A NOVEL APPROACH TO QUANTIFY CYBERSECURITY
FOR ELECTRIC POWER SYSTEMS

by
Paul R. Kaster, Jr.
Copyright by Paul R. Kaster, Jr.
All Rights Reserved
A thesis submitted to the Faculty and the Board of Trustees of the Colorado School of Mines in partial fulfillment of the requirements for the degree of Doctor of Philosophy (Electrical Engineering).

Golden, Colorado

Date: ________________

Signed: ___________________

Paul R. Kaster, Jr.

Signed: ___________________

Dr. Pankaj Sen
Thesis Advisor

Golden, Colorado
Date: ________________

Signed: ___________________

Dr. Atef Elsherbeni
Professor and Head
Department of Electrical Engineering and Computer Science
Electric Power grid cybersecurity is a topic gaining increased attention in academia, industry, and government circles, yet a method of quantifying and evaluating a system’s security is not yet commonly accepted. In order to be useful, a quantification scheme must be able to accurately reflect the degree to which a system is secure, simply determine the level of security in a system using real-world values, model a wide variety of attacker capabilities, be useful for planning and evaluation, allow a system owner to publish information without compromising the security of the system, and compare relative levels of security between systems. Published attempts at quantifying cybersecurity fail at one or more of these criteria. This document proposes a new method of quantifying cybersecurity that meets those objectives. This dissertation evaluates the current state of cybersecurity research, discusses the criteria mentioned previously, proposes a new quantification scheme, presents an innovative method of modeling cyber attacks, demonstrates that the proposed quantification methodology meets the evaluation criteria, and proposes a line of research for future efforts.
## TABLE OF CONTENTS

ABSTRACT .......................................................................................................................... iii

LIST OF FIGURES ............................................................................................................... vii

LIST OF TABLES .................................................................................................................. ix

ACKNOWLEDGMENTS ...................................................................................................... xii

CHAPTER 1 INTRODUCTION .................................................................................. 1

1.1 Introduction .................................................................................................................. 1

1.2 The Smart Grid ............................................................................................................. 1

1.3 Cybersecurity ................................................................................................................. 4

1.4 Cybersecurity Fundamentals ....................................................................................... 5

1.4.1 Confidentiality, Integrity, and Availability ................................................................. 6

1.4.2 Cybersecurity Core Functions .................................................................................... 9

1.4.3 Rural Electric Case Study ......................................................................................... 17

1.5 Cybersecurity Quantification ....................................................................................... 18

CHAPTER 2 LITERATURE REVIEW .............................................................................. 20

2.1 Probabilistic / Stochastic Methods ............................................................................. 20

2.1.1 Mean Time To Security Failure ............................................................................... 20

2.1.2 Probabilistic Risk Assessment ................................................................................. 23

2.1.3 Vulnerability Graphs .............................................................................................. 24

2.1.4 Mean Time to Compromise .................................................................................... 26

2.1.5 Time To Compromise ............................................................................................. 28

2.2 Measures of Effectiveness ........................................................................................... 29

2.3 Ideal Based Metrics .................................................................................................... 31

