It is clear that the probability of being in the state 1 decreases and approach zero eventually. This indicates that the probability of a vulnerability being "Not-discovered" over the time is decreasing and eventually reaches zero at the time of the "discovery" (Figure 3.3). Once a vulnerability is discovered, the probability of being "Exploited" over time indeed increases. And as the system security activities also will immediately take place, the probability of being "Patched" also increases. This behavior is presented in Figures 3.4 and 3.6, respectively. There is also a time gap between the disclosure and patching of the vulnerability. Initially, the probability of the vulnerability being "Disclosed not patched" will rise for a very short period of time then will decrease eventually as this is not an absorbing state in the life cycle.

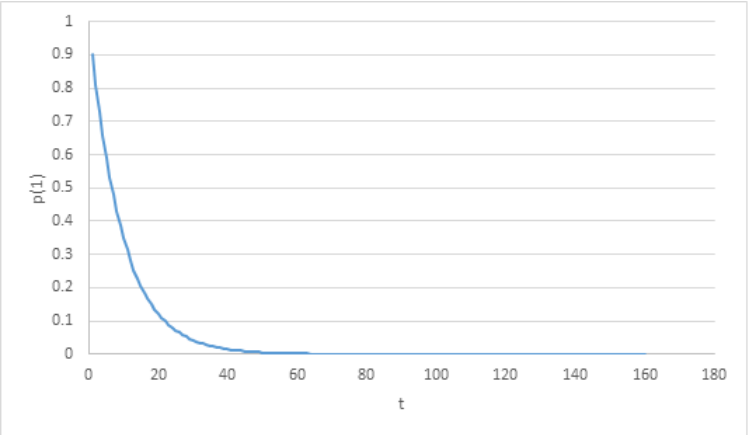

Figure 3.3: Probability of being Not discovered

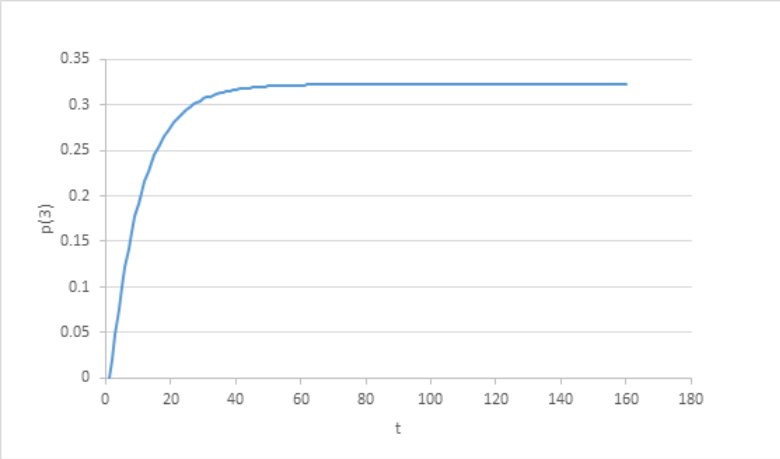

Figure 3.4: Probability of being Exploited

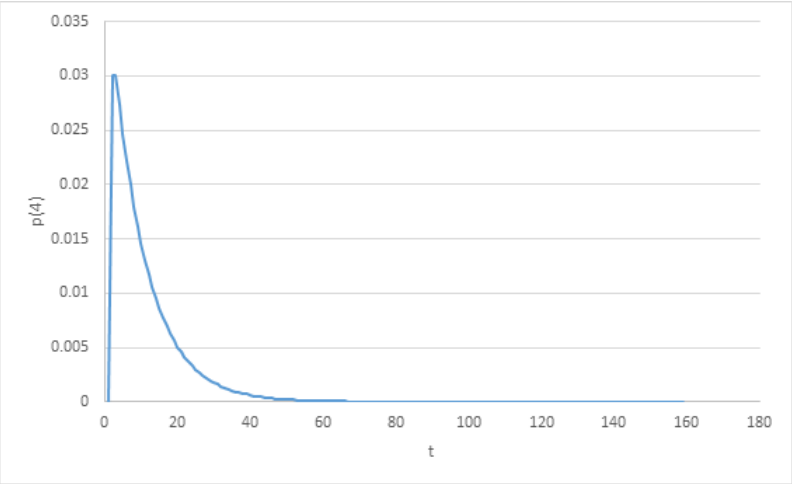

Figure 3.5: Probability of being Disclosed -Not Patched

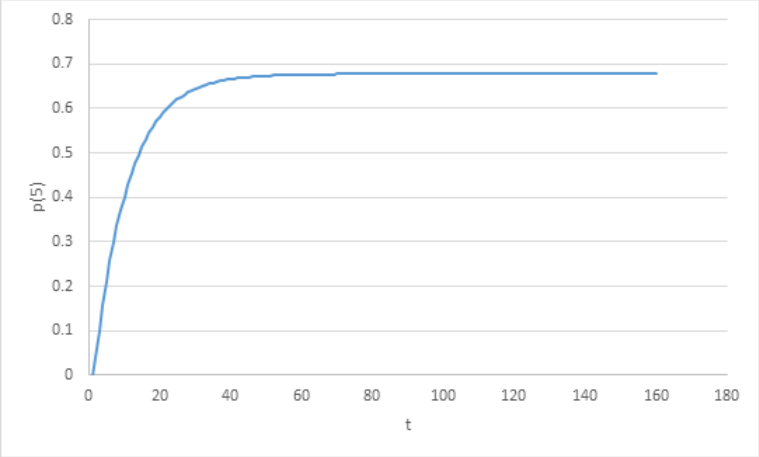

Figure 3.6: Probability of being Patched

For a better understanding, comparison and to have a more generalize observation we proceed to check the behavior of these probabilities over the time with different probability assigned values. We change λ_1 values and compare the probability changes in each state with time. The following graphs, illustrate the behavior of each state for λ_1= 0.1, 0.2, 0.4, 0.5 and 0.7. Figure 3.7, 3.8, 3.9 and 3.10 represent those

behaviors graphically. Each graph presents the behavior of the probability of being in that "state" of the life cycle over time. It is interesting to observe that the initial probability that we assign for $\lambda 1$ did not really affect much on the behavior of the probability over time.

However, it is important to note that a vulnerability with a higher initial probability of being discovered will go to stationarity faster than to those with a lower initial probability of being discovered. This is observable from the graphs labeled Probability of being Exploited as a function of time and Probability of being Patched as a function of time in Figures Figure 3.8 and 3.10 respectively.

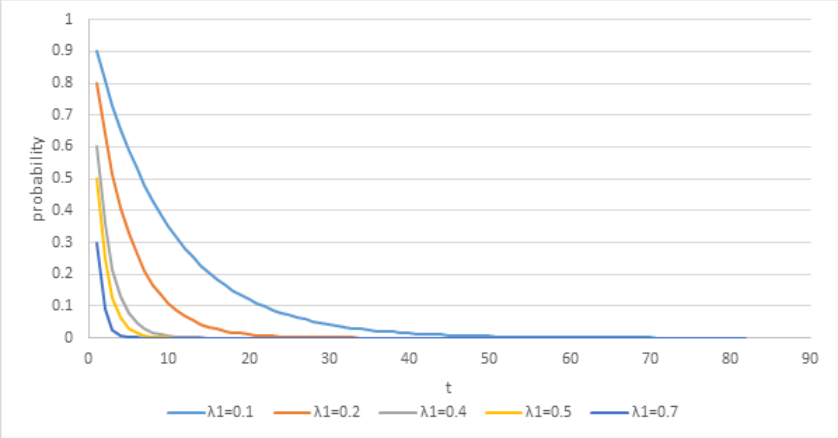

Figure 3.7: Probability of being Patched

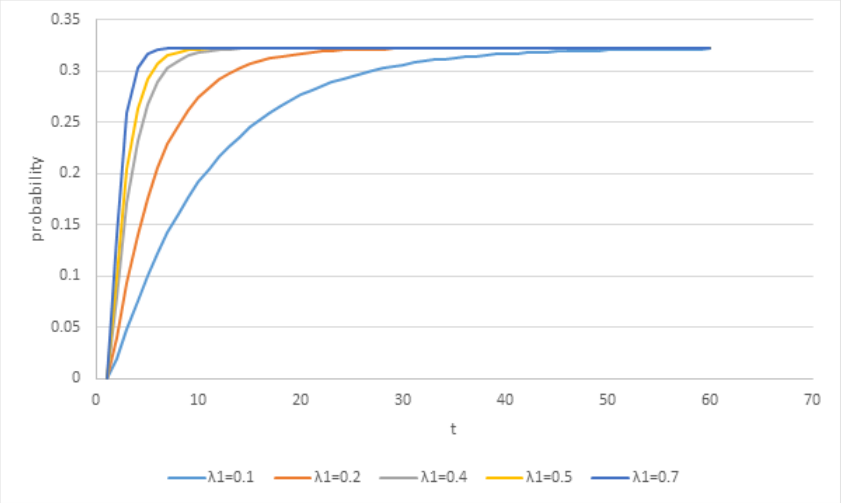

Figure 3.8: Probability of being Patched

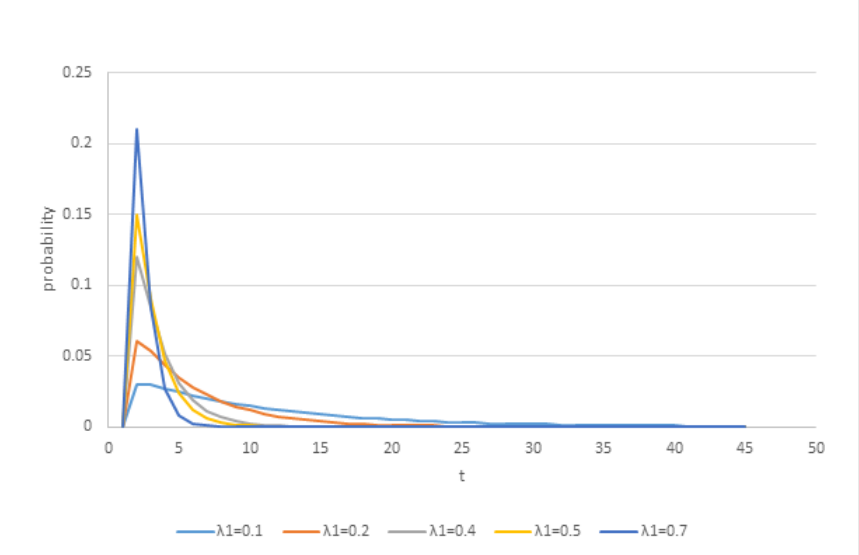

Figure 3.9: Probability of being Patched

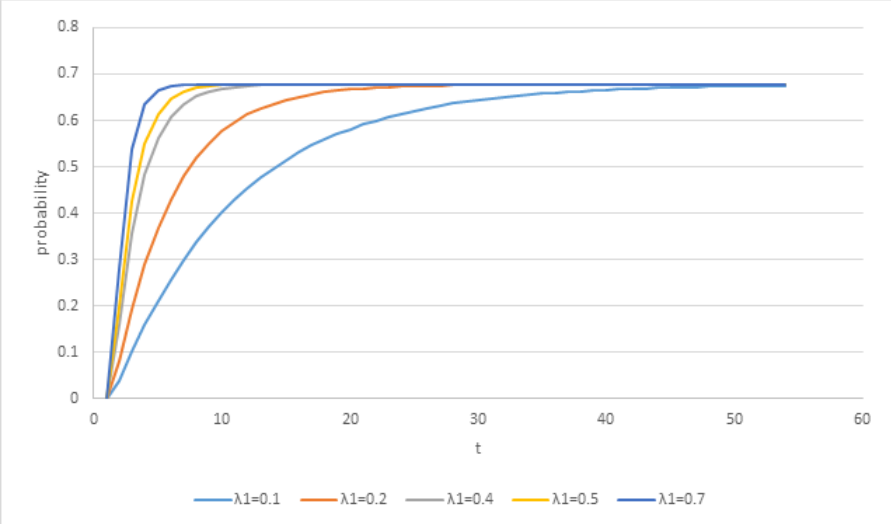

Figure 3.10: Probability of being Patched

3.5 The Risk Factor and Parametric Model

3.5.1 Introducing the Risk Factor and Evaluating the Risk Level as a Function of Time

Vulnerabilities which have been discovered but not patched represents a security risk [19], [25] and [26] which can lead to considerable financial damage or loss of reputation (credibility). Therefore estimating the risk is very important and in the present study we introduce a method to evaluate the risk level [34], [35] of discovered vulnerabilities. By examining Figure 3 we discussed above, that is related to the state Exploited in the Vulnerability Life Cycle, we can clearly see the pattern of exploitability as a function of time. As a function of time, the probability of being exploited increases significantly up to some stage and then eventually become stable.

To evaluate the risk factor of exploiting with respect to the time we consider the changes in the probability and also the CVSS score of a specific vulnerability. We explore the use of the CVSS vulnerability metrics which are publicly available and are being used for ranking the strength of all vulnerabilities.

Let's proceed to define the risk factor as follows:

Let, v_i be any specific vulnerability. Then,

$$\text{Risk}v_i(t) = Pr(v_i\text{is in the state 3 at time}t) \times \text{Exploitability score}(v_i) \tag{3.5.6}$$

We shall use this definition of the Risk Factor in developing our proposed statistical model to evaluate the risk behavior [28], [35].

3.5.2 Development of a Parametric Model to Predict the Probability of Vulnerability Being Exploited.

To accomplish our objective, we developed two statistical models [36], [37] where the response variable Y is the probability of being exploited and is driven by the attributable variable t, the time. At first, for statistical accuracy to homogenize the variance we filtered the data using natural logarithm, lnt . For the second model, to obtain a better fit to the data we introduce a term with an inverse transformation in addition to the filter using the natural logarithm.

Thus, the proposed final forms of the statistical model to estimate the probability of being exploited at time t is given in the table below.

For $\lambda_1 = 0.1$, $\lambda_2 = 0.2$, $\lambda_3 = 0.3$, $\lambda_4 = 0.4$, $\lambda_5 = 0.4$, $\lambda_6 = 0.6$ values we proposed a model to predict the probability at different time intervals as follows.

Table 3.1: Proposed Models for Estimating the Probability of being exploited at time t

Model	R^2	R^2_{adj}
Y= 0.0868 + 0.0523ln(t)	0.7544	0.7528
Y= 0.1772 - 0.27189(1/t) + 0.0326 ln(t)	0.8526	0.8507

As an example, let's take a specific vulnerability labeled as CVE-2016-0467. This has CVSS Base score 4.00, which categorized as medium score with "Impact sub score: 2.9" and "Exploitability sub score: 8.0". For This vulnerability we can measure risk as follows.

Risk for exploit($v_i(t)$) = Pr (v_i is in state 3 at time t) $\times Exploitabilityscore(v_i)$,

=(0.1772 - 0.27189(1/t) + 0.0326 ln(t)) * 8.

Using Equation above, we can predict the risk factor of specific vulnerability at any time interval.

This is an excellent model that gives us an R^2 of 0.8526 and R^2_{adj} of 0.8507. The R^2 , named Coefficient of Determination tells us how much can the change in the response variable be explained and predicted by the attributable variables of the model and considered as the key criterion in evaluating the quality of a model. In other words, R^2 equals to the ratio of the Sum of Squares of the Regression to the Total Sum of Squares. That is,

$$R^2 = \frac{SS_{Reg}}{SS_{Total}} = 1 - \frac{SSS_{Res}}{SS_{Total}}. \tag{3.5.7}$$

Let's consider an example to illustrate these two models further. For the given values

for λ_1 to λ_6 given above, consider the values of the response variable Y (Probability of being exploited) at several values of time t. Table 3.2 illustrates several results obtained and we can obtain the Sum of Squared Error for the model using such data.

Table 3.2: Probabilities estimated using two models for several values of time t.

t	Model 1 Estimate	Model 2 Estimate
1	0.0868	-0.09469
2	0.123051598	0.063851598
3	0.144257423	0.122384761
18	0.237966443	0.256321119
19	0.240794159	0.258878711
20	0.243476798	0.261266372
28	0.261074296	0.27611951
29	0.262909572	0.277598327
30	0.264682623	0.279016035
58	0.299161169	0.304882684
59	0.300055208	0.305519416
60	0.300934221	0.306144133
88	0.320964715	0.320071521
89	0.321555682	0.320474602
90	0.322140046	0.320872795
98	0.326593799	0.323895552
99	0.327124768	0.324254543
100	0.327650401	0.324609648

While the second model qualify to be much better as R^2 is higher compared to the first model as we mentioned previously, it should be noted here that our comparison with respect to the probability of being exploited is in comparison with the probability obtained from our transition metrics for a particular time t.

We can generate such set of models for different vulnerabilities involving different CVSS score and improve further for predicting probabilities with respect to critical stages in Vulnerability Life Cycle of a particular Vulnerability.

3.6 Contribution

Using of the Markov model Approach to Vulnerability Life Cycle, we can have a better understanding of the behavior of a vulnerability as a function time. In the present study we have developed a successful statistical model to estimate the probability of being in a certain stage of a particular vulnerability in its life cycle. In sections 3.3 and 3.4 we have presented our methodology of using the Markov approach and Life Cycle Graph Analysis. This analysis with the application of Markov Chain Theory gave us the basis for calculating estimates for probabilities for different stages of a life cycle of the vulnerability considered.