2.3.1 Common Vulnerability Scoring System .................................................................. 31

2.3.2 VEA-ility ............................................................................................................... 35
<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1</td>
<td>Smart grid interconnected networks</td>
<td>4</td>
</tr>
<tr>
<td>1.2</td>
<td>CIA Balancing Act</td>
<td>7</td>
</tr>
<tr>
<td>1.3</td>
<td>Simple Power System</td>
<td>10</td>
</tr>
<tr>
<td>1.4</td>
<td>Core Function Process</td>
<td>11</td>
</tr>
<tr>
<td>2.1</td>
<td>State Transition Diagram for Intrusion Tolerant System</td>
<td>21</td>
</tr>
<tr>
<td>2.2</td>
<td>Discreet Time Markov Process</td>
<td>22</td>
</tr>
<tr>
<td>2.3</td>
<td>Process to Conduct a Level 1 PRA</td>
<td>24</td>
</tr>
<tr>
<td>2.4</td>
<td>Sample Vulnerability Graph</td>
<td>25</td>
</tr>
<tr>
<td>2.5</td>
<td>Sample Event Tree</td>
<td>25</td>
</tr>
<tr>
<td>2.6</td>
<td>Sample Safe Rating</td>
<td>26</td>
</tr>
<tr>
<td>2.7</td>
<td>State-space Model for MTTC</td>
<td>27</td>
</tr>
<tr>
<td>2.8</td>
<td>TTC: Attempts vs. Vulnerability Pool</td>
<td>28</td>
</tr>
<tr>
<td>2.9</td>
<td>TTC Probability of usable vulnerability</td>
<td>28</td>
</tr>
<tr>
<td>2.10</td>
<td>TTC for Various Vulnerability Pools</td>
<td>29</td>
</tr>
<tr>
<td>2.11</td>
<td>CDC Buildup</td>
<td>30</td>
</tr>
<tr>
<td>2.12</td>
<td>Probability of Successful Defense</td>
<td>30</td>
</tr>
<tr>
<td>2.13</td>
<td>CVSS Attack Vector Flow Chart</td>
<td>31</td>
</tr>
<tr>
<td>2.14</td>
<td>Simple Attack Graph</td>
<td>38</td>
</tr>
<tr>
<td>2.15</td>
<td>Petri Net Example</td>
<td>40</td>
</tr>
<tr>
<td>2.16</td>
<td>Petri Net for Attack on Smart Meter</td>
<td>40</td>
</tr>
<tr>
<td>2.17</td>
<td>Attack Method Development</td>
<td>42</td>
</tr>
<tr>
<td>2.18</td>
<td>Impact of Countermeasures</td>
<td>43</td>
</tr>
<tr>
<td>2.19</td>
<td>Attack tree</td>
<td>45</td>
</tr>
<tr>
<td>2.20</td>
<td>Leaf Vulnerability Index</td>
<td>45</td>
</tr>
<tr>
<td>2.21</td>
<td>Combination Approach Analytical Model</td>
<td>46</td>
</tr>
<tr>
<td>2.22</td>
<td>Combination Approach Network</td>
<td>46</td>
</tr>
<tr>
<td>2.23</td>
<td>Combination Approach Attack Graph</td>
<td>47</td>
</tr>
<tr>
<td>2.24</td>
<td>Attack Graph and TTC for Substations</td>
<td>47</td>
</tr>
<tr>
<td>3.1</td>