Further in section 3.5, we have also developed a "RISK FACTOR", and statistical models to estimate the risk for a particular vulnerability being exploited combining our methodology with the exploitability score given in the CVSS score. Using the developed method, we can evaluate the risk level of a particular vulnerability at a certain time.

These developments ensure us with a great advantage in taking measures to avoid exploitations and introduce patches for the vulnerability before attacker takes the advantage of that particular vulnerability.

4 Nonlinear Stochastic Models for Predicting the Exploitability

4.1 Introduction

Obtaining complete information regarding discovered vulnerabilities looks extremely difficult. Yet, developing statistical models requires a great deal of such complete information about the vulnerabilities. In chapter two, we introduced a new concept of Risk Factor [25] of vulnerability which was calculated as a function of time. We introduced the use of Markovian approach [24], [31] to estimate the probability of a particular vulnerability being at a particular state of the vulnerability life cycle.

In this chapter, 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 [25]. Finally, we present a new set of Non-Linear Statistical Models [26] that can be used in estimating the probability of being exploited as a function of time. Our study is based on typical security system and vulnerability data that is available. However, our methodology and system structure can be applied to a specific security system [39] 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 companys security system

in its strategic plan to monitor and improve its process for not being exploited.

Risk is an unavoidable phenomena 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 [31], [33] and [38] in various dimensions. The number of such attacks and the magnitude of the hazards have been heavily increasing throughout recent years. Hackers are getting more active and effective. The risk is getting higher. System administrators and defending professionals are working hard to understand attackers, attacking strategies and effectively defend attacking attempts. To establish successful defending platforms a proper understanding of the risk associated with a given vulnerability is required. If we have effective models that enable the defenders and system administrators to successfully predict the risk of a given vulnerability being exploited as a function of time it will be helpful to plan and implement security measures, allocate relevant resources and defend the systems accordingly. In this chapter, we improve the Markovian approach of Vulnerability Life Cycle Analysis that we presented in chapter three to come up with better modelling techniques to evaluate the risk factor using probability theory and statistical methods.

The key objective of this chapter is to propose and present a rational set of methods to identify the probabilities for each different state in the vulnerability life cycle [4], [47] and use this information to develop three different statistical models to evaluate the Risk Factor of a particular vulnerability at time t. In chapter two, we introduced the strategy of using Markov processes [24], [48] to obtain the tran-

sition probability matrix of all the states of a particular vulnerability as a function of time. We iterated the Markov process and determined that it reaches the steady state with probabilities of reaching the absorbing states [24]. Two absorbing states were identified as exploited and patched states. We proceeded to introduce the Risk Factor that can be used as an index of the risk of the vulnerability being exploited. Finally, we presented successful statistical models that can calculate the Risk Factor more conveniently without going through the Markovian process.

However, in this process, we used a logical and realistic approach to assign initial probabilities for each state of the vulnerability. In this study, we introduce more relevant and sophisticated sets of methods to assign the initial probabilities for each state of Vulnerability Life Cycle based on several logical assumptions. We use the CVSS score [5], [43] as we did earlier, but here we calculate and introduce initial probabilities taking the entire CVE Data Base [6] into consideration.

Finally, using our new methods, we develop three new statistical models for vulnerabilities that differ based on their vulnerability score ranging from 0 to 10 as low risk (0-3.9), medium risk (4-6.9) and high risk (7-10). Using these models the user will be able to estimate the Risk of a particular vulnerability being exploited at time t and to observe the expected behavior of the vulnerability throughout its life cycle.

4.2 Background and Related Methodologies

4.2.1 Vulnerability Life Cycle Analysis Method

Markov chain process approach that we introduced in chapter three, to develop the transition probability matrix included all the important states of Vulnerability Life Cycle. The Vulnerability Life Cycle Graph [3], [41], [46] that we discussed is presented below by Figure 4.1. When we draw a Life Cycle Graph for a given vulnerability, it has several nodes which represent the stages of the Vulnerability Life Cycle. In chapter three we assigned logical probabilities for a hacker to reach each state by examining the properties of a specific vulnerability. With two absorbing states that are named Patched state and Exploited states, we obtained a model the Life Cycle Graph as an absorbing Markov chain [24], [48].

We define, λ_i = the probability of transferring state i to state j.

In actual situations the probability of discovering a vulnerability can be assumed very small. Therefore, for λ_1 we can assign a small value. Then we assigned probabilities to $\lambda_2, \lambda_3, \lambda_4, \lambda_5$, accordingly.

Using these transition probabilities we can derive the absorbing transition probability matrix for a Vulnerability Life Cycle, which follows the properties defined under Markov Chain Transformation Probability Method.

However, in this chapter, instead of randomly assigning transition probabilities for each of the state presented in the Life Cycle, we use a new set of methods that

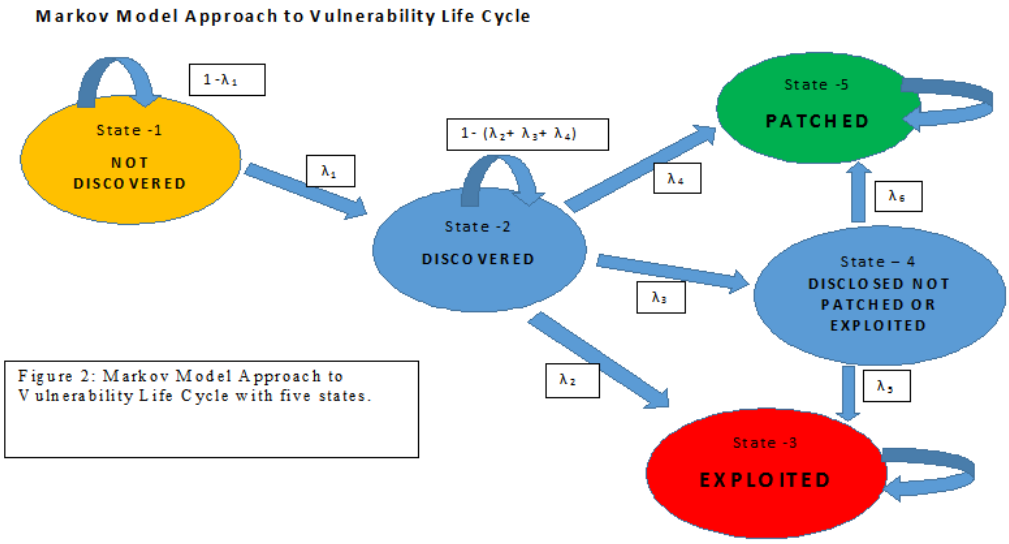

Figure 4.1: Markov Model Approach to Vulnerability Life Cycle with Five States.

are probabilistically more reliable. It is challenging to acquire a complete set of information relevant to Vulnerabilities in a manner that we can calculate the required probabilities conveniently. Therefore, we use available and reliable data resources about Vulnerabilities to develop our methodology that we discuss in the sections that follows.

4.2.2 Common Vulnerability Scoring System (CVSS) and Common Vulnerabilities and Exposures (CVE)

It is important to remind here the usage of Common Vulnerability Scoring System (CVSS) and CVE Details [6], as we gather data from those resources. As we discussed

in details in chapter 3, 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) [1] 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 only 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 (FIRST). CVSS establishes a standard measure of how much concern a vulnerability warrants, compared to other vulnerabilities, so that efforts can be prioritized. The scores range from 0 to 10. Vulnerabilities with a base score in the range 7.0-10.0 are High, those in the ranges 4.0-6.9 are Medium, and 0-3.9 are Low. Hence, the three transition probability matrices and statistical models that we develop in this chapter are based on this classification of the CVSS score.

Common Vulnerabilities and Exposures (CVE) is a dictionary resource and CVE Detail website [6] provides us with the data base in the basic categories with different CVSS scores. CVE Details provide us with quantities of vulnerabilities in different levels of magnitudes ranging from 0 to 10. Instead of randomly assigning a reasonable probability for each different states λ_i's, we now use these data resources

as per their availability in estimating probabilistically reliable values for each state. Our approach in assigning initial probabilities into each state of the Life Cycle is discussed in the subsection below.

4.2.3 Methodology of assigning initial probabilities

Our objective now is focused on assigning initial probabilities for different states in the Life Cycle. In Table 4.1, below we present these initial probabilities that are required in our present study. Estimating them requires a great deal of data resources. To estimate λ_1 as an example, requires the total number of vulnerabilities in each category ranging from 0 to 10 in magnitudes, and information on their discovery with respect to time. Similarly for other states, we need the number of vulnerabilities discovered, exploited before disclosed, exploited after discovery but before patched, patched before the disclosure, patched after the disclosure, under each CVSS score level.

We start with the CVSS scores available for each Vulnerability and categorize them and take the counts for the three different levels of vulnerabilities for possible states. However, it should be noted that, there are no data resources available providing all the data requirements here we have. Therefore, when the CVSS classifications available in the CVE detail website satisfy our requirements, we use those data and when they are not sufficient to make a reliable estimate we use information given by Stefan Frei in his thesis [4] and Secunia Vulnerability information report [2].

We categorized 75705 vulnerabilities according to their CVSS score and under each

Table 4.1: Transition Probabilities in the Vulnerability Life Cycle.

Probability-λ_i	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

of the three categories to find out number of total vulnerabilities and number of exploitations. We shall use this information to assign probabilities of discovery (λ_1) and exploitability(λ_2) for each CVSS score level.

To assign probabilities for Disclosed but not yet patched or exploited (λ_3), Patched before disclosed (λ_4), exploited after disclosed (λ_5) and patched after disclosed (λ_6) we used Secunia vulnerability report information [40], [45] and Frei's results given in his study [4].

Estimating λ_1

To calculate an estimate for λ_1 , the probability of a vulnerability is being discovered for three categories of CVSS score, it is ideal to have an estimate for the population of total number of (known and unknown) vulnerabilities at a particular time so that we can get the proportion of discovered vulnerability out of the total. But, at a given time, it is impossible to know the total number of vulnerabilities in the cyber world as the number of vendors, application software, system software and other apps are

uncountable, so are the number of vulnerabilities that could be existing. Therefore, to have a logical estimate for the total number of vulnerabilities for each year, we first calculated the cumulative number of vulnerabilities, and then calculated the number of vulnerabilities discovered in a particular year as a proportion of cumulative number of vulnerabilities in the next calendar year. Once we have taken these proportions considering all the years from 1999 till 2015, we took the average of those proportions to be our estimate for λ_1.

Assumptions Made for λ_1

When calculating λ_1, it was assumed that, the number of unknown vulnerabilities in a particular year are discovered in the next year and the accumulated number of vulnerabilities in a particular year is an estimate for the population size of the vulnerabilities in the previous year.

Estimating λ_2

Estimate for λ_2, the probability of a particular vulnerability being exploited before patched or disclosed was calculated using the data provided in the CVE Detail website. The entire set of exploited vulnerabilities were calculated for 10 different categories (or CVSS score levels) of interest.

Estimating λ_3, λ_4, λ_5 and λ_6

λ_3, the probability of a vulnerability being disclosed but not yet patched or exploited is calculated using the equation, $\lambda_3 = 1 - (\lambda_2 + \lambda_4)$.

For λ_4, the probability of a vulnerability being patched before disclosed, we used information available in Secunia Report on Vulnerability.

To estimate λ_5, probability of a vulnerability being exploited after disclosed and λ_6, probability of a vulnerability being patched after disclosed we used information given by Stefan Frei in his doctoral thesis [4]. Frei, estimates that the probability of a vulnerability being exploited after it is disclosed is greater than the probability of it being patched. His estimates are that, there is a probability around 0.6, for a disclosed vulnerability being exploited. Therefore we, in developing our model used, fix values of 0.6 and 0.4 respectively for λ_5 and λ_6.

Table 4.2 below presents our results on probabilities for each state with respect to each category/level of vulnerability.

Table 4.2: Estimates of Transition Probabilities for each Category of Vulnerabilities .

Vulnerability level	λ_1	λ_2	λ_3	λ_4	λ_5	λ_6
Low	0.1777	0.01630	0.18369	0.8	0.6	0.4
Medium	0.1888	0.08104	0.11896	0.8	0.6	0.4
High	0.1804	0.14755	0.05244	0.8	0.6	0.4

Using these transition probabilities for each level we can now derive the absorbing transition probability matrix for a Vulnerability Life Cycle, which follows the properties defined under Markov Chain Transformation Probability Method [7], [20] and

[48].

4.3 Transition Matrix for Vulnerability Life Cycle

4.3.1 Executing the Markov process to Transition Probability matrix

Now that we have the Vulnerability Life Cycle Graph with two absorbing states and initial probability estimates for each state, we can write the general form of the transition probability matrix for vulnerability life cycle as follows.

$$P = \begin{bmatrix} 1-\lambda_1 & \lambda_1 & 0 & 0 & 0 \\ 0 & 0 & \lambda_2 & \lambda_3 & \lambda_4 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.3.1}$$

Where,

$P_t(t)$- Probability that the system is in state i at time$t$.

For $t = 0$ we have

$P_1(0)$= 1, Probability that the system is in State 1 at the beginning ($t = 0$).

$$P_2(0) = 0, P_3(0) = 0, P_4(0) = 0, P_5(0) = 0.$$

Therefore, the initial probability can be given as $\begin{bmatrix} 1 & 0 & 0 & 0 & 0 \end{bmatrix}$, that is, the probabilities of each state of the Vulnerability Life Cycle initially. It is clear that, the

"State 1" (Not Discovered) with probability of one represents that at the initial time (for $t = 0$), the Vulnerability is not yet been discovered and therefore the probabilities for all others stages are zero.

Now, for three different categories of Vulnerabilities, we can iterate the transition probability matrix using Markovian process until the matrix reaches its steady state. The iteration algorithm is explained below.

For $t = 0$, we have

$$P^{(0)} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \end{bmatrix},$$

For $t = 1$, results in

$$P^{(1)} = P^{(0)}P,$$

For $t = 2$, we can write

$$P^{(2)} = P^{(0)}P^{(2)},$$

And thus, for $= n$, we have

$$P^{(n)} = P^{(0)}P^{(n)}.$$

Using this method, we can find the pattern of probability that is changing with time and is related to each "state" and then to work on finding the statistical model that can fit the vulnerability life cycle.

As an example, for the vulnerabilities in Category one, where $\lambda_1 = 0.1777$, $\lambda_2 = 0.0163$, $\lambda_3 = 0.1837$, $\lambda_4 = 0.8$, $\lambda_5 = 0.6$, $\lambda_6 = 0.4$ the transition probability matrix is written as follows:

$$P = \begin{bmatrix} 0.8223 & 0.1777 & 0 & 0 & 0 \\ 0 & 0 & .0163 & 0.1837 & 0.8 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0.6 & 0 & 0.4 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}. \tag{4.3.2}$$

As we execute this algorithm, for the vulnerabilities of category one, the stationarity (steady state) was reached (considering to 4 decimal digits) at $t = 86$, that is, the minimum number of steps so that the vulnerability reaches its absorbing states is 86 and the resulting vector of probabilities for each of the absorbing states is obtained as the output of the calculation process. As shown below, the transition probabilities are completely absorbed into the two absorbing states which gives the probability of the vulnerability being exploited and the probability of the vulnerability will be patched. All other states have reached the probability of zero. That is,

$$P = \begin{bmatrix} 0.8223 & 0.1777 & 0 & 0 & 0 \\ 0 & 0 & .0163 & 0.1837 & 0.8 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0.6 & 0 & 0.4 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}, \tag{4.3.3}$$

After 86 iterations stabilized matrix is as follows,

$$P^{86} = \begin{bmatrix} 0 & 0 & 0.1265 & 0 & 0.8735 \\ 0 & 0 & 0.1265 & 0 & 0.8735 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0.6 & 0 & 0.4 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}. \tag{4.3.4}$$

$$\overrightarrow{P^{(86)}} = \overrightarrow{P^{(0)}} P^{(86)} = \begin{bmatrix} 0 & 0 & 0.1265 & 0 & 0.8735 \end{bmatrix}.$$

That is, it will take the hacker 86 steps and a 12.7% chance to exploit the security system and 87.3% probability to reach the patched state. Thus we are sure that after t=86, one of the two states will be reached.

Initially, we defined the 3^{rd} state as "the state of being exploited" and the 5^{th} state as "the state of being patched" in the vulnerability life cycle. Based on the current data resources available relevant to the vulnerabilities of category one we can use these results as estimates for the probabilities of being exploited and being patched. The results from this Markovian model for the vulnerability life cycle shows that the sum of the resulting probabilities equals to one ($0.1265 + 0.8735 = 1$). This in other words indicates that our model estimates that one of these results are expected after t=86 (ex: after 86 days) for a vulnerability in category one. Hence, it is clear that once the "steady state" is achieved, for a vulnerability of category one, estimates of the probability of being exploited is 12.65% and the probability of being patched is

87.35%.

Similarly, for vulnerabilities of categories two and three, the transition probability matrices can be obtained. Transition probability matrices and resulting steady state vectors for those categories are given below.