<td>Set of All Potential Vulnerabilities</td>
<td>51</td>
</tr>
<tr>
<td>3.2</td>
<td>Set of All Known Vulnerabilities</td>
<td>51</td>
</tr>
<tr>
<td>Figure 3.3</td>
<td>Set of All Vulnerabilities Present in a Target System</td>
<td>52</td>
</tr>
<tr>
<td>Figure 3.4</td>
<td>Set of Target Vulnerabilities with an Attack Vector</td>
<td>52</td>
</tr>
<tr>
<td>Figure 3.5</td>
<td>Set of Vulnerabilities an Adversary can Exploit</td>
<td>53</td>
</tr>
<tr>
<td>Figure 3.6</td>
<td>Set of Vulnerabilities Present, Exploitable, With an Attack Vector</td>
<td>53</td>
</tr>
<tr>
<td>Figure 3.7</td>
<td>Simple Scenario</td>
<td>54</td>
</tr>
<tr>
<td>Figure 3.8</td>
<td>Simple scenario (continued)</td>
<td>55</td>
</tr>
<tr>
<td>Figure 3.9</td>
<td>System with Multiple Attack Vectors</td>
<td>56</td>
</tr>
<tr>
<td>Figure 3.10</td>
<td>Example network</td>
<td>63</td>
</tr>
<tr>
<td>Figure 4.1</td>
<td>Step 1 Attack Scenario</td>
<td>67</td>
</tr>
<tr>
<td>Figure 4.2</td>
<td>Test 1 Results</td>
<td>68</td>
</tr>
<tr>
<td>Figure 4.3:</td>
<td>Step 2 Attack Scenario</td>
<td>68</td>
</tr>
<tr>
<td>Figure 4.4</td>
<td>Test 2 Cyber Power vs. Success Rate</td>
<td>70</td>
</tr>
<tr>
<td>Figure 4.5</td>
<td>Test 2 Cyber Power vs. Success Rate, Detailed View</td>
<td>71</td>
</tr>
<tr>
<td>Figure 4.6</td>
<td>Test 2 Cyber Power vs. Attempts</td>
<td>72</td>
</tr>
<tr>
<td>Figure 4.7</td>
<td>Step 3 Attack Scenario</td>
<td>73</td>
</tr>
<tr>
<td>Figure 4.8</td>
<td>Test 3 Cyber Power vs. Success Rate</td>
<td>74</td>
</tr>
<tr>
<td>Figure 4.9</td>
<td>Test 3 Cyber Power vs. Attempts</td>
<td>75</td>
</tr>
<tr>
<td>Figure 4.10</td>
<td>Test 4 Cyber Power vs. Success Rate</td>
<td>77</td>
</tr>
<tr>
<td>Figure 4.11</td>
<td>Test 4 Cyber Power vs. Time</td>
<td>78</td>
</tr>
<tr>
<td>Figure 4.12</td>
<td>Test 5 Cyber Power</td>
<td>83</td>
</tr>
<tr>
<td>Figure 4.13</td>
<td>Test 5 Success Rate</td>
<td>83</td>
</tr>
<tr>
<td>Figure 4.14</td>
<td>Test 5 Difficulty</td>
<td>84</td>
</tr>
<tr>
<td>Figure 4.15</td>
<td>High-level Representation of Proposed Test Bed</td>
<td>86</td>
</tr>
<tr>
<td>Figure 4.16</td>
<td>Portability Cyber Power vs. Success Rate</td>
<td>90</td>
</tr>
<tr>
<td>Figure 4.17</td>
<td>Portability Cyber Power vs. Time</td>
<td>91</td>
</tr>
<tr>
<td>Figure 5.1</td>
<td>Sample safe rating</td>
<td>95</td>
</tr>
</tbody>
</table>
## LIST OF TABLES