For vulnerabilities of Category 2;

$$P = \begin{bmatrix} 0.8112 & 0.1888 & 0 & 0 & 0 \\ 0 & 0 & .081 & 0.119 & 0.8 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0.6 & 0 & 0.4 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}, \tag{4.3.5}$$

After 80 iterations stabilized matrix is as follows,

$$P^{80} = \begin{bmatrix} 0 & 0 & 0.1524 & 0 & 0.8476 \\ 0 & 0 & 0.1524 & 0 & 0.8476 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0.6 & 0 & 0.4 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}. \tag{4.3.6}$$

$$\overrightarrow{P^{(80)}} = \overrightarrow{P^{(0)}} P^{(80)} = \begin{bmatrix} 0 & 0 & 0.1524 & 0 & 0.8476 \end{bmatrix}.$$

For vulnerabilities of Category 3;

$$P = \begin{bmatrix} 0.8196 & 0.1804 & 0 & 0 & 0 \\ 0 & 0 & 0.1476 & 0.0524 & 0.8 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0.6 & 0 & 0.4 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}, \tag{4.3.7}$$

After 84 iterations stabilized matrix is as follows,

$$P^{84} = \begin{bmatrix} 0 & 0 & 0.1790 & 0 & 0.821 \\ 0 & 0 & 0.1790 & 0 & 0.821 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0.6 & 0 & 0.4 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}. \tag{4.3.8}$$

$$\overrightarrow{P^{(84)}} = \overrightarrow{P^{(0)}} P^{(84)} = \begin{bmatrix} 0 & 0 & 0.1790 & 0 & 0.821 \end{bmatrix}.$$

Table 4.3 below summarizes our results. Number of iterations (steps) that it takes to reach the steady states and resulting row vectors of probabilities for each three categories of vulnerabilities are given in this table.

Table 4.3: Number of iterations (steps) to reach the steady state and Steady State Vector for each category of Vulnerability .

Vulnerability level	Number of iterations	Steady state probability	Probability of being exploited	Probability of being patched	Sum
Low	86	[0 0 0.1265 0 0.8735]	0.1265	0.8735	1
Medium	80	[0 0 0.1524 0 0.8476]	0.1524	0.8476	1
High	84	[0 0 0.179 0 0.821]	0.179	0.821	1

4.4 Risk Factor Model-Calculating the Risk as a Function of Time

Now that we have the steady state vector with the probabilities for patching and getting exploited, we can calculate the risk of a particular vulnerability using the risk factor that we introduced in the previous chapter.

$$\text{Risk}v_i(t) = Pr(v_i\text{is in the state 3 at time t}) \times \text{Exploitability score}(v_i) \qquad (4.4.9)$$

Exploitability score for the vulnerability can be taken from the CVSS score as we mentioned earlier. With our results for three different levels of vulnerabilities, now we have a better index for the risk factor since our initial probabilities were not just chosen randomly, but were estimated using the available and reliable data sources. As an example, lets consider a vulnerability in the lower level with an exploitability score of 2.4. Assume that we need to find the Risk factor of that vulnerability at $t = 50$. Then, using the Markov process we can come up with the resulting vector of the vulnerability that gives us the probabilities of being in each different state at that particular time. However, iterating Markov process for each time would not be a very efficient process due to the analytical calculations. Therefore, we proceed to

move on to develop three different nonlinear statistical models that make it much more convenient for the designed calculation.

To further explain the usage of the Risk Factor lets take an example. Consider a vulnerability given in the Table 4.4 below. With the published date and the exploitability score known for that vulnerability, we can now calculate the risk of being exploited at a particular date from the published date. For the first vulnerability V_1 (CVE 2016-0911) which is a low risk vulnerability the risk factor is 0.2474 and for the other two categories of medium and high risk levels, vulnerabilities V_2(CVE 2016-2832) and V_3(CVE 2016-3230), risk factors are 0.3667 and 1.17702 respectively.

Table 4.4: Three vulnerabilities in each categories with their details and the calculated risk factors.

Vulnerability	Published date	CVSS score	$e(v_j)$	(t_j)	$R(v_j(t))$
V_1(CVE 2016-0911)	6/19/2016	1.9 (Low)	3.4	5	0.2474
V_2(CVE 2016-2832)	6/13/2016	4.3 (medium)	2.8	11	0.3667
V_3(CVE 2016-3230)	6/15/2016	9 (High)	8	9	1.702

The risk factor can be graphed as a function of time. The figure below shows the behavior of the risk factor of the middle level vulnerability V_2(CVE 2016-2832) over a time period of 101 days starting from 6/13/2016. We notice that the risk factor increases rapidly within around first 10 days indicating that once a vulnerability is published, the risk of being exploited rapidly increases. Even after this rapid increase, the risk does not show a decreasing behavior. This specific behavior is due to our

model structure of the vulnerability life cycle. That is, consisting with two absorbing states (being exploited and being patched), we assume that either one of two outcomes are possible for a given vulnerability. Therefore, considering state of being exploited as an absorbing state the life cycle does not move to any other state beyond being exploited which explains why this graph stay increased without decreasing over the time.

Figure 4.2 above illustrates the behavior of the Risk Factor as a function of

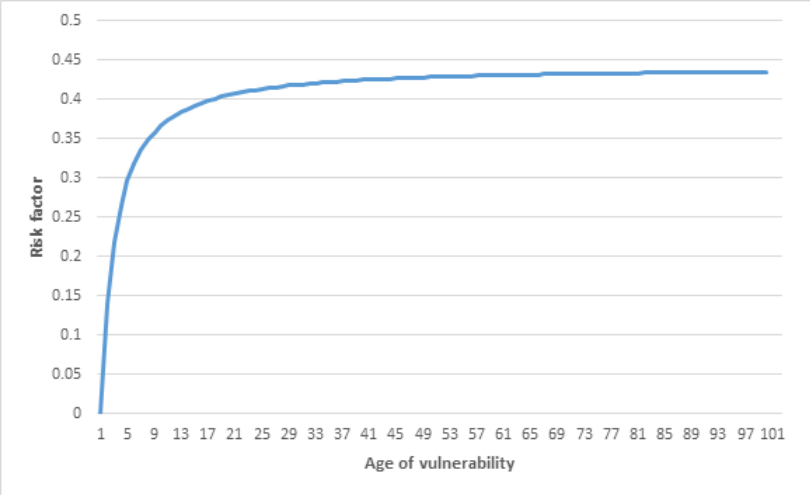

Figure 4.2: Behavior of the Risk Factor as a function of time

time. The curve shows a rapid increase in the risk factor initially as expectable since the vulnerability immediately create a risk with its discovery and disclosure. Based on the graph, we can conclude that over the time with a life cycle consisting two absorbing states, the Risk Factor of a given vulnerability increases rapidly and become stable at a higher level of risk without decreasing back. This behavior exemplifies the threat any vulnerability would impose on an information system. As far as a proper patch is released and installed a probable harm from a given vulnerability

increases monotonously. However, it should not be misinterpreted in the view point that the risk from a given vulnerability never reduces. Our Absorbing Markovian Model does not consider some of the interactions that might take place in the real world situations. Our intention here is to show the impact of a vulnerability until it is not patched. Outcomes from the situations where patching attempts and exploit attempts after and before disclose should be explained in much border modeling aspect of the vulnerability life cycle.

4.5 Non Linear Statistical Models for Exploitability

4.5.1 Model Building

In the previous section we developed an analytical algorithm that identifies the number of steps (time) that the transition probability matrix of the vulnerability life cycle will reach a steady state. Thus, for a given vulnerability in the categories of Low, Medium and High risk levels, we can include with the probability of being exploited (having hacked) and the probability of being patched as a function of time. However, this process is time consuming and the Markovian iteration process would be quite difficult to perform every time. Using this approach to find the minimum number of steps for each category we obtained t=86 steps for category one vulnerabilities, t=80 steps for category two vulnerabilities and t=84 steps for category three vulnerabilities. Then, we recorded the probability of being exploited at the each step. Thus, we have for each category a 2 x 86, 2 x 80 and 2 x 84, matrices of information, respectively. Our

goal is to utilize this information and develop a statistical model for each category to be able to predict the probability of being exploited as a function of time and thus bypassing the analytical difficulties.

A sample of the data for each category is shown in Appendix C. All these of the data sets exhibit nonlinear behavior and thus multiple regression is not applicable. After very exhaustive research, we were able to identify two sets of nonlinear statistical models for each category.

The general analytical focus of the statistical models that we found are of the forms:

Model 1: Y (exploitation probability) $= \alpha_0 + \ \alpha_1 \frac{1}{t} + \alpha_2 \ln t \ + \varepsilon$

and

Model 2: Y (exploitation probability) $= \beta_0 + \ \beta_1 \frac{1}{t} + \beta_2 \ln t \ + \varepsilon$,

Where, Y is the probability of being exploited, α and β are the vector of coefficients or weights, t being the time given in steps and ε is the modelling error [49]. We used the method of maximum Likelihood estimation to obtain the estimates of the coefficients that drives these models [36], [50].

Model-1

The best nonlinear statistical model that we developed for Low, Medium and High Vulnerability categories are given below along with their R^2(coefficient of determination), R^2_{adj} (R^2adjusted).

Low (Category one) risk vulnerabilities:

$$Y = 0.084197 - 0.116756 \left(\frac{1}{t}\right) + 0.011321 \ ln\,(t)\,,$$

with R^2= 0.8684, $R^2_{adj} = 0.8653$.

Medium (Category two) risk vulnerabilities:

$$Y = 0.111073 - 0.143992 \left(\frac{1}{t}\right) + 0.011461 \; ln\,(t)\,,$$

with R^2= 0.8888, $R^2_{adj} = 0.8859$.

High (Category three) risk vulnerabilities:

$$Y = 0.133927 - 0.169314 \left(\frac{1}{t}\right) + 0.012375 \; ln\,(t)\,,$$

with R^2= 0.8988, $R^2_{adj} = 0.8963$.

As we will discuss R^2 reflects on the quality of the proposed model.

Model-2

In investigating to see if we can improve the precision of the model 1, we have found that by implementing another logarithmic filter to our initial model to further homogenizing the variance of our data. We obtained a set of models that gives us better results increasing the accuracy of our prediction approximately by 9% compared to the Model 1. New model equations for each of the categories are given below.

Low (Category one) risk vulnerabilities:

$$Y = 0.135441 - 0.308532 \left(\frac{1}{t}\right) + 0.002030 \; ln\,(ln\,(t))$$

with R^2= 0.9576, $R^2_{adj} = 0.9566$.

Medium (Category two) risk vulnerabilities:

$$Y = 0.169518 - 0.356821\left(\frac{1}{t}\right) + 0.007011\ ln\left(ln\left(t\right)\right)$$

with R^2= 0.962, $R^2_{adj} = 0.961$.

High (Category three) risk vulnerabilities:

$$Y = 0.135441 - 0.308532\left(\frac{1}{t}\right) + 0.002030\ ln\left(ln\left(t\right)\right)$$

with R^2= 0.9588, $R^2_{adj} = 0.9577$.

Thus, model 2 is a significant improvement in the R^2 over model 1.

Both models give very good predictions of the probability of exploitation as a function of time. However, "Model-2" seems to give better predictions because of the additional logarithmic filtering that we applied to homogenize the variance further. Table 4.5 and 4.6 summarizes the 6 model equations with respective R^2(coefficient of determination), R^2_{adj} (R^2adjusted) values for convenient comparison.

Table 4.5: Nonlinear Statistical Model 1 to estimate the probability of being exploited as a function of time.

Category	Model 1 Equation	R^2	R^2_{adj}
Low (0-4)	$Y(t) = 0.0841970.116756(1/t) + 0.011321\ln(t)$	0.8684	0.8653
Medium (4-7)	$Y(t) = 0.1110730.143992(1/t) + 0.011461\ln(t)$	0.8888	0.8859
High (7-10)	$Y(t) = 0.1339270.169314(1/t) + 0.012375\ln(t)$	0.8988	0.8963

Table 4.6: Nonlinear Statistical Model 2 to estimate the probability of being exploited as a function of time.

Category	Model 2 Equation	R^2	R^2_{adj}
Low (0-4)	$Y(t) = 0.1354410.308532(1/t) - 0.002030\ln(\ln t)$	0.9576	0.9566
Medium (4-7)	$Y(t) = 0.169518 - 0.356821(1/t) - 0.007011\ln(\ln t)$	0.962	0.961
High (7-10)	$Y(t) = 0.1917010.383521(1/t) - 0.00358\ln(\ln t)$	0.9588	0.9577

4.5.2 Evaluation of the Models

We used R^2(coefficient of determination), R^2_{adj} (R^2adjusted) and residual analysis using actual data that we did not use in the model building to validate the accuracy and the quality of these models. $R^2$is commonly used to measure the goodness of a statistical model and is defined as,

$$R^2 = \frac{SS_{Reg}}{SS_{Total}} = 1 - \frac{SS_{Res}}{SS_{Total}},$$

Where SS_{Res} or SSE is the Sum of Squares of Residual and SS_{Total} is the Total Sum of Squares. It is also referred to as the Coefficient of Determination. In our case the R^2=.96 states that the model is an excellent fit such that the 96% of the behavior in the response variable (probability of being exploited) is explained and predicted by the attributable variable (time- t) and only a 4% of the change in the response variable is not explained due to the variance.

In order to be more confident in interpreting the value of R^2 we also calculate the R^2_{adj} (R^2adjusted) to address the issue of bias.

R^2_{adj} (R^2adjusted) is defined by

$$R^2_{adj} = 1 - \frac{(n-1)\ SS_{Res}}{(n-p)\ SS_{Total}},$$

Where, n is the sample size and, p is the number of risk factors (attributable variables) in our models. The closer the $R^2$and R^2_{adj} to one, the higher the quality of our models.

We also performed residual analysis of all the models to determine if the error factor has significantly contributed to the accuracy of our models. In all cases, the residual error was not significant. Finally we tested all our models with the actual data that we did not include in developing the models and the results were exceptional.

As mentioned, we needed a best fitting three Statistical models to calculate the risk factor conveniently. In other words, we expected to obtain a best fitting model that can replace the Markovian iteration and hence to avoid the difficulty in estimating of the probabilities for time *"t"* earlier to the "steady state". With these new models we have achieved our goal.

4.6 Contribution

In this chapter, we have improved the models we build up in chapter three.. We have improved the calculation methods of initial probabilities and creating 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.

5 A Comprehensive Analysis on Vulnerability Space

5.1 Introduction

In chapter two, we introduced the Vulnerability Space as a probability space. We defined the sample space of vulnerabilities and presented a set of events that could occur in the subspace of discovered vulnerabilities. We also mentioned that there is a relationship between these events and the states of vulnerabilities [26].

In this chapter we further discuss these events and their likelihood in the vulnerability space. It is our objective in this chapter to analyze the vulnerability space in several aspects so that a complete foundation on statistical analysis on vulnerabilities is developed. This chapter will present all currently identified events and states belongs to the vulnerability space and define them using probability theory. In chapters 3 and 4 we looked in to the behavior of one particular vulnerability over time and used our inferences from available data to obtain useful estimates to help cybersecurity related decision making [8], [24]. In this chapter, we will look in to the Big Picture of vulnerability space and analyze it to lay down a strong foundation for the use of any future Cybesecurity related study.

5.2 States of a Vulnerability and the Likelihood of States in the Vulnerability Space

Any vulnerability during its life time passes through a series of stages that we identify as states of vulnerability. This behavior of a vulnerability over time affects the likelihood of states in the vulnerability space as we mentioned in chapter two. List of events in a vulnerability life cycle as mentioned in section 2.5 moves vulnerabilities into such different states during their life time. Cumulative effect of such events will constitute the probability estimates an observer will have in the vulnerability space. In this chapter, we introduce a new logical approach using a Venn diagram that presents all the states in vulnerability space after they are discovered.

Any vulnerability can be placed in to a specific subset which can be categorized based on events. These key events are Discovery of Vulnerability, Disclosure of Vulnerability, Exploitation of Vulnerability and Patch released of Vulnerability [4]. These subsets and corresponding probabilities of events that generate the elements of these subsets are of our interest in this section. But, occurrence of these states are not in an exact order always. Depending on who discovers the vulnerability and several other factors, order in occurrence of these states could vary from one vulnerability to another. In this chapter, we introduce a new approach of vulnerability analysis using Venn diagram.