<table>
<thead>
<tr>
<th>Table 1.1</th>
<th>CIA Definitions ...................................................................................... 6</th>
</tr>
</thead>
<tbody>
<tr>
<td>Table 1.2</td>
<td>Potential Impact Levels ........................................................................ 8</td>
</tr>
<tr>
<td>Table 1.3</td>
<td>Core Functions ................................................................................... 11</td>
</tr>
<tr>
<td>Table 1.4</td>
<td>Risk .................................................................................................... 12</td>
</tr>
<tr>
<td>Table 1.5</td>
<td>Risk Assessment Example ..................................................................... 12</td>
</tr>
<tr>
<td>Table 1.6</td>
<td>Adversaries ........................................................................................ 16</td>
</tr>
<tr>
<td>Table 2.1</td>
<td>PRA Steps .......................................................................................... 23</td>
</tr>
<tr>
<td>Table 2.2</td>
<td>MTTC Example State Dwell Times ....................................................... 27</td>
</tr>
<tr>
<td>Table 2.3</td>
<td>CVSS Metrics ..................................................................................... 32</td>
</tr>
<tr>
<td>Table 2.4</td>
<td>CVSS Example Metrics ........................................................................ 33</td>
</tr>
<tr>
<td>Table 2.5</td>
<td>CVSS Scores ....................................................................................... 34</td>
</tr>
<tr>
<td>Table 2.6</td>
<td>RoS Classes ....................................................................................... 36</td>
</tr>
<tr>
<td>Table 2.7</td>
<td>Security Countermeasures .................................................................... 41</td>
</tr>
<tr>
<td>Table 2.8</td>
<td>Correlation Between Weakest Link, CVSS, and TTC ............................... 48</td>
</tr>
<tr>
<td>Table 2.9</td>
<td>Correlation Between TTC and Number of Vulnerabilities ...................... 49</td>
</tr>
<tr>
<td>Table 3.1</td>
<td>Summary of Vulnerability Sets ............................................................ 53</td>
</tr>
<tr>
<td>Table 3.2</td>
<td>Security Model Concepts ..................................................................... 57</td>
</tr>
<tr>
<td>Table 3.3</td>
<td>Example Sets ...................................................................................... 59</td>
</tr>
<tr>
<td>Table 3.4</td>
<td>Vulnerability Characteristics .............................................................. 59</td>
</tr>
<tr>
<td>Table 3.5</td>
<td>Measures of Security .......................................................................... 61</td>
</tr>
<tr>
<td>Table 3.6</td>
<td>Example vulnerabilities ....................................................................... 63</td>
</tr>
<tr>
<td>Table 4.1</td>
<td>Evaluation criteria ............................................................................. 65</td>
</tr>
<tr>
<td>Table 4.2</td>
<td>Step 2 Possible Outcomes ................................................................... 69</td>
</tr>
<tr>
<td>Table 4.3</td>
<td>Vulnerability Characteristics .............................................................. 76</td>
</tr>
<tr>
<td>Table 4.4</td>
<td>Test Vulnerabilities ............................................................................ 79</td>
</tr>
<tr>
<td>Table 4.5</td>
<td>Test Scenarios ................................................................................... 81</td>
</tr>
<tr>
<td>Table 4.6</td>
<td>Accuracy Results ............................................................................... 82</td>
</tr>
<tr>
<td>Table 4.7</td>
<td>Correlations ....................................................................................... 84</td>
</tr>
<tr>
<td>Table 4.8</td>
<td>Security Demonstration ...................................................................... 88</td>
</tr>
<tr>
<td>Table 5.1</td>
<td>Evaluation criteria ............................................................................. 93</td>
</tr>
<tr>
<td>Table 5.1</td>
<td>Vulnerability Ratings .................................................................................. 96</td>
</tr>
<tr>
<td>----------</td>
<td>---------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Table A.1</td>
<td>Selected Power Grid Cybersecurity Standards ............................................ 108</td>
</tr>
<tr>
<td>Table A.2</td>
<td>Selected Power Grid Cybersecurity Standards ............................................ 109</td>
</tr>
<tr>
<td>Table A.3</td>
<td>Selected Power Grid Cybersecurity Standards ............................................ 110</td>
</tr>
<tr>
<td>Table A.4</td>
<td>Selected Power Grid Cybersecurity Standards ............................................ 111</td>
</tr>
</tbody>
</table>
ACKNOWLEDGMENTS