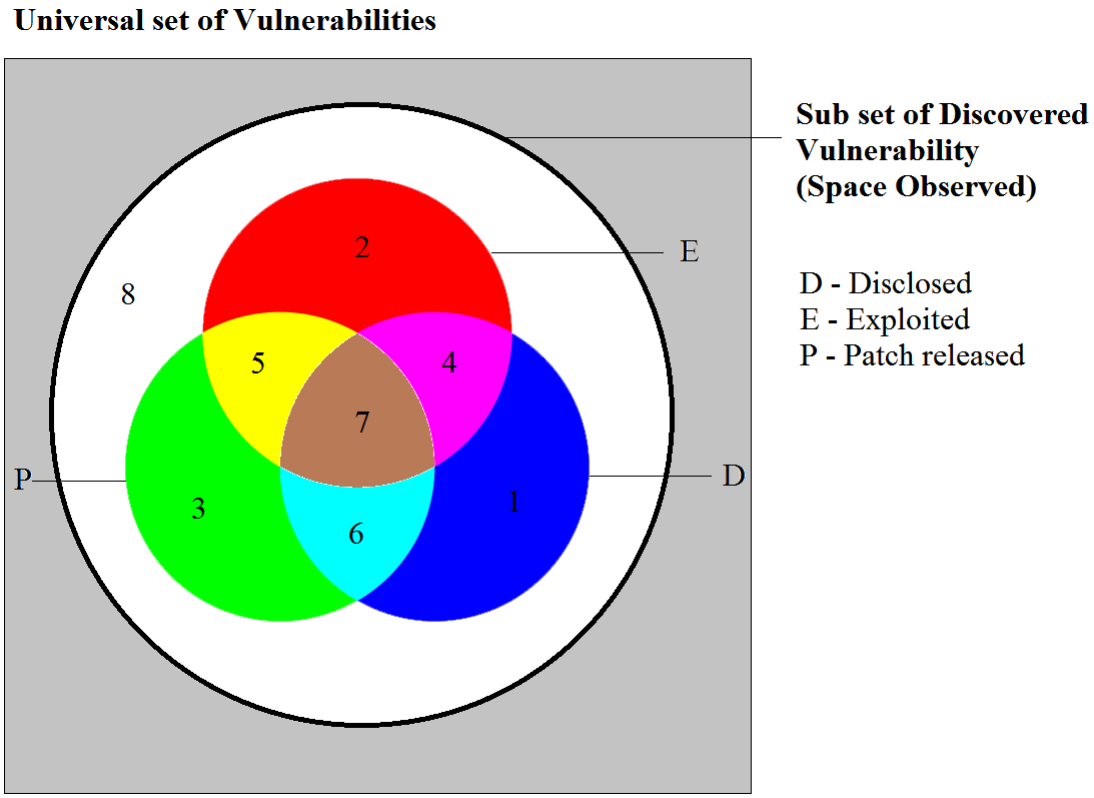

Figure 5.1: Venn diagram representing the Vulnerability Space

5.2.1 Venn diagram for Vulnerability

Figure 5.1 above illustrates all the probable states of vulnerabilities in the vulnerability space. The square represents the universal set of the vulnerabilities including undiscovered vulnerabilities. Black circle inside the square represents the subset of Discovered Vulnerabilities by any human being. Three intersecting circles represents three main states namely Disclosed (D), Exploited(E) and Patch released(P). A Vulnerability that has been discovered by someone, but no other action taken by any party is represented by the white space outside three intersecting circles represented

by the number 8. States represented by numbers from 1 to 7 with respective colors can be explained as follows. Grey color area outside the black circle represents the set of undiscovered, unobserved and therefore unknown vulnerabilities. These states are further explained below.

Dark Blue (1)

This area represents the vulnerability state where a vulnerability is disclosed to the public but it is neither exploited nor is a patch released for that. In other words this area represents the, set of vulnerabilities Disclosed but has not been Exploited or Patch released. Probability of a vulnerability being in this state can be given as follows.

$$\Pr[D \cap (E \cup P)^c] = \Pr(\text{vulnerability is disclosed but not patch released or exploited})$$

Red (2)

The area given in Red color and number 2, represents the state where a vulnerability is exploited but it is neither disclosed nor is a patch released. In other words, this area represents the set of vulnerabilities in the Vulnerability Space, that is Exploited but has not been Disclosed or Patch released. Probability of a vulnerability being in this state can be given as follows.

$$\Pr[E \cap (D \cup P)^c] = \Pr(\text{vulnerability is exploited but not disclosed or patch released})$$

Green (3)

The area given in Green color and number 3, represents the state where a patch is released for a vulnerability but it is neither Disclosed nor Exploited. In other words, this area represents the set of vulnerabilities in the Vulnerability Space that is patch released but has not been Disclosed or Exploited. Probability of a vulnerability being in this state can be given as follows.

$$\Pr[P \cap (E \cup P)^c] = \Pr(\text{vulnerability is patch released but not disclosed or exploited})$$

Pink (4)

The area given by Pink color and number 4, represents an intersection of Disclosure and Exploitation. That is the state where a vulnerability is both Disclosed and Exploited but has not been Patch released. In other words, this area represents the set of vulnerabilities in the vulnerability space that has been Disclosed and Exploited but not Patch released. Probability of a vulnerability being in this state can be given as follows.

$$\Pr[(D \cap E) \cup P^c] = \Pr(\text{vulnerability is disclosed and exploited but not patch released})$$

Yellow (5)

The area given by Yellow color and number 5, represents an intersection of Patch release and Exploitation. That is, the state where a vulnerability is both Patched and Exploited but has not been Disclosed. In other words, this area represents the set of vulnerabilities in the vulnerability space that has been Patch released and Exploited

but not Disclosed. Probability of a vulnerability being in this state can be given as follows.

$$Pr[(E \cap P) \cap D^c] = \text{Pr(vulnerability is exploited and patch released but not disclosed)}$$

Light Blue (6)

The area given by Light Blue color and number 6, represents an intersection of Disclosure and Patch release. That is, the state where a vulnerability is both Disclosed and Patch released but has not been Exploited. In other words, this area represents the set of vulnerabilities in the vulnerability space that has been Patch released and Disclosed but not Exploited. Probability of a vulnerability being in this state can be given as follows.

$$\text{Pr}[(D \cap P) \cap E^c] = \text{Pr(vulnerability is disclosed and exploited but not patch released)}$$

Brown (7)

The area given by the White color in the middle of the figure and number 7, represents the intersection of all three possible events Disclosure, Exploitation and Patch release. That is, the state where a vulnerability is Disclosed, Exploited and also Patch released. In other words, this area represents the set of vulnerabilities in the vulnerability space that has been Disclosed, Exploited and also Patch released. Probability of a vulnerability being in this state can be given as follows.

$$\text{Pr}[(D \cap E \cap P)] = \text{Pr(vulnerability is disclosed,exploited and patch released)}$$

5.3 Relationship between Cyber Events and Vulnerability States

Now that we have explained all the important events that we consider regarding vulnerabilities and all the states of vulnerabilities, it is important to discuss the relationship between those events and status of vulnerabilities. Table 5.1 below illustrates the this relationship. As illustrated in the Table 5.1, Event 1 constitute the State 1

Event #	Event	State 1	State 2	State 3	State 4	State 5	State 6	State 7
1	Event that a discovered vulnerability is "disclosed" to the public before it is "exploited" or "patch is released"							
2	Event that a discovered vulnerability is "disclosed" to the public after a "patch is released" before it is "exploited"							
3	Event that a discovered vulnerability is "disclosed" to the public after it is "exploited" but before the "patch is released"							
4	Event that a discovered vulnerability is "disclosed" to the public after it is both exploited and released with a patch developed							
5	Event that a discovered vulnerability is "released with a patch" before it is "disclosed" or "exploited"							
6	Event that a discovered vulnerability is "released with a patch" before it is "exploited" but after it is "disclosed"							
7	Event that a discovered vulnerability is "released with a patch" before it is "disclosed" but after it is "exploited"							
8	Event that a discovered vulnerability is "released with a patch" after it is both "disclosed" and "exploited"							
9	Event that a discovered vulnerability is "exploited" before it is "disclosed" or "patch is released"							
10	Event that a discovered vulnerability is "exploited" before it is "disclosed" but after "patch is released"							
11	Event that a discovered vulnerability is "exploited" before the "patch is released" but after "disclosed"							
12	Event that a discovered vulnerability is "exploited" after it is both "disclosed" and "patch is released"							

Table 5.1: Relationship between the events in vulnerability space and states of vulnerabilities

in the vulnerability space. That is, if someone discloses a vulnerability to the public before it is exploited or a patch is released, the vulnerability is said to be in State 1 that we defined early. (However, before further explanations, it must be noted that numbers we have given for events and states in this study are only for reference purposes and do not constitute any order.) Such a disclosure without releasing a patch is extremely dangerous and one of our objectives in the present study is to

assess the consequences of such a disclosure using available data. Event 9 as mentioned in the Table 5.1, constitutes the State 2 of a vulnerability. That is the case where a discovered vulnerability is exploited before it is disclosed or patch is released. Such a situation arises mostly if an Attacker discovers a vulnerability and execute an exploit. This is one of the major concerns and vulnerabilities that has no patch developed patch pose a greater threat. What is referred to as Zero Day Vulnerabilities [44] comes under this category. Vulnerabilities in this category exist known only to the attackers and therefore considered as major threat in Cybersecurity. Event 5 as mentioned in the Table 5.1, constitutes the Sate 3 of a vulnerability. That is the case where a discovered vulnerability is released with a patch before it is disclosed or exploited. Once this happens, vulnerability initially go to the Sate 2 however, it is expectable that with the releasing of the patch users will install the patch in the computer systems knowing or unknowing the vulnerability. It is possible for a vendor to release the patch first and then after a several days or weeks vulnerability will also be disclosed to the public. Practically, this is logical procedure. As the vendor or system administrator discovers a vulnerability, efforts are taken to develop a patch without disclosing it to the public. Once the patch is developed, the vendor would release it as an update for the software allowing users to install it. The Vendor may monitor the system for any further issues to see if the patch is strong enough and if there are other effects of installing the patch etc. After some time once the Vendor is confidence about the installed patch, information regarding the vulnerability may be disclosed to the public.

Events 3 and 11 as mentioned in the Table 5.1, constitute the State 4 of Vulnerability. There are two possible cases here. One is the case where a discovered vulnerability is disclosed to the public after it is exploited but before the patch is released. The other case is where a discovered vulnerability is exploited before the patch is released but after disclosed. Both these events occur before a patch is released and the vulnerability is in the state such that it is both exploited and disclosed. However, likelihood of two events causing this state mainly depends on who discovers the vulnerability. In one case, Exploitation occurs before the Discloser. This is where the hacker is in the lead. The hacker (or attacker) discovers the vulnerability, then, he creates an exploit code and exploits the vulnerability executing that exploit code. The vendor would find out this vulnerability after some attacks by himself or through a malware detector [32] who identifies the exploit and hence the vulnerability. But still, the vender is behind the hacker. On the other hand, there could be a situation where a vulnerability is discovered by a vendor or a malware detector (ex: White Hat-Hacker [18]) and for some reasons disclose it before a patch is released considering the nature of the vulnerability and some other factors. In such a case, vender is in lead to the Hacker. But, before the patch is developed and released to the users, hackers or attackers could develop an exploit for this disclosed vulnerabilities and execute it. Therefore, in any scenario State 04 represents a strong position for the attacker against the vendor. Events 7 and 10 as mentioned in the Table 5.1, constitute the State 5 of Vulnerability. Those are the two cases where in the first case, a discovered vulnerability is released with a patch before it is disclosed but after it is exploited and

in the second case, a discovered vulnerability is exploited before it is disclosed but after patch is released. Both these events occur before a disclosure of a vulnerability and the vulnerability is in the state such that it is both exploited and released with a patch. Similar to the State 4 this state can also be explained from two different point of views of the hacker and the vendor. That is, a hacker might discover a vulnerability first and execute and exploit for that. Then, after some time from this hackers execution, vender gets to know about the vulnerability and release a patch as soon as possible. So, in such a case the vulnerability has been already exploited but then a patched is released so that users can install the patch and secure themselves of possible exploit. On the other hand, if the vulnerability is first discovered by the vendor, and as soon as possible release a patch so that the users can install the released patch. However, the vulnerability is not publicly disclosed. If a hacker somehow find out this vulnerability and create an exploit, it is still possible to execute the exploit on the systems where the released patch is not yet installed properly. However, it is clear that, since there is no public disclosure of the vulnerability, there is a very little chance for an exploitation. However, both theoretically and practically it is not impossible.

Events 2 and 6 as mentioned in the Table 5.1 constitute the State 6 of Vulnerability. Those are the cases where a discovered vulnerability is disclosed to the public after a patch is released before it is exploited and where a discovered vulnerability is released with a patch before it is exploited but after it is disclosed. This state is considerably favorable as it is an intersection of being Patch released and Disclosed

and both these events occur before an exploitation of the vulnerability. Therefore, both the possibilities can be explained in Vendor Lead perspective. That is, a situation, either vendor discloses a vulnerability soon after the release of the patch or the vendor released the patch first and then disclose the vulnerability. Events 4, 8 and 12 as mentioned in the Table 5.1, constitute the State 7 of vulnerability. This is the intersection of all three main events Disclosure, Exploitation and Patch release. Three events 4, 8 and 12 explains the different ways that a vulnerability could come into this intersection. In the next section, we present a new Venn diagram associating all these events with their probabilities that allows us to distinguish the role of hacker and vendor with respect to the events.

5.4 Explicit Venn diagram for Vulnerability

Now that we have identified all the probable events in the vulnerability space with respect to discovered vulnerabilities, we can further analyze the likelihood of those events using available data resources. However, it is very important to recognize the origination of these events in reference to the parties associated with the each event. That is, some events in the vulnerability space are derived by actions of attackers or hackers while some other events are derived by actions of system administrators, malware detectors and Ethical (White Hat) Hackers [18]. Intention of the parties associated directly characterizes the outcomes of events in the vulnerability space. Therefore, the likelihood or the probability of a particular event to occur in the vulnerability space should be defined separately based on the intention of the party

associated. To further illustrate this nature in the vulnerability space, we use an Explicit Venn Diagram. In this Venn diagram we illustrate the regions in different colors and numbers while considering the nature or originators intention.

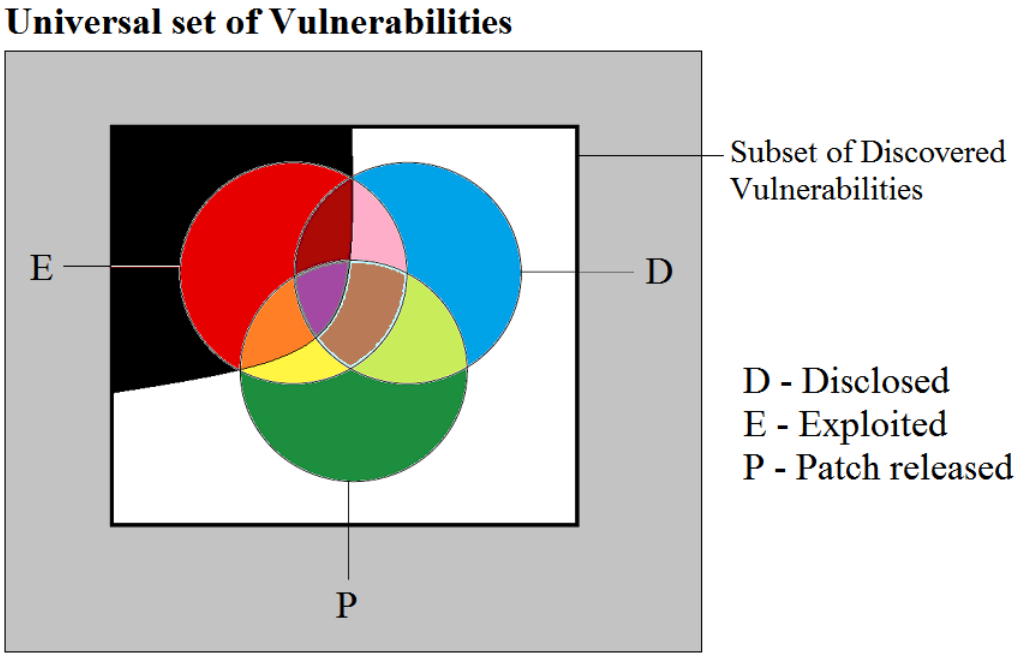

Figure 5.2: Explicit Venn diagram considering the nature of intention of the discoverer

In Table 5.1, we explained the relationship between 7 different states of a vulnerability and their relationship with the 12 different events that occur in the Vulnerability Space. Now, we further analyze these events and states considering the likelihood to occur them. We can also consider the specific nature and characteristics of those states and look into available data and check for real world consequences of these events and states.

5.4.1 Identifying and Differentiating the Events Based on the Intention of the Discoverer

In the Vulnerability spaces, events are generated and defined with the discovery of a vulnerability. In reality, the first perception of a vulnerability in the Cyber Space can be identified as the discovery. However, the consequences and events after a discovery is directly depend on the intention of the discoverer. We consider that the intention of the discoverer could be either good or bad. One with a good intention, as an example a system administrator, an independent malware detector or an ethical hacker might discover a particular vulnerability. On the other hand, someone such as a hacker or an attacker, with bad intentions looking for a particular exploitation might find a vulnerability. Therefore, from this point onward, we consider this scenario and separately identify the outcomes for both the cases. In the Figure 5.2 above, the area covered by the BLACK LINE going through circles illustrates the states related to events of vulnerabilities where the intention of the discoverer is considered as BAD. The other area of the Venn diagram illustrates the states related to events of vulnerabilities where the intention of the discoverer of the vulnerabilities are GOOD. For better understanding of the Venn diagram with related states and events lets remind the areas mentioned. Three circles D, E and P represents the states of Discloser, Exploitation and Patch release respectively. As we explained in detail in the previous sections, there are intersections of these states caused by events. As an example, a discovered vulnerability could be disclosed to the public after it is exploited but before the patch is released. This event is attributed to the State 4 (colored in Pink) in the

Figure 5.1. But, it should also be noted that the event where a discovered vulnerability is exploited before the patch is released but after disclosed also attributed to State 4. Even though, these two different events characterizes the same state of a vulnerability in the vulnerability space, it is quite clear that two events must result in different consequences in reality. As we can understand, the event where an exploitation could occur before a patch release, or a discloser is a result of a Black Hat Hackers BAD intentions and actions. Therefore, State 4 in Figure 5.1 should be further analyzed. The same situation is there for State 5 and State 7. Once the discoverer is considered as in Figure 5.3, these states are now clear for our analysis. States 5, 6, 7, 8, 9 and 10 in Figure 5.3 represents the states derived from discoveries with bad intentions (discoveries by attackers or hackers). these states Who at first discover the vulnerability and nature of intentions makes a complete difference in outcomes. Figure 5.3 below combined with Table 5.2, satisfactorily explain us these differences. With this analysis on the vulnerability space as a probability space, now we are in a position to finalize our comprehensive analysis on the states and related events. As presented in Figure 5.1 and Table5.1 in this chapter we had 12 events in the subspace of Discovered vulnerabilities defining 7 states. But, now that we have considered the intention of the discoverer of a vulnerability and the possible differences in leading the order of events we have 10 different states within the intersecting circles of the Venn diagram. As Figure 5.3 presents, intersection of state of Exploited and Disclosed are now divided into two parts as State 5 and State 6. Similarly, the intersection of Exploited state and Patch released is also divided in to two parts as State 7 and State

Event #	Event	State 1	State 2	State 3	State 4	State 5	State 6	State 7	State 8	State 9	State 10
1	Event that a discovered vulnerability is "disclosed" to the public before it is "exploited" or "patch is released"										
2	Event that a discovered vulnerability is "disclosed" to the public after a "patch is released" before it is "exploited"										
3	Event that a discovered vulnerability is "disclosed" to the public after it is "exploited" but before the "patch is released"										
4	Event that a discovered vulnerability is "disclosed" to the public after it is both exploited and released with a patch developed										
5	Event that a discovered vulnerability is "released with a patch" before it is "disclosed" or "exploited"										
6	Event that a discovered vulnerability is "released with a patch" before it is "exploited" but after it is "disclosed"										
7	Event that a discovered vulnerability is "released with a patch" before it is "disclosed" but after it is "exploited"										
8	Event that a discovered vulnerability is "released with a patch" after it is both "disclosed" and "exploited"										
9	Event that a discovered vulnerability is "exploited" before it is "disclosed" or "patch is released"										
10	Event that a discovered vulnerability is "exploited" before it is "disclosed" but after "patch is released"										
11	Event that a discovered vulnerability is "exploited" before the "patch is released" but after "disclosed"										
12	Event that a discovered vulnerability is "exploited" after it is both "disclosed" and "patch is released"										

Table 5.2: Events and their relationship to the states in the vulnerability space considering the intention of the discoverer

8. The area of the intersection of all three circles is also now divided in to two parts as State 9 and State 10. Analytical observation of these new states came into context with the separation we presented with the Black line that we used to separate the vulnerabilities discovered by attackers or hackers. This separate identification reveals several important events in the vulnerability space. All the states including the new observations considering who discovered the vulnerability is discussed in details with their probabilities in the next section.

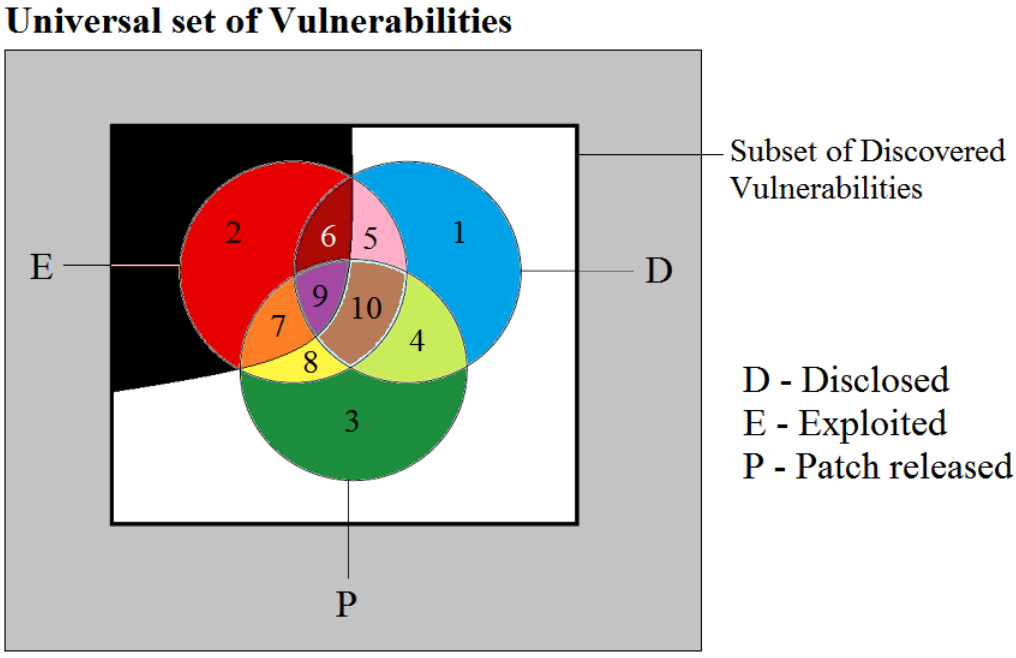

Figure 5.3: Explicit Venn diagram considering the nature of intention of the discoverer in relation to the events in Table 5.2.

5.5 Inferences on States of the Vulnerability Subspaces

5.5.1 Probabilities and inferences regarding Non-intersected states

States 1, 2 and 3 as we defined earlier represents non intersecting states. Out of these 3 states only the State 02 represents the involvement of an attacker. Relevant probabilities of a vulnerabilities observed at random being in each of these states are given below.

State 01

Probability of a discovered vulnerability observed at random being disclosed given that it is not exploited and not patch released can be estimated using the definition of the conditional probabilities as follows.

$$Pr(D|(E^{'} \cap P^{'})) = \frac{Pr[D \cap (E^{'} \cap P^{'})]}{Pr(E^{'} \cap P^{'})} = \frac{\frac{n([D \cap (E^{'} \cap P^{'})])}{n(S)}}{\frac{n(E^{'} \cap P^{'})}{n(S)}} = \frac{n([D \cap (E^{'} \cap P^{'})])}{n(E^{'} \cap P^{'})} \quad (5.5.1)$$

To obtain better statistical inferences on this state it is required to obtain the data on number of disclosed vulnerabilities that has not been exploited or patch released and the number of vulnerabilities that are not been taken any action after discovered by any party. The latter is almost impossible to estimate using the available data. Number of vulnerabilities kept hidden by vendors and hackers unknown. Same challenge is there for other states below.

State 02

Probability a discovered vulnerability being exploited given that it is both not disclosed and not patch released.

$$Pr(E|(P^{'} \cap D^{'})) = \frac{Pr[E \cap (P^{'} \cap D^{'})]}{Pr(P^{'} \cap D^{'})} = \frac{n([E \cap (P^{'} \cap D^{'})])}{n(P^{'} \cap D^{'})} \quad (5.5.2)$$

State 03

Probability of being patch released given that it is not disclosed and not exploited.

$$Pr(P|(D^{'} \cap E^{'})) = \frac{Pr[P \cap (D^{'} \cap E^{'})]}{Pr(D^{'} \cap E^{'})} = \frac{n([P \cap (D^{'} \cap E^{'})])}{n(D^{'} \cap E^{'})} \tag{5.5.3}$$

It should be noticed for estimating probabilities with respect to states 2 and 3 also we need reasonable estimates the hidden number of vulnerabilities by all the parties. We expect to address this issue in this section. Figure 5.4 below can be used to discuss this further. Figure 5.4 illustrates two additional states (states 11 and 12) that we

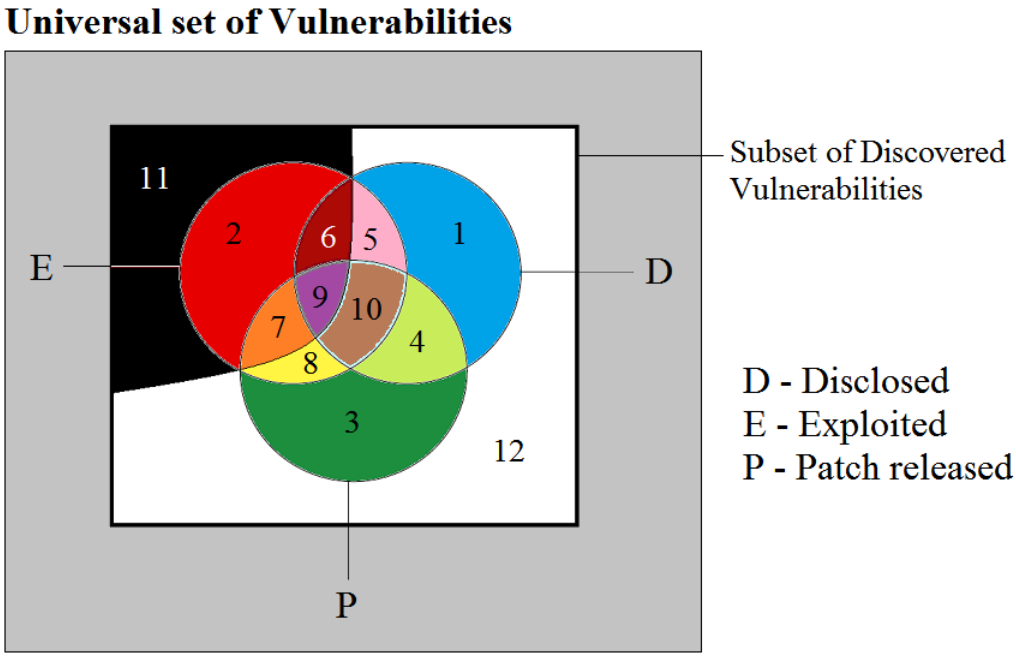

Figure 5.4: Explicit Venn diagram considering the nature of intention of the discoverer stating discovered but unknown vulnerabilities by attackers and vendors.

didnt discuss before. These states represents the following subsets of vulnerabilities.

State 11

Vulnerabilities that are discovered with bad intentions (by hackers or attackers) but has not yet been exploited or disclosed by any other means to the public or vendors.

State 12

Vulnerabilities that are discovered with good intentions (by vendors) but has not yet been patch released or disclosed by any other means to the public or vendors.

These two states (11 and 12) are of the unknown information to us. But, for our statistical inferences of the interest it is crucial to obtain some reasonable estimates with respect to these two states. To calculate conditional probabilities stated above with respect to states 1, 2 and 3 we have to have those estimates. Quality inferences on State 2 is of extreme importance in Cybersecurity. State 2 represents a Danger Zone where the attackers/hackers are in lead and execute exploits unknown to vendors first, and even after the exploit is executed and become known to vendors, no successful patches available.

5.5.2 Probabilities and inferences regarding intersected states

States 4, 5, 6, 7, 8, 9, and 10 represents intersecting states of main events. These events and their likelihood is discussed in brief below.

State 4

Probability of a discovered vulnerability observed at random being disclosed and patched released given that it is not exploited can be given as follows.

$$Pr((P \cap D)|E^{'}) = \frac{Pr[(P \cap D) \cap E^{'}]}{Pr(E^{'}} = \frac{n([(P \cap D) \cap E^{'}])}{n(E^{'}} \tag{5.5.4}$$

To obtain an estimate for this probability we need data or suitable estimates for number of elements in states 1, 3, 4, 11 and 12. Estimates for numbers of vulnerabilities in states 1, 3 and 4 can be obtained using available vulnerability data. However, obtaining proper estimates for discovered but hidden vulnerabilities is a challenge as we discussed above. Similar challenge is present in other states with intersections. Therefore, it is clear that estimates for parameters with respect to States 11 and 12 is of the importance. States 5 to 10 are given below with their corresponding probabilities.

State 5

Probability of being exploited given that it is disclosed and not patch released

$$Pr(E|(P^{'} \cap D)) = \frac{Pr[E \cap (P^{'} \cap D)]}{Pr(P^{'} \cap D)} = \frac{n([E \cap (P^{'} \cap D)])}{n(P^{'} \cap D)} \tag{5.5.5}$$

State 6

Probability of being disclosed given that it is exploited but not patched released

$$Pr(D|(P^{'} \cap E)) = \frac{Pr[D \cap (P^{'} \cap E)]}{Pr(P^{'} \cap E)} = \frac{n([D \cap (P^{'} \cap E)])}{n(P^{'} \cap E)} \tag{5.5.6}$$

State 7

Probability of being patch released given that it is not disclosed but exploited

$$Pr(P|(D^{'} \cap E)) = \frac{Pr[P \cap (D^{'} \cap E)]}{Pr(D^{'} \cap E)} = \frac{n([P \cap (D^{'} \cap E)])}{n(D^{'} \cap E)} \tag{5.5.7}$$

State 8

Probability of being exploited given that it is not disclosed but patch released

$$Pr(E|(P \cap D^{'})) = \frac{Pr[E \cap (P \cap D^{'})]}{Pr(P \cap D^{'})} = \frac{n([E \cap (P \cap D^{'})])}{n(P \cap D^{'})} \tag{5.5.8}$$

State 9

Probability of being patch released and disclosed given that it is exploited

$$Pr((P \cap D)|E) = \frac{Pr[(P \cap D) \cap E]}{Pr(E} = \frac{n([(P \cap D) \cap E])}{n(E} \tag{5.5.9}$$

State 10

Probability of being exploited given that it is both disclosed and patch released

$$Pr(E|(P \cap D)) = \frac{Pr[E \cap (P \cap D)]}{Pr(P \cap D)} = \frac{n([E \cap (P \cap D)])}{n(P \cap D)} \tag{5.5.10}$$

While, all these states represents different subspaces in the vulnerability space, there are some sates of higher importance than the others. It should be clear that, most

dangerous states are those with Exploited event. States, 2, 5, 6, 7, 8, 9 and 10 are all related to Exploitation in some way. However, States coming into exploitation with a Vendor leading state is of the knowledge for the vendor and defending professionals. That is, states 8 and 10 are actually discovered by the vendor, but for some reason there are possibilities for exploitations. However, these threats are known to the vendor and necessary steps can be taken to avoid further damages. But, states 2, 6, 7 and 9 represents a subspace with Attackers Lead. All vulnerabilities in these states are first discovered by attackers. Exploitation occurs unknown to the vendors. Vendors might know about these sets of vulnerabilities only after attacks are executed and severe damages occurred. Therefore, these states represents a real threat. Out of these four states, states 7 and 9 represents the vulnerabilities where, after exploitation somehow, a patch is developed and released therefore, no further damages can be generated from those vulnerabilities. But, states 2 and 6 are extreme situations. These two states represents vulnerabilities in attackers lead and there are no solutions for them at all. Therefore, we consider states 2 and 6 as Danger Zones in vulnerability space. State 5 is also considered a Danger Zone because it represents the set of vulnerabilities that has been exploited after the discloser. Even though it is known to the vendor, such a vulnerability has no Patch available yet. Therefore, this is also a high threat situation.

5.6 Danger Zones of the Vulnerability Space

As discussed in the previous section states 2, 5 and 6 are the most dangerous zones considering the strong position that attackers possess with respect to vulnerabilities in those two states. Vulnerabilities in state 2 are discovered by attackers and then execute exploits, breach systems where the vendors have no knowledge of them at all. Even after disastrous attacks, vendors are unable to develop a patches and release them to users. Users have no idea, because these vulnerabilities are not disclosed to the public too. State 6 has the same scenario except that they are disclosed to the public.

Considering this situation, analysis of these two Danger Zones are of higher importance. To obtain better inferences regarding these states, there is a need for a powerful and comprehensive data resources which we do not have access at this point. However, using available data resources we analyze these states and expects to obtain useful observations to develop better predictive models.

Table 5.3 categorize the vulnerability data set contains 46310 vulnerabilities since 1998 to 2011 given presented by M. Shahzad [21] in his research paper.

As we are interested in State 6, which is an intersection of Exploitation and Discloser, lets consider the behavior of 15363 vulnerabilities from 1998 to 2011 recorded with respect to this state. The time gap between the discloser and exploitation calculated in days regarding these vulnerabilities are given in the Table 5.4 below. Let ted represent the difference between exploitation time and disclosed time of a specific vulnerability. As the table illustrates some of the vulnerabilities have been even

Table 5.3: categorize the vulnerability data set (1998 to 2011).

Category of information available	of Vulnerabilities
Disclosed date	46310
Patched date with Disclosed date	9667
Exploited date with Disclosed date	15363
Both Patched and Exploitation date	1424

Table 5.4: Number of vulnerabilities calculated considering the time gap between discloser and exploitation.