This dissertation would not have been possible without the support of my advisor (Dr. Pankaj Sen), the rest of my committee (Dr. Ravel Ammerman, Dr. Dinesh Mehta, Dr. Benjamin Kroposki, Dr. Chester Van Tyne), the United States Air Force Academy (Col Jeffrey Butler, Col Anne Clark, Dr. Andy Laffely), and the Air Force Institute of Technology (Capt John Scheuren, Mr. Luke Whitney). I would also like to thank Jahi Simbai, Lori Sisneros, and Dr. Dick De Blasio for their assistance. Most of all, I would like to thank Dr. Stephanie Miller for her support throughout the process.

The views expressed in this article are those of the author and do not necessarily reflect the official policy or position of the Air Force, the Department of Defense or the U.S. Government.
CHAPTER 1 INTRODUCTION

Cybersecurity is an area of ever increasing concern for all utility providers across North America. Operators are becoming more reliant on the “smart grid” for improved operations and services. This increased reliance makes it more vulnerable while, at the same time, it amplifies the potential consequences of a successful cyber-attack.

1.1 Introduction [1]

This dissertation begins in Chapter 1 by discussing the Smart Grid and providing a brief overview of the importance of cybersecurity. It proceeds to discuss cybersecurity fundamentals, using the example of a rural electric utility to provide context. Once this background is presented, the core question of quantification is introduced. Chapter 2 proceeds to do a thorough literature review on the topic of cybersecurity as it relates to electrical power, and an even more thorough review on the published research surrounding quantification. Chapter 3 presents a new approach to quantification, including a new model and methodology. Chapter 4 describes the evaluation criteria used to validate the model. A new methodology for simulating cyber attacks is presented, and the results of these simulations are compared to the evaluation criteria. Chapter 5 concludes this document by discussing the impact of the research and future research to be conducted.

1.2 The Smart Grid

The North American electric power grid (grid) is becoming ever more reliant on the backbone of communications systems commonly called the Smart Grid. Title XIII of the Energy Independence and Security Act of 2007 (EISA) specifies the characteristics of Smart Grid. They can be summarized as:

- Increased use of digital information and controls technology.
• Deployment and integration of distributed resources and generation, including renewable resources.
• Development and incorporation of demand response, demand-side resources, and energy-efficiency resources.
• Deployment of `smart' technologies for metering, communications concerning grid operations and status, and distribution automation.
• Integration of `smart' appliances and consumer devices.
• Deployment and integration of advanced electricity storage and peak-shaving technologies.
• Provision to consumers of timely information and control options.
• Development of standards for communication and interoperability of appliances and equipment connected to the electric grid, including the infrastructure serving the grid.
• Identification and lowering of unreasonable or unnecessary barriers to adoption of smart grid technologies, practices, and services. [2]

The grid faces a variety of evolving and expanding threats, requiring a proactive response to develop robust, resilient networks. In that vein, a great deal of guidance exists on how to provide proper cybersecurity for the Smart Grid. Various agencies including the North American Electric Reliability Corporation (NERC), the International Organization for Standardization (ISO), the International Electrotechnical Commission (IEC), the International Society of Automation (ISA), the National Institute of Standards and Technology (NIST), and the Institute of Electrical and Electronics Engineers (IEEE) (to name a few) have published guidance, frameworks, and standards attempting to address the issue.

Much of the efforts to investigate the cyber threat and cyber defenses have been hampered by an absence of standardized terminology, common understanding in defining the scope, and coordinated research efforts. Cybersecurity itself has not been quantified in a standard way so as to allow measurement, testing, and comparison. To that effect, increased collaboration and interconnection between academia, national labs, and (most critically) the utility industry is required.
Inherently, the Power Grid uses equipment and architectures that distinguish it from more traditional communications networks. This has a number of impacts.

First, the Power Grid has unique vulnerabilities: Almost by definition, the unique equipment in the grid provides unique opportunities for an attacker. An example would be protective relays, devices that are not routinely used outside of the power industry. As with any other device containing software, electronics, and network connection, protective relays are vulnerable to unauthorized access and control. The unique devices and software used in protective relaying warrant specific testing for cybersecurity vulnerabilities.

While the Power Grid has unique vulnerabilities, it also has tools that augment its defensibility: The Power Grid has unique monitoring capabilities and physical characteristics that can enable improved detection and response. For example, a poorly designed false data injection attack on a bus could be easily identified if it does not solve the power flow equations [3]. Further research into the capabilities and limitations of SCADA devices, synchrophasors, state estimation techniques, and other Power Grid-specific traits is required.

The Power Grid also has unique communications requirements: This can include very low data rate two-wire phone lines servicing a remote SCADA device, or a critical control link requiring sub-cycle latency for safety and operational regions, among others. In the first case, cybersecurity may not be required due to the lack of utility it provides an attacker [4]. In the latter case, the low latency requirements may eliminate the ability to use standard defensive schemes such as encryption, authentication, passwords, and firewalls, necessitating a more nuanced design.