Year	Vulnerabilities $t_{ed} < 0$	Vulnerabilities $t_{ed} = 0$	Vulnerabilities $0 < t_{ed} < 7$	Vulnerabilities $7 < t_{ed} < 30$	Vulnerabilities $t_{ed} > 30$
98	2	39	1	-	1
99	6	147	1	-	2
00	5	226	3	-	9
01	9	256	12	2	12
02	6	532	50	6	25
03	19	343	68	19	34
04	88	1177	132	37	37
05	58	1905	133	58	61
06	60	2569	272	60	61
07	40	1705	159	38	40
08	14	2726	14	14	15
09	9	1358	12	10	11
10	12	545	25	15	15
11	-	40	1	2	1

exploited before the discloser date. Many vulnerabilities have been exploited on the date of discloser itself. Within first week, after the first week within a month and after a month exploitations are calculated and presented. $t_ed < 0$ shows that exploit for a given vulnerability is released before its publicly disclosed. $t_ed = 0$ shows that exploit for a given vulnerability is released on the day its publicly disclosed. $t_ed > 0$ shows that exploit for a given vulnerability is released after its publicly disclosed. As

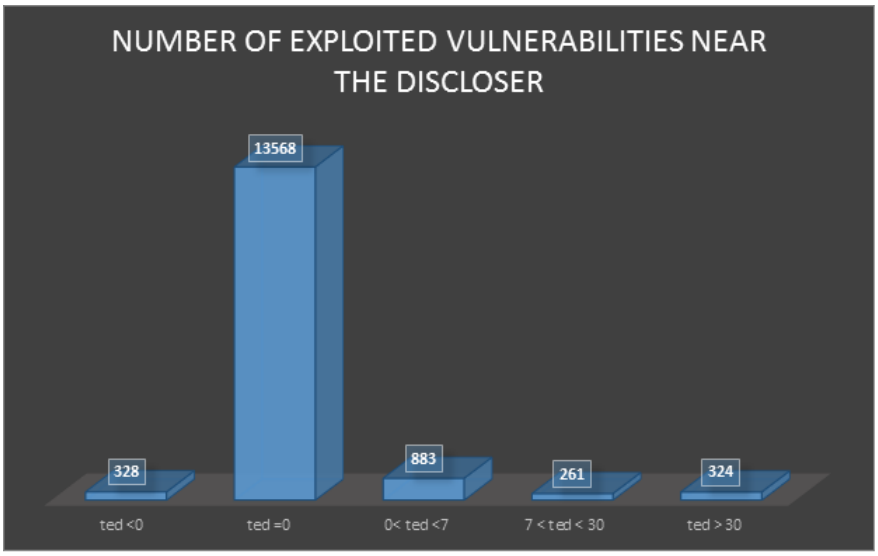

Figure 5.5: Number of Exploited Vulnerabilities near the Discloser.

illustrated in the Figure 5.5, it is clear that the number of vulnerabilities that is getting exploited on the day of the discloser is way higher than other times. This is a peculiar observation. The reason however is clear. Once a vulnerability is disclosed, many attackers or hackers will immediately identify it and create an exploit. This indeed shows the capacity of and skill level of modern hackers. All this same day and after discloser exploitations comes under Sate 5 in the vulnerability space illustration in Figure 5.4. There are 328 vulnerabilities that has been exploited before the discloser.

These exploitations comes under State 06 which we consider a Danger Zone where users have no knowledge at the time of exploitation.

Now, lets put our specific attention on this 328 vulnerabilities which were exploited before the discloser. We can calculate the probability of such a vulnerability out of the subspace of vulnerabilities recorded for both the discloser date and exploited date. All these include in the area (States 5 and 6) of the intersection of discloser and exploitation. The scatter plot in the Figure 5.6, of the probabilities in the each year does not show any specific behavior.

Even though we dont have sufficient data points for a quality analysis, it is

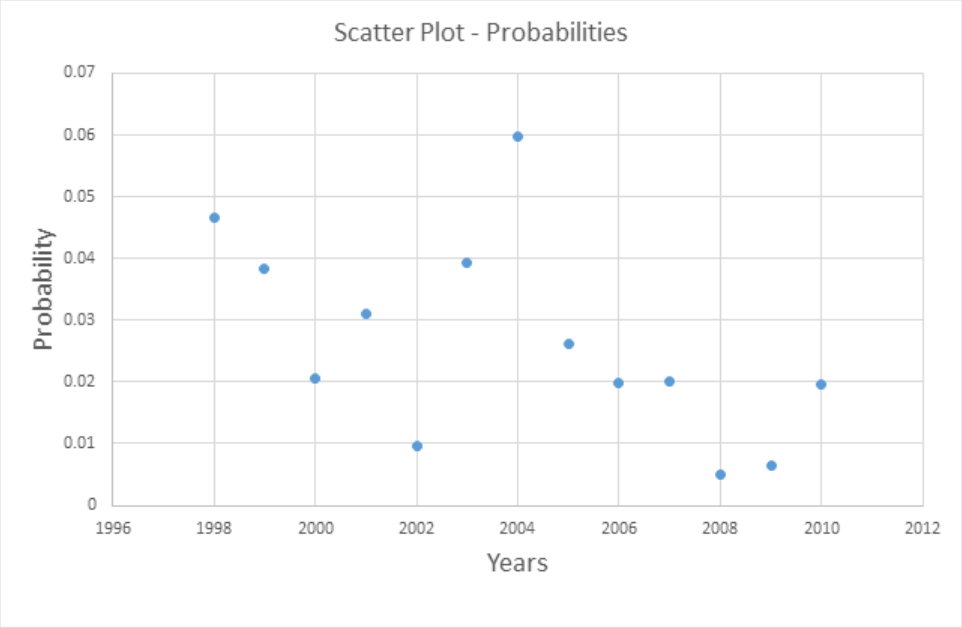

Figure 5.6: Scatter Plot of Probabilities of Vulnerability exploited before the discloser.

clear from this scatter plot that the probability behavior of these vulnerabilities looks a non-stationary. Proper analyses on these critical vulnerabilities and their behavior is very important. But, the data we possess at the point is not sufficient for such a

study. However, from the data recorded in other sources such as Secunia Vulnerability Reports [2] it shows a significant increase in the Zero Day Vulnerabilities in recent years. Therefore, studies on these vulnerabilities will become more important in Cybersecurity in the future.

5.7 Contributions

In this chapter, we have conducted a comprehensive analysis on the Vulnerability Space that we introduced in chapter two. We developed a new approach using Venn diagrams to analyze all the states of vulnerabilities in the vulnerability space. This presentation of vulnerabilities as a probability space and analysis of events and states lay down a strong foundation for understanding the behavior of vulnerabilities in different circumstances. Relationship of events occur in the vulnerability space with corresponding states clarify many issues in understanding the behavior of vulnerabilities.

6 Introduction to Extended Vulnerability Life Cycle Model

6.1 Introduction

In this chapter, we integrate our results from all previous chapters. Vulnerability Life Cycle model that we used in chapter 3 and 4 will be further extended to develop a complete Life Cycle Model. The Life Cycle graph that we present will incorporate all the states and events we discussed in this study with respect to vulnerability space. The Venn diagrams we presented in chapter 5, in addition to illustrating probabilities of the events and states of the vulnerability space, provides us a better approach for an extended Vulnerability Life Cycle. However, a complex Vulnerability Life Cycle with many nodes might be hard to use analytically. Therefore, we may present a Master Vulnerability Life Cycle for the completion of the concept and discuss Sub-Life Cycles based on the practical and analytical needs. That way, according to the need of a study anyone can extract Sub-Life Cycles and use them analytically.

6.2 Extended Vulnerability Life Cycle

Vulnerability Life Cycle that we proposed in chapter 3 in the present study had five states. Those states were given as Non Discovered, Discovered, Exploited, Patched and Disclosed but not patched or exploited. That life cycle graph covered the main events in a vulnerability life cycle. However, as we understood in chapters 2 and 5 it is clear that there are many other important events and corresponding states in a vulnerability life cycle. Therefore, there is a need for better and comprehensive Vulnerability Life Cycle Model which is capable of illustrating as many events and states of vulnerability space as possible.

Very first consideration in our new approach is to separately consider who discovers a vulnerability. That is in other words, in developing our new life cycle graph, the intention of the discoverer is explicitly considered. It is clear that the resulting events are highly depending on the intention of the discoverer of the vulnerability.

Figure 6.1 below presents our new Extended Life Cycle graph. This new Life Cycle graph has 11 states. At first, the Birth of a vulnerability is considered. As any man made software could have a weakness, it is considered that vulnerabilities are born at the introduction of the software itself.

Next states are about the discovery. In this Life Cycle, instead of one state to represent the discovery, we allocate two different states representing the discoverers intention also. Therefore, the discovery of a vulnerability by a White Hat hacker (that is with good intentions) is considered separately from the discovery by an attacker or Black Hat hacker (that is with bad intentions).

After that, probable events are identified separately. Discussing this Vulnerability Life Cycle graph with the Venn Diagram of the Vulnerability space will give us a clear understanding.

As Figure 6.1 illustrates, if a vulnerability is discovered by a White Hat

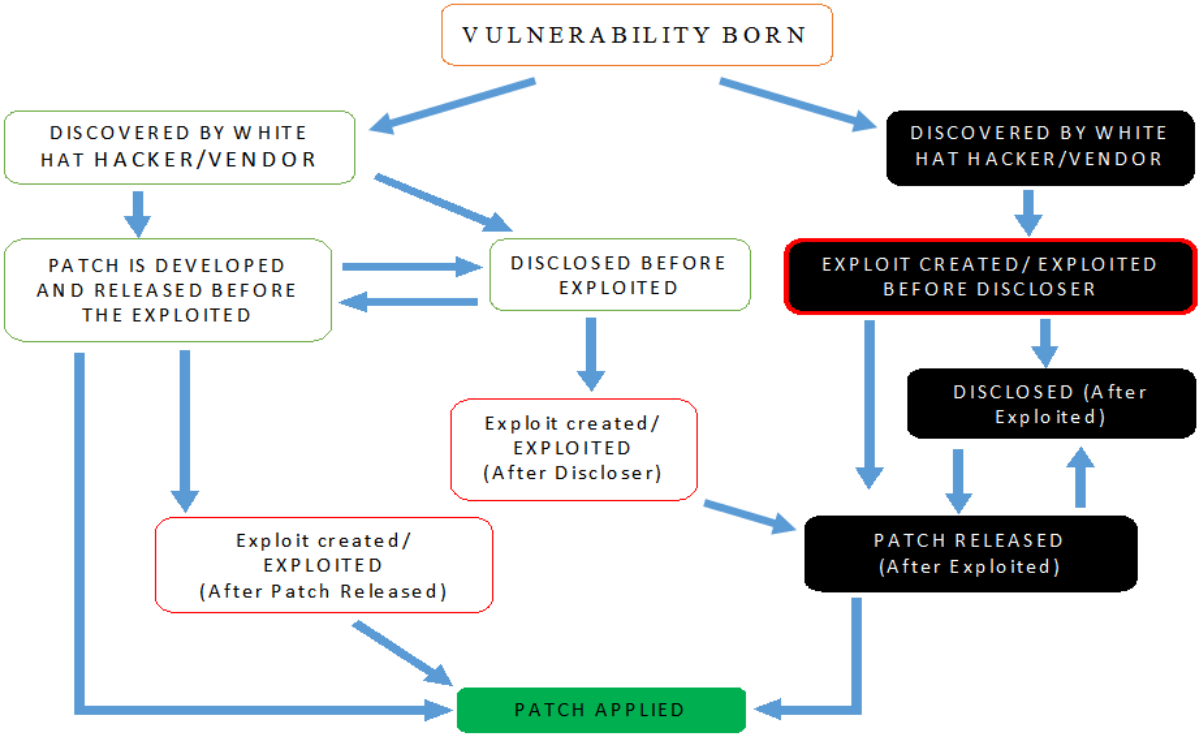

Figure 6.1: Extended Vulnerability Life Cycle.

hacker or Vendor parties, the very first effort will be to develop a successful patch and release it. But, some times, patching vulnerabilities at once could not be feasible. In such cases, there is a possibility that the vulnerability is announced without a patch. According to the Secunia Report of 2016 Vulnerability Review, out of detected 17147 vulnerabilities in 2016, there are about 19 percent which did not have a patch at the date of Discloser. However, if the vulnerability is discovered by the vendors and they

successfully develop the patch and release it, the probability of that vulnerability being exploited becomes negligible. Such an event is only probable if a particular user block the software to get updated with the released Patch. Sometimes, a vulnerability discovered by the vendor parties or White Hat Hackers and disclosed it to the public without a patch, it is possible that Black hat Hackers will develop an exploit code and execute it.

On the other hand, if the vulnerability is discovered by Black Hat Hackers (with bad intentions) probable events are more likely harmful or disastrous. Vulnerabilities unknown but to these hackers with an exploit is developed are a major threat to Cybersecurity. In recent year, a recorded trend of increasing numbers of Zero Day vulnerabilities are observed. Once attacks are taken place and successful exploitations occurs, vendors might get to know about the vulnerability and then tries to develop a patch. Once the Patch is available, it will be released and installed. In our new approach of the Vulnerability Life Cycle all these probable events are considered and represented with States and paths through them.

6.3 Relationship of the Extended Vulnerability Life Cycle and States and Events of the Vulnerability Space

Figure 6.2 illustrates the same Vulnerability Life Cycle with respective initial probabilities assigned. Lets discuss this Life Cycle with the Venn diagram of the vulnerability space given in Figure 6.3.

First, as mentioned in Figure 6.3, the Venn diagram, States 2, 6, 7, 9 and 11 represents

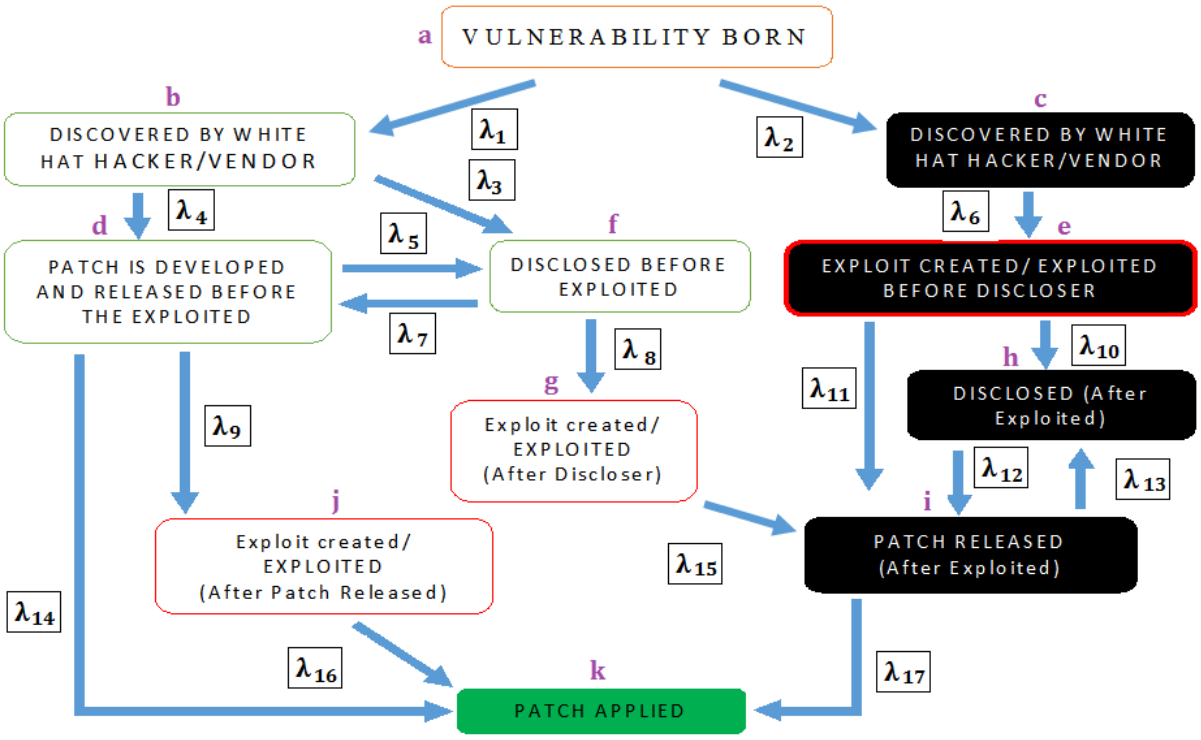

Figure 6.2: Extended Vulnerability Life Cycle with assigned probabilities for transitions.

vulnerabilities discovered by Black Hat Hackers or attackers. States c, e, h and i in Figure 6.2 illustrates events corresponding to those states in the Vulnerability Space. Similarly, states 1, 3, 4, 5, 8, 10 and 12 represent the sunset of vulnerabilities discovered by White Hat Hackers or vendor parties. The process of generating these states in the Vulnerability Life Cycle is illustrated by States b, d, f, g, i and k. However, it should be noted that state k in the Life Cycle Model represents the process of exploited vulnerabilities getting patched too.