The Smart Grid is a network of networks: These networks include Transmission, Bulk Generation, Operations, Distribution, Marketing, the Service Provider, and the Customers [5]. NIST Interagency and Internal Report (NISTIR) 7628 provides a good visual description of the network interconnectivity, shown in Fig. 1 Each network is a unique security case. For example, if the Customer network is compromised, there is potential for great economic damage but the grid will likely still function. If the Transmission network is compromised, there is potential for physical damage to the grid
similar to the 2003 blackout. Care must be taken in how these networks interact. Clearly, there is no need for the Bulk Generation and Service Provider networks to interact, and providing that connection would add a significant vulnerability to the networks. Less obviously, regulatory requirements may prohibit the Marketing network from getting certain data from the Operations network. The sheer complexity of the system warrants further research.

Figure 1.1 Smart grid interconnected networks [5]

1.3 Cybersecurity

The potential impact of an effective cyber-attack on the grid is extreme. Economically, outages and disturbances from all events cost at least $119 billion per year, and a single event (such as the 2003 blackout) can cost up to $6 billion [6] [7]. Strategically, a United States Department of Defense (DoD) report in 2008 found that 99% of DoD energy originates “outside the fence” and that insufficient backup capacity exists to ensure continuity in the event of an outage lasting several months [8]. Additionally, the Government Accountability Office (GAO) found in 2009 that 31 of the DoD’s 34 most critical assets rely on commercial power as their primary source of electricity [9].

In 2011, the commissioners of the Federal Energy Regulatory Commission (FERC) stated that “the threat of a cyber-attack on the grid was the top threat to electricity
reliability in the United States [10]. In 2012, Robert Mueller, the director of the Federal Bureau of Investigation (FBI) projected that the cyber threat will surpass physical terrorist attacks as the “number one threat to the country” [11]. That same year, Defense Secretary Leon Panetta described a cyber-attack on the power grid as a “cyber-Pearl Harbor” [12].

A 2013 report found that “the electric grid is the target of numerous and daily cyber-attacks.” Several utilities in the report indicated that they were under “daily,” “constant,” or “frequent” attacks including probing attacks, phishing, and malware infection. A single utility reported up to 10,000 attack attempts each month [11].

The volume and potential impact of the cyber threat is vast, and it is in the national interest to ensure that the grid is secured. The threat exists at more than just the national level; attacks can occur against specific companies that can drastically affect the corporate operations. On 22 November 2014, Sony Pictures suffered a cyber tack that “initially caused crippling computer problems for workers at Sony, who were forced to work with pen and paper.” [13] The attackers released confidential data that proved embarrassing to Sony, stole and released intellectual property on the internet, and caused a major operational disruption when Sony chose to not release a film in theaters as a result of the attack [13]. While not the same industry as rural power providers, this does indicate the potential impact that a cyber attack can have on a business.

Aside from the threat posed by an attack, small utilities must also ensure that they comply with the Critical Infrastructure Protection (CIP) standards set by the North American Electric Reliability Corporation (NERC), as well as any applicable state and local standards. Failure to comply with these standards can lead to fines or other punitive action, seriously affecting operations and the operator’s bottom line.

1.4 Cybersecurity Fundamentals

A thorough analysis of cybersecurity would be incomplete without presenting certain key concepts. This is particularly important to avoid confusion, as certain words (such as “vector”) have different meanings based on the context. This section will present these fundamental concepts.
1.4.1 Confidentiality, Integrity, and Availability

Cybersecurity can be summed up as a continuous process of assessing the network and implementing physical, technical, or policy controls that balance Confidentiality, Integrity, and Availability (CIA). Table 1.1 provides detailed definitions for each of these terms.

Table 1.1 CIA Definitions [14]

<table>
<thead>
<tr>
<th>Term</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Confidentiality</td>
<td>“Preserving authorized restrictions on information access and disclosure, including means for protecting personal privacy and proprietary information”</td>
</tr>
<tr>
<td>Integrity</td>
<td>“Guarding against improper information modification or destruction, and includes ensuring information non-repudiation and authenticity”</td>
</tr>
<tr>
<td>Availability</td>
<td>“Ensuring timely and reliable access to and use of information”</td>
</tr>
</tbody>
</table>

Satisfying these three criteria is a constant balancing act because of their interdependency, as an action to improve one trait may have a detrimental impact on the other two. A simple example would be disconnecting a system from any other system, or “unplugging” it, and locking it in a room under guard. This would maximize Confidentiality and Integrity, since nothing could possibly touch the system without alerting the guard. It would drastically reduce availability and local access would be potentially inconvenient.

A fourth factor that must always be considered is “Cost”. Some systems legitimately require high levels of Confidentiality, Integrity, and Availability. An example would be some varieties of Advanced Metering Infrastructure (AMI). Users can reasonably expect that their energy use and related billing are not publicly available (high Confidentiality). Utilities and consumers must be certain that metering and billing information is accurate (high Integrity). Some users, such as large industrial plants, may require billing information that is accurate up to the minute in order to minimize their energy costs (high Availability). In this case, the utility will need to implement an
expensive system incorporating encrypted high speed data lines, secure data centers, and robust data backups to ensure that the high level of service is provided. Figure 1.2 shows this balance.

Most other applications, however, allow for one or more of those criteria to be degraded in favor of the other(s). The interfaces between control systems (e.g. relays) are a good example. The information that those interfaces convey must be accurate in order for the operator to make the correct decision (high Integrity), and operators may need to respond to changes within minutes or seconds (high Availability). The operator may not be concerned with anyone else seeing that same data (low Confidentiality), as long as standards for Integrity and Availability are met. In this case, the utility may not need to use encrypted, secure, communication lines as long as the lines and sensors are reliable.

For each system, the utility must examine its unique CIA requirements. The above examples used a potentially over-simplified assessment process in order to demonstrate the concept. A more formal method would be to examine the level of impact that a failure of that attribute will have on the overall system, the utility, and the customer. A standard way of viewing these levels is to categorize them as Low, Moderate, or High. Table 1.2 is a simple tool to help make this assessment.
Table 1.2 Potential Impact Levels [5]

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Failure</th>
<th>Impact Level</th>
</tr>
</thead>
<tbody>
<tr>
<td>Confidentiality</td>
<td>Unauthorized disclosure</td>
<td>Low: Limited impact</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Moderate: Serious impact</td>
</tr>
<tr>
<td></td>
<td></td>
<td>High: Severe or catastrophic impact</td>
</tr>
<tr>
<td>Integrity</td>
<td>Unauthorized modification or destruction</td>
<td>Low: Limited impact</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Moderate: Serious impact</td>
</tr>
<tr>
<td></td>
<td></td>
<td>High: Severe or catastrophic impact</td>
</tr>
<tr>
<td>Availability</td>
<td>Disruption of Access</td>
<td>Low: Limited impact</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Moderate: Serious impact</td>
</tr>
<tr>
<td></td>
<td></td>
<td>High: Severe or catastrophic impact</td>
</tr>
</tbody>
</table>

Another example may be illustrative at this point. Most utilities have networks connecting their back offices that perform functions ranging from e-mail to timesheets to developing strategic business practices. Unauthorized disclosure of this information could give competitors and unfair business advantage or could violate laws governing handling personal information. At its most extreme, this could cause the utility to go bankrupt as competitors use this information to their own ends. The potential impact is “severe or catastrophic,” so one would assess the Confidentiality impact level as “High.” Unauthorized modification or destruction of that data would most certainly be more than an inconvenience, and could potentially result in major problems such as accounting difficulties. It is somewhat less likely to cause the utility to collapse, however, so the Integrity impact level would be “Moderate” (though there could be an argument for “High”). Rarely, however, is this type of information needed within seconds. A disruption in access would be an inconvenience, and could have an operational impact, but that impact would likely be limited. For that reason, one would categorize the Availability impact level as “Low.”

These individual cases are for discussions only. It is clear that there is no single rule that fits all individual needs.
1.4.2 Cybersecurity Core Functions

The CIA model presented earlier serves as a model for why to guide a cybersecurity strategy. The Core Functions, in turn, serve as a model for how one addresses the topic.