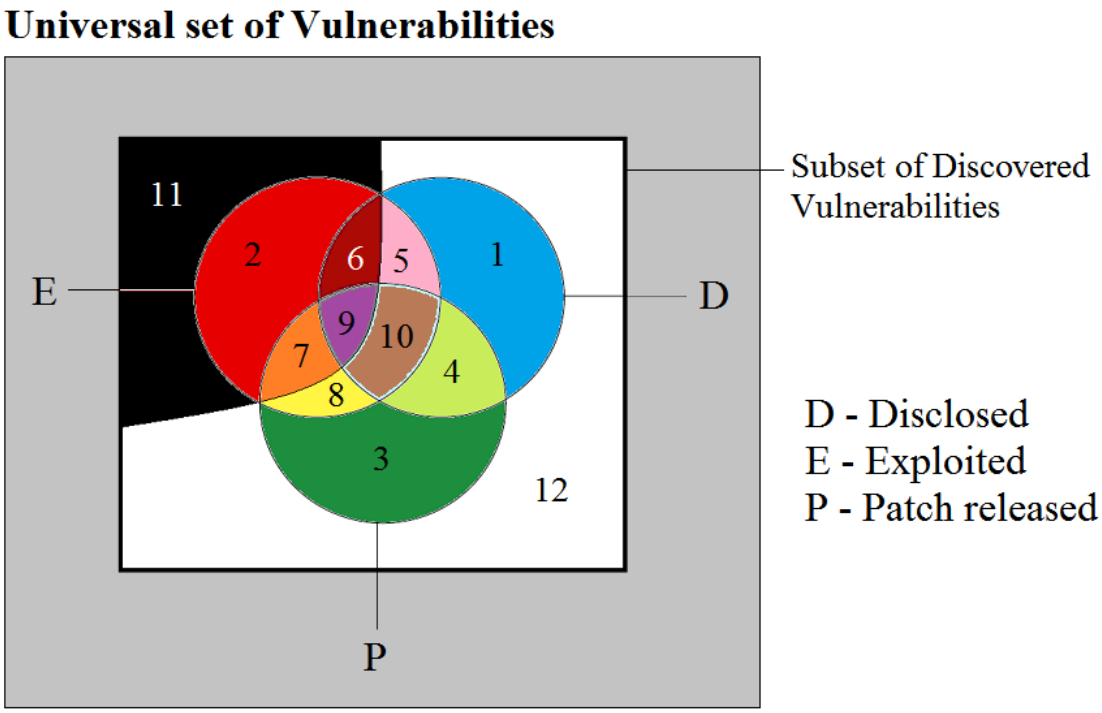

Figure 6.3: Explicit Venn diagram considering the nature of intention of the discoverer stating discovered but unknown vulnerabilities by attackers and vendors.

6.3.1 Relationship among Initial Probabilities, corresponding events and States

λ_1 - Probability of the event an undiscovered Vulnerability being discovered by a White Hat Hacker or Vendor Parties. Subset of vulnerabilities generated from this transition of the Vulnerability Life Cycle is represented by the sum of states 1, 3, 4, 5, 8, 10 and 12 of the Vulnerabilities Space.

λ_2 - Probability of the event an undiscovered Vulnerability being discovered by an attacker or Black Hat Hacker. Subset of vulnerabilities generated from this transition

of the Vulnerability Life Cycle is represented by the sum of states 2, 6, 7, 9 and 11 of the Vulnerabilities Space.

λ_3 - Probability of the event a Vulnerability discovered by a White Hat Hacker or Vendor Parties being disclosed before exploited without a patch released. Subset of vulnerabilities generated from this transition of the Vulnerability Life Cycle is represented by the state 1 of the Vulnerabilities Space.

λ_4 - Probability of the event a Vulnerability discovered by a White Hat Hacker or Vendor Parties being released with a patch before exploited and before disclosed. Subset of vulnerabilities generated from this transition of the Vulnerability Life Cycle is represented by the state 3 of the Vulnerabilities Space.

λ_5 - Probability of the event a Vulnerability discovered by a White Hat Hacker or Vendor Parties being disclosed before exploited after the patch is released. Subset of vulnerabilities generated from this transition of the Vulnerability Life Cycle is represented by the state 4 of the Vulnerabilities Space.

λ_6 - Probability of the event a Vulnerability discovered by an attacker being exploited before disclosed and before patch is released. Subset of vulnerabilities generated from this transition of the Vulnerability Life Cycle is represented by the state 2 of the Vulnerabilities Space.

λ_7 - Probability of the event a Vulnerability discovered by a White Hat Hacker or Vendor Parties being patch is released before exploited after the discloser. Subset of vulnerabilities generated from this transition of the Vulnerability Life Cycle is represented by the state 4 of the Vulnerabilities Space.

λ_8 - Probability of the event a Vulnerability discovered by a White Hat Hacker or Vendor Parties being exploited before patch is released after discloser. Subset of vulnerabilities generated from this transition of the Vulnerability Life Cycle is represented by the state 5 of the Vulnerabilities Space.

λ_9 - Probability of the event a Vulnerability discovered by a White Hat Hacker or Vendor Parties being exploited after patch is released before discloser. Subset of vulnerabilities generated from this transition of the Vulnerability Life Cycle is represented by the state 8 of the Vulnerabilities Space.

λ_{10} - Probability of the event a Vulnerability discovered by an attacker being disclosed after exploited and before patch is released. Subset of vulnerabilities generated from this transition of the Vulnerability Life Cycle is represented by the state 6 of the Vulnerabilities Space.

λ_{11} - Probability of the event a Vulnerability discovered by an attacker being patch released after exploited and before disclosed. Subset of vulnerabilities generated from this transition of the Vulnerability Life Cycle is represented by the state 7 of the Vulnerabilities Space.

λ_{12} - Probability of the event a Vulnerability discovered by an attacker being patch released after exploited and disclosed. Subset of vulnerabilities generated from this transition of the Vulnerability Life Cycle is represented by the state 9 of the Vulnerabilities Space.

λ_{13} - Probability of the event a Vulnerability discovered by an attacker being disclosed after exploited and patch released after. Subset of vulnerabilities generated from

this transition of the Vulnerability Life Cycle is represented by the state 9 of the Vulnerabilities Space.

λ_{14} - Probability of the event a Vulnerability discovered by a White Hat Hacker or Vendor Parties dies before discloser and before exploited. Complete death of a vulnerability is an ideal phenomena and a theoretical definition. That is the state where all the computers where the software was installed and operated is installed with the patch. Such an ideal state is only assumed here for the analytical completeness. This event is not represented in our Venn diagram of the Vulnerability Space since the event actually means that a vulnerability is eliminated form entire space of vulnerabilities.

λ_{15} - Probability of the event a Vulnerability discovered by a White Hat Hacker or Vendor Parties being patch released after discloser and exploited. Subset of vulnerabilities generated from this transition of the Vulnerability Life Cycle is represented by the state 10 of the Vulnerabilities Space.

λ_{16} - Probability of the event a Vulnerability discovered by a White Hat Hacker or Vendor Parties dies after patch released and after exploited. Complete death of a vulnerability is an ideal phenomena and a theoretical definition. That is the state where all the computers where the software was installed and operated is installed with the patch. Such an ideal state is only assumed here for the analytical completeness. This event is not represented in our Venn diagram of the Vulnerability Space since the event actually means that a vulnerability is eliminated form entire space of vulnerabilities.

λ_{17} - Probability of the event a Vulnerability discovered by an attacker dies after

discloser, exploited and patch released. Complete death of a vulnerability is an ideal phenomena and a theoretical definition. That is the state where all the computers where the software was installed and operated is installed with the patch. Such an ideal state is only assumed here for the analytical completeness. This event is not represented in our Venn diagram of the Vulnerability Space since the event actually means that a vulnerability is eliminated form entire space of vulnerabilities.

6.4 Markov Approach and Transition Probability Matrix for the Extended Vulnerability Life Cycle

Now that we have presented and discussed the Extended Vulnerability Life Cycle, we can theoretically apply Markovian process and develop Transition Probability Matrix using the same process that we presented in chapter 3 of the present study. However, it should be noted that, in real world scenario, obtaining necessary information with respect estimating initial probability parameters is beyond the reality. Even though there are sound statistical methodologies that can be applied, obtaining needed data sources is a challenge for us. While the National Vulnerability Database and CVE detail website maintain many vulnerability and update, it is recorded from other sources that lots of detected vulnerabilities are actually missing in the database. Data available for us at this point do not possess all the aspects of our consideration. But, what we discussed in this this chapter and chapter 5 could present a strong foundation for any future analysis on vulnerability data. There is no doubt that, organizations and the governments will be compelled to take necessary steps for better and effective

data collection, integration and management. With such available data recourses this comprehensive Life Cycle Model for vulnerabilities will provide better indications and information that will be useful for decision making.

Conducting the same process discussed in chapter 3, here 11 x 11, Transition Probability Matrix for the Extended Vulnerability Life Cycle model with given initial probabilities is given below. The Matrix presents all 10 transient states and one absorbing states.

$$P = \begin{pmatrix} 0 & \lambda_1 & \lambda_2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \lambda_4 & 0 & \lambda_3 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \lambda_6 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \lambda_5 & 0 & 0 & 0 & \lambda_9 & \lambda_{14} \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \lambda_{10} & \lambda_{11} & 0 & 0 \\ 0 & 0 & 0 & \lambda_7 & 0 & 0 & \lambda_8 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \lambda_{15} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \lambda_{12} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \lambda_{13} & 0 & 0 & \lambda_{17} \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \lambda_{16} \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \tag{6.4.1}$$

We can maintain the same assumptions for this Matrix and iterate it until the steady state is achieved. Resulting vectors of probabilities with respect to different states such as Exploitation and Patch release with variations presented in the life

cycle will give abundance of useful information about a vulnerability as a function of time. In addition relevant Risk Factors and other indices can be defined accordingly.

6.5 Contributions

In this chapter, comprehensive Extended Vulnerability Life Cycle was presented. This Life Cycle model along with the Vulnerability Space developed and presented in the previous chapter, will lay down a strong foundation for any kind of Cybersecurity and vulnerability related study. Present chapter also have discussed relationship of the life cycle model with events and states in the Vulnerability Space. Finally, applicability of the Markov approach and resulting Transition Probability Matrix is presented and the applicability is therefore established given that the necessary data are obtained.

7 Future Research

Further to the results obtained in the present study, in the future, it is expected to develop a statistical model that incorporates all the risk factors that identifies the vulnerability (vulnerability scores as the response variable) of a given software (or a computer system). Having developed such a statistical model, we will be able to statistically identify the vulnerability as a function of contributable variables that drive a given operating system. Proposed future model, not only will be able to predict the vulnerabilities, but also would identify possible interactions and rank the attributable variables with respect to their contributions into the vulnerability of a given operating system.

Additionally, it is important to conduct analytical studies to evaluate the design of the experiments with respect to the manner by which NVD uses their scaling process to identify the initial vulnerability command. Secondly it is important to analyze the information from the vulnerability levels entering into two equations that NVD has developed and uses as shown in Appendix A. We need to evaluate the effectiveness of the final vulnerability scores and the process of obtaining those scores using the equations about the Base score. The current state is that the process is

descriptive and we cannot obtain a degree of confidence on how accurate is this vulnerability score that characterizes the behavior of an operating software with the data available.

Software reliability is also extremely important in obtaining acceptable reliability that a given software with discovered vulnerabilities could maintain in response to the vulnerability space. We expect to propose a parametric software reliability analysis utilizing the vulnerability scores as and vulnerability states as failure time for a given software system. Preliminary results indicate that the parameters that drive the probabilistic behavior of vulnerabilities behave randomly. Thus, a Bayesian software reliability is applicable in such cases if we can statistically justify its applicability.

Application of Markov process and other better statistical methodologies on our new Extended Vulnerability Life Cycle Model is of our key expectations. We expect to obtain necessary data from the relevant authorities in conducting combined analytical studies further.

References

[1] NVD, National vulnerability database, http://nvd.nist.gov/.

[2] Secunia Vulnerability Review 2015: Key figures and facts from a global information security perspective, March 2015.

[3] 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.

[4] S. Frei, Security Econometrics: The Dynamics of (IN) Security, Ph.D. dissertation at ETH Zurich, 2009.

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

[6] CVE details. http://www.cvedetails.com/

[7] S.Abraham and S.Nair, "Cyber Security Analytics: A stochastic model for Security Quantification using Absorbing Markov Chains" Journal of Communications Vol. 9, No. 12, December 2014, pp. 899-907.

[8] Phongphun Kijsanayothin, (2010) Network Security Modeling with Intelligent and Complexity Analysis. Ph.D. Dissertation, Texas Tech University.

[9] 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. 219228, May 2007.

[10] Alhazmi, O. H. and Malaiya, Y. K. (2008) Application of Vulnerability Discovery Models to Major Operating Systems, IEEE Transactions on Reliability, Vol. 57, No. 1, 2008, pp. 14-22.

[11] Joh, H. and Malaiya, Y.K. (2010) A framework for Software Security Risk Evaluation using the Vulnerability Lifecycle and CVSS Metrics, Proc. International Workshop on Risk and Trust in Extended Enterprises, November 2010,pp.430-434.

[12] Leyla Bilge, Tudor Dumitras, An Empirical Study of Zero-Day Attacks in the real world. ACM's Conference on Computer and Communications Security, Oct. 16-18, 2012.

[13] 2016 U.S Government Cybersecurity report. https://cdn2.hubspot.net/hubfs/533449/

[14] H. Joh, J. Kim, and Y. K. Malaiya, Vulnerability Discovery Modeling Using Weibull Distribution, 2008, pp. 299300.

[15] S. Noel, M. Jacobs, P. Kalapa, and S. Jajodia. Multiple Coordinated Views for Network Attack Graphs. In VIZSEC'05: Proc. of the IEEE Workshops on

Visualization for Computer Security, Minneapolis, MN, October, 2005, pages 99106.

[16] 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.

[17] Mehta, V., C. Bartzis, H. Zhu, E. M. Clarke, and J. M. Wing (2006). Ranking attack graphs. In D. Zamboni and C. Kr ugel (Eds.), Recent Advances in Intrusion Detection, Volume 4219 of Lecture Notes in Computer Science, pp. 127144. Springer.

[18] M. Zhao, J. Grossklags, and K. Chen, An Exploratory Study of White Hat Behaviors in a Web Vulnerability Disclosure Program, 2014, pp. 5158.

[19] T. Bass, Intrusion detection system and multi-sensor data fusion, Communications of the ACM, vol. 43, no. 4 pp. 99-105, 2000.

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

[21] M. Shahzad, M. Z. Shafiq, and A. X. Liu. A large scale exploratory analysis of software vulnerability life cycles. In Proceedings of the 2012 International Conference on Software Engineering, 2012.

[22] L. Wang, A. Singhal, and S. Jajodia, "Measuring overall security of network configurations using attack graphs," Data and Applications Security XXI, vol. 4602, pp. 98-112, August 2007.

[23] L. Wang, T. Islam, T. Long, A. Singhal, and S. Jajodia, "An attack graph-based probabilistic security metric," DAS 2008, LNCS 5094, pp. 283-296, 2008.

[24] 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. http://dx.doi.org/10.4236/jis.2016.73008

[25] 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. http://dx.doi.org/10.4236/jis.2016.74022

[26] Rajasooriya, S. , Tsokos, C. and Kaluarachchi, P. (2017) Cyber Security: Nonlinear Stochastic Models for Predicting the Exploitability. Journal of Information Security, 8, 125-140. doi: 10.4236/jis.2017.82009.

[27] Symantec, Internet security threat report 2016-Volume 21, https://resource.elq.symantec.com

[28] Gleich, David F. (January 2015). "PageRank Beyond the Web". SIAM Review. 57 (3): 321363.

[29] Hewett, R. and P. Kijsanayothin (2008). Host-centric model checking for network vulnerability analysis. In ACSAC 08: Proceedings of the 2008 Annual Computer Security Applications Conference, Washington, DC, USA, pp. 225234. IEEE Computer Society.

[30] S.Abraham and S.Nair, "A Stochastic Model for Cyber Security Analytics" Tech

Report 13-CSE-02, CSE Dept, Southern Methodist University, Dallas, Texas, 2013.

[31] R. Sawilla and X. Ou. Googling Attack Graphs. Techni-cal Report TM-2007-205, Defense Research and Devel-opment Canada, September 2007.

[32] D. Geer and J. Harthorne. Penetration Testing: A Duet. In ACSAC'02: Proc. of the 18th Annual Computer Security Applications Conference, page 185, Washington, DC, USA, 2002. IEEE Computer Society.

[33] S. Noel and S. Jajodia. Understanding Complex Network Attack Graphs through Clustered Adjacency Matrices. In ACSAC '05: Proceedings of the 21st Annual Computer Security Applications Conference, pages 160-169, Washington, DC, USA, 2005. IEEE Computer Society.

[34] M. Stamp (2011). Information Security: Principles and Practice, second edition, Hoboken, NJ: Wiley-Interscience.

[35] I. Mkpong-Ruffin, D. Umphress, J. Hamilton, and J. Gilbert, Quantitative software security risk assessment model, ACM workshop on Quality of protection, 2007, pp. 3133.

[36] Izenman, A. (2008). Modern multivariate statistical techniques (Vol. 1). New York: Springer.

[37] Hansen, B. E. (2009). Lecture notes on nonparametrics. Lecture notes.

[38] Ou, X., Boyer, W., McQueen, M.: A Scalable Approach to Attack Graph Generation. ACM (2006)

[39] J. Jones, Estimating software vulnerabilities, Security & Privacy,IEEE, vol. 5, no. 4, pp. 2832, July-Aug. 2007.

[40] H. Okamura, M. Tokuzane and T. Dohi, "Quantitative Security Evaluation for Software System from Vulnerability Database," Journal of Software Engineering and Applications, Vol. 6 No. 4A, 2013, pp. 15-23. doi: 10.4236/jsea.2013.64A003.

[41] Ingols, K., Lippmann, R., Piwowarsi, K.: Practical attack graph generation for network defense. In: 22nd Annual Conference on Computer Security Application, pp. 121130 (2006)

[42] Hongzhu Qiao, Chris P.Tsokos, Best efficient estimates of the intensity function of the power law process. Journal of Applied Statistics, vol. 25, No. 1, pp 111-120, 1998

[43] Mell, P., Scarfone, K. and Romanosky, S. (2007) A Complete Guide to the Common Vulnerability Scoring System Version 2.0. FIRST-Forum of Incident Response and Security Teams, 1-23.

[44] Bilge, L.and Dumitras, T. (2012) Before We Knew It: An Empirical Study of Zero- Day Attacks in the Real World. Proceedings of the 2012 ACM Conference on Computer and Communications Security, Raleigh, 16-18 October 2012, 833-844.

[45] Krsul, I.V. (1998) Software Vulnerability Analysis. Doctoral Dissertation, Purude University, Indiana.

[46] Jha, S., Sheyner, O. and Wing, J. (2002) Two Formal Analyses of Attack Graphs. Proceedings of 15th IEEE Computer Security Foundations Workshop, Cape Breton, 24-26 June 2002, 49-63.

[47] S. Frei, M. May, U. Fiedler, and B. Plattner, Large-scale vulnerability analysis, in Proceedings of the 2006 SIGCOMM workshop on Large-scale attack defense, 2006, pp. 131138.

[48] Bolch, G., Greiner, S., de Meer, H. and Trivedi, K.S. (2006) Queueing Networks and Markov Chains: Modeling and Performance Evaluation with Computer Science Applications. John Wiley & Sons, Somerset.

[49] Trivedi, K.S. (2002) Probability & Statistics with Reliability, Queuing and Computer Science Applications. John Wiley & Sons, New Work.

[50] R statistics tool. http://www.r-project.org

Appendices

Appendix A - Common Vulnerability Scoring System (CVSS)Version 2.0 Calculations

Scoring equations and algorithms for the base, temporal and environmental metric groups are described below. Further discussion of the origin and testing of these equations is available at www.first.org/cvss.

The **Base equation** is the foundation of CVSS scoring. The base equation is:

```
BaseScore = (((0.6*Impact)+(0.4*Exploitability)1.5)*f(Impact))

Impact = 10.41*(1-(1-ConfImpact)*(1-IntegImpact)*(1-AvailImpact))

Exploitability = 20* AccessVector*AccessComplexity*Authentication


f(impact)= 0 if Impact=0, 1.176 otherwise


AccessVector (AV) = case AccessVector of

requires local access: 0.395

adjacent network accessible: 0.646

network accessible: 1.0


AccessComplexity (AC) = case AccessComplexity of

high: 0.35

medium: 0.61

low: 0.71
```

```
Authentication (AU) = case Authentication of

requires multiple instances of authentication: 0.45

requires single instance of authentication: 0.56

requires no authentication: 0.704


ConfImpact (C)= case ConfidentialityImpact of

none: 0.0

partial: 0.275

complete: 0.660


IntegImpact (I) = case IntegrityImpact of

none: 0.0

partial: 0.275

complete: 0.660


AvailImpact (A)= case AvailabilityImpact of

none: 0.0

partial: 0.275

complete: 0.660
```

Appendix A (Continued)

Temporal Equation

If employed, the temporal equation will combine the temporal metrics with the base score to produce a temporal score ranging from 0 to 10. Further, the temporal score will produce a temporal score no higher than the base score, and no less than 0.33 lower than the base score. The temporal equation is:.

```
TemporalScore = (BaseScore*Exploitability*RemediationLevel*ReportConfidence)

Exploitability = case Exploitability of
unproven: 0.85
proof-of-concept: 0.9
functional: 0.95
high: 1.00
not defined: 1.00

RemediationLevel = case RemediationLevel of
official-fix: 0.87
temporary-fix: 0.90
workaround: 0.95
unavailable: 1.00
not defined: 1.00
```

```
ReportConfidence = case ReportConfidence of

unconfirmed: 0.90

uncorroborated: 0.95

confirmed: 1.00

not defined: 1.00
```

Environmental Equation If employed, the environmental equation will combine the environmental metrics with the temporal score to produce an environmental score ranging from 0 to 10. Further, this equation will produce a score no higher than the temporal score. The environmental equation is:

```
EnvironmentalScore =(AdjustedTemporal+(10-AdjustedTemporal) *CDP) *TD)


AdjustedTemporal = TemporalScore recomputed with the
BaseScores Impact sub- equation replaced with the AdjustedImpact equation


AdjustedImpact = min(10,10.41*(1-(1-ConfImpact*ConfReq)*(1-IntegImpact*IntegReq)
*(1-AvailImpact*AvailReq)))


CollateralDamagePotential (CDP) = case CollateralDamagePotential of
none: 0
low: 0.1
low-medium: 0.3
medium-high: 0.4
high: 0.5
not defined: 0


TargetDistribution (TD)= case TargetDistribution of
```

```
none: 0

low: 0.25

medium: 0.75

high: 1.00

not defined: 1.00


ConfReq = case ConfReq of

low: 0.5

medium: 1.0

high: 1.51

not defined: 1.0


IntegReq = case IntegReq of

low:0.5

medium: 1.0

high: 1.51

not defined: 1.0


AvailReq = case AvailReq of

low:0.5

medium: 1.0

high: 1.51
```

```
not defined: 1.0
```

Appendix B

Example-CVE-2002-0392

Below, we provide steps of how CVSS is used for three different metric.

```
--------------------------------------------------

BASE METRIC   EVALUATION SCORE

--------------------------------------------------

Access Vector [Network] (1.00)

Access Complexity [Low] (0.71)

Authentication [None] (0.704)

Confidentiality Impact [None] (0.00)

Integrity Impact [None] (0.00)

Availability Impact [Complete] (0.66)



Impact = 10.41*(1-(1)*(1)*(0.34)) == 6.9

Exploitability = 20*0.71*0.704*1 == 10.0 f(Impact) = 1.176

BaseScore = (0.6*6.9 + 0.4*10.0  1.5)*1.176 =7.8


--------------------------------------------------

TEMPORAL METRIC   EVALUATION SCORE

--------------------------------------------------

Exploitability [Functional]     (0.95)
```

Remediation Level [Official-Fix] (0.87)

Report Confidence [Confirmed] (1.00)

TEMPORAL SCORE=round(7.8 * 0.95 * 0.87 * 1.00) == (6.4)

ENVIRONMENTAL METRIC EVALUATION SCORE

Collateral Damage Potential [None - High] {0 - 0.5}

Target Distribution [None - High] {0 - 1.0}

Confidentiality Req. [Medium] (1.0)

Integrity Req. [Medium] (1.0)

Availability Req. [High] (1.51)

AdjustedImpact = min(10,10.41*(1-(1-0*1)*(1-0*1)*(1-0.66*1.51))= (10.0)

AdjustedBase =((0.6*10)+(0.4*10.0)1.5)*1.176 = (10.0)

AdjustedTemporal = (10*0.95*0.87*1.0) = (8.3)

EnvScore = round((8.3+(10-8.3)*{0-0.5})*{0-1})= (0.00 - 9.2)

Appendix C

Matrix values used for model building under each category.

Low Vulnerability (0-3.9)		Medium Vulnerability (4-6.9)		High Vulnerability (7-10)	
Y(i)	t(i)	Y(i)	t(i)	Y(i)	t(i)
0.002897	1	0.0153	1	0.026618	1
0.024865	2	0.041188	2	0.054112	2
0.042929	3	0.062188	3	0.076645	3
0.057784	4	0.079223	4	0.095114	4
0.069998	5	0.093042	5	0.110251	5
0.080042	6	0.104252	6	0.122657	6
0.088302	7	0.113345	7	0.132825	7
0.095093	8	0.120722	8	0.141159	8
0.126521	77	0.152416	71	0.179021	75
0.126521	78	0.152416	72	0.179021	76
0.126521	79	0.152416	73	0.179021	77
0.126521	80	0.152416	74	0.179021	78
0.126521	81	0.152416	75	0.179021	79
0.126521	82	0.152416	76	0.179021	80
0.126521	83	0.152416	77	0.179021	81

0.126521 84 0.152416 78 0.179021 82

0.126521 85 0.152416 79 0.179021 83

0.126521 86 0.152416 80 0.179021 84

Appendix D

Scientific Research Publishing, Inc.

SCIRP, http://www.scirp.org

JIS COPYRIGHT FORM

This form is intended for original material submitted to the Journal Journal of Information Security (JIS) sponsored by Scientific Research Publishing Inc. (SCIRP). The following agreement must be signed and returned to the JIS Editorial Office before the manuscript can be published. Please read it carefully and keep a copy for your files

Journal: Journal of Information Security (JIS)

Paper ID: 7800396

Paper Title:Stochastic Modelling of Vulnerability Life Cycle and Security Risk Evaluation(7800396)

Author(s): Sasith Rajasooriya

COPYRIGHT CONCESSION AGREEMENT

The undersigned hereby grants SCIRP a nonexclusive copyright that may exist in and to the above Work, and any revised or expanded derivative works submitted to the Journal JIS by the undersigned based on the Work on the understanding that on completion of the layout SCIRP will make the final paper available online without delay, SCIRP guarantees that no university library or individual reader will ever have to buy a subscription or buy access through pay-per-view fees to access the papers published in the Journal JIS.

The undersigned hereby warrants that the Work is original and that he/she is the author of the Work; to the extent the Work incorporates text passages, figures, data or other material from the works of others, the undersigned has obtained any necessary permission including permission from any and all co-authors.

RETAINED RIGHTS, TERMS, AND CONDITIONS

Since the undersigned grants SCIRP a nonexclusive copyright, the undersigned retains the original copyright while SCIRP is granted the same set of rights including the right to sublicense the Work.

SCIRP will publish the Work under a Creative Commons license.By default This is the Creative Commons Attribution 4.0 International License,

CC BY: http://creativecommons.org/licenses/by/4.0/.

Alternatively upon request it is also possible for SCIRP to publish under:Creative Commons Attribution-NonCommercial 4.0 International License,

CC BY- NC:http://creativecommons.org/licenses/by-nc/4.0/.

Authors choice (specify CC BY-NC only if you do not want the default CC BY)

Authors and their employers retain all proprietary rights in any process, procedure, or article of manufacture described in the Work.

Authors who are US Government employees may reproduce or authorize others to reproduce the Work, material extracted verbatim from the Work, or derivative works to the extent permissible under USs law for works authored by US Government employees, and for the authors personal use or for company or organizational use, provided that the source and any SCIRP copyright notice are indicated, the copies are not used

in any way that implies SCIRP endorsement of a product or service of any employer, and the copies themselves are not offered for sale.

In the case of a Work performed under a China Government contract or grant, the SCIRP recognizes that the China Government has royalty-free permission to reproduce all or portions of the Work, and to authorize others to do so, for official China Government purposes only, if the contract/grant so requires.

SCIRP Copyright

It is the formal policy of SCIRP to be granted the nonexclusive copyrights to all copyrightable material in its technical publications and to the individual contributions contained therein, in order to protect the interests of SCIRP, its authors and their employers, and, at the same time, to facilitate the appropriate re-use of this material by others. SCIRP distributes its technical publications throughout the world and does so by various means such as hard copy, microfiche, microfilm, and electronic media. It also abstracts and may translate its publications, and articles contained therein, for inclusion in various compendiums, collective works, databases and similar publications.

No royalties are paid to authors if SCIRP produces revenues by these activities.

Author/Employer Rights

If you are employed and prepared the Work on a subject within the scope of your employment, the copyright in the Work belongs to your employer as a work-for-hire. In that case, SCIRP assumes that when you sign this Form, you are authorized to do so by your employer and that your employer has consented to granting the nonexclusive

copyright, to the representation and warranty of publication rights, and to all other terms and conditions of this Form. If such authorization and consent has not been given to you, an authorized representative of your employer should sign this Form as the Author.

In the event the above work ts not accepted and published by SCIRP or is withdrawn by the author(s) before acceptance by SCIRP, the foregoing granting the nonexclusive copyright shall become null and void and all materials embodying the Work submitted to JIS will be destroyed.

The author signs for and accepts responsibility for releasing this material on behalf of any and all co-authors.

Creative Commons License Type:

O CC BY

● CC BY-NC

Attention: Please choose either one of these two above.

Signature: Sasith Rajasooriya

Date (06/20/2016)

Appendix D (Continued)

Scientific Research Publishing, Inc.

SCIRP, http://www.scirp.org

JIS COPYRIGHT FORM

This form is intended for original material submitted to the Journal Journal of Information Security (JIS) sponsored by Scientific Research Publishing Inc. (SCIRP). The following agreement must be signed and returned to the JIS Editorial Office before the manuscript can be published. Please read it carefully and keep a copy for your files

Journal: Journal of Information Security (JIS)

Paper ID: 7800440

Paper Title:CYBER SECURITY: NONLINEAR STOCHASTIC MODELS FOR PREDICTING THE EXPLOITABILITY(7800440)

Author(s): Sasith Rajasooriya

COPYRIGHT CONCESSION AGREEMENT

The undersigned hereby grants SCIRP a nonexclusive copyright that may exist in and to the above Work, and any revised or expanded derivative works submitted to the Journal JIS by the undersigned based on the Work on the understanding that on completion of the layout SCIRP will make the final paper available online without delay, SCIRP guarantees that no university library or individual reader will ever have to buy a subscription or buy access through pay-per-view fees to access the papers published in the Journal JIS.

The undersigned hereby warrants that the Work is original and that he/she is the author of the Work; to the extent the Work incorporates text passages, figures, data or other material from the works of others, the undersigned has obtained any necessary permission including permission from any and all co-authors.

RETAINED RIGHTS, TERMS, AND CONDITIONS

Since the undersigned grants SCIRP a nonexclusive copyright, the undersigned retains the original copyright while SCIRP is granted the same set of rights including the right to sublicense the Work.

SCIRP will publish the Work under a Creative Commons license.By default This is the Creative Commons Attribution 4.0 International License,

CC BY: http://creativecommons.org/licenses/by/4.0/.

Alternatively upon request it is also possible for SCIRP to publish under:Creative Commons Attribution-NonCommercial 4.0 International License,

CC BY- NC:http://creativecommons.org/licenses/by-nc/4.0/.

Authors choice (specify CC BY-NC only if you do not want the default CC BY)

Authors and their employers retain all proprietary rights in any process, procedure, or article of manufacture described in the Work.

Authors who are US Government employees may reproduce or authorize others to reproduce the Work, material extracted verbatim from the Work, or derivative works to the extent permissible under USs law for works authored by US Government employees, and for the authors personal use or for company or organizational use, provided that the source and any SCIRP copyright notice are indicated, the copies are not used

in any way that implies SCIRP endorsement of a product or service of any employer, and the copies themselves are not offered for sale.

In the case of a Work performed under a China Government contract or grant, the SCIRP recognizes that the China Government has royalty-free permission to reproduce all or portions of the Work, and to authorize others to do so, for official China Government purposes only, if the contract/grant so requires.

SCIRP Copyright

It is the formal policy of SCIRP to be granted the nonexclusive copyrights to all copyrightable material in its technical publications and to the individual contributions contained therein, in order to protect the interests of SCIRP, its authors and their employers, and, at the same time, to facilitate the appropriate re-use of this material by others. SCIRP distributes its technical publications throughout the world and does so by various means such as hard copy, microfiche, microfilm, and electronic media. It also abstracts and may translate its publications, and articles contained therein, for inclusion in various compendiums, collective works, databases and similar publications.

No royalties are paid to authors if SCIRP produces revenues by these activities.

Author/Employer Rights

If you are employed and prepared the Work on a subject within the scope of your employment, the copyright in the Work belongs to your employer as a work-for-hire. In that case, SCIRP assumes that when you sign this Form, you are authorized to do so by your employer and that your employer has consented to granting the nonexclusive

copyright, to the representation and warranty of publication rights, and to all other terms and conditions of this Form. If such authorization and consent has not been given to you, an authorized representative of your employer should sign this Form as the Author.

In the event the above work ts not accepted and published by SCIRP or is withdrawn by the author(s) before acceptance by SCIRP, the foregoing granting the nonexclusive copyright shall become null and void and all materials embodying the Work submitted to JIS will be destroyed.

The author signs for and accepts responsibility for releasing this material on behalf of any and all co-authors.

Creative Commons License Type:

O CC BY

● CC BY-NC

Attention: Please choose either one of these two above.

Signature: Sasith Rajasooriya

Date (03/22/2017)