National Institute of Standards and Technology (NIST) identifies five Core Functions that should be considered when building a cybersecurity strategy: Identify, Protect, Detect, Respond, and Recover. Each is defined in Table 1.3.

The purpose of cybersecurity is to continuously ensure the required level of Confidentiality, Integrity, and Availability. This is done by “Identifying” risk, assets, data, and capabilities, “Protecting” the infrastructure from attack, “Detecting” and “Responding” to any attack, and “Recovering” from the incident. Each of these concepts is defined in Table 1.3.

These functions will be continuous and simultaneous; however, when initially facing the task, Identify serves as the preferred starting point. The utility provider will perform a thorough risk analysis and examine its existing systems. Of the core functions, this is the most critical, as it forms the base upon which the others reside. An impeccable protection scheme with the best sensors and the best personnel able to respond to and recover from a threat can be rendered moot if the vulnerability attacked is unknown, or if the capabilities of the attacker are underestimated. This continuous process is shown in Figure 1.4.

In order to understand these functions, one must understand the Risk Assessments, Adversaries, and Controls. The Chinese military philosopher Sun Tzu said, “If you know the enemy and know yourself, you need not fear the result of a hundred battles.” [15] Thus, as part of the Identify function, the utility operator must understand potential adversaries as well as they understand their own systems. The first step in this process is to conduct a thorough risk assessment as described in Table 1.3.

Risk (R) is a function of Threats (T), Vulnerabilities (V), and Consequences (C) [16]. Often, the relationship is presented as in (1.1).
\[ R = T \times V \times C \] (1.1)

T and V are probabilities. T is the probability that a Threat exists, is capable of conducting an effective attack, and is actively conducting said attack. V is the probability that a Vulnerability exists that can be exploited by a particular Threat. R and C are in units of value, such as dollars or hours. This simple equation would be applied multiple times for each unique Threat, Vulnerability, or Consequence.

Consider a very simple example. Figure 1.3 shows a utility that has two vulnerable breakers that could be attacked. There are two threats to those breakers: external hackers and disgruntled employees. Each breaker has consequences in terms of down time and financial cost in terms of lost income when the downstream customers are not paying for service.

![Figure 1.3 Simple Power System](image)

Delving into statistical values, crime reports, internal assessments, and a few guesses, the utility determines probabilities and values for the variables. It determines that the Threat probability of an external hacker is 80%, and the Threat probability of a disgruntled employee is 20%. The Vulnerability probability for breaker A to a hacker is 5%, and to a disgruntled employee is 40%. For breaker B, those values are 3% and 30%, respectively. The Consequences for Relay A are $1,000 and 1 hour, and for breaker B are $500 and 2 hours. The utility then develops Table 1.5, where each cell shows the quantified Risk.

In this example, the greatest financial risk to the utility is a disgruntled employee attacking breaker A, but the greatest risk in terms of outage duration would be a
disgruntled employee attacking breaker B. This knowledge will allow the utility to develop a rationale for cybersecurity strategy that addresses the greatest risks.

Table 1.3 Core Functions [17]

<table>
<thead>
<tr>
<th>Term</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Identify</td>
<td>“Develop the organizational understanding to manage cybersecurity risk to systems, assets, data, and capabilities.”</td>
</tr>
<tr>
<td>Protect</td>
<td>“Develop and implement the appropriate safeguards to ensure delivery of critical infrastructure services.”</td>
</tr>
<tr>
<td>Detect</td>
<td>“Develop and implement the appropriate activities to identify the occurrence of a cybersecurity event.”</td>
</tr>
<tr>
<td>Respond</td>
<td>“Develop and implement the appropriate activities to take action regarding a detected cybersecurity event.”</td>
</tr>
<tr>
<td>Recover</td>
<td>“Develop and implement the appropriate activities to maintain plans for resilience and to restore any capabilities or services that were impaired due to a cybersecurity event.”</td>
</tr>
</tbody>
</table>

Figure 1.4 Core Function Process
Table 1.4 Risk [18]

<table>
<thead>
<tr>
<th>Term</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Risk (R)</td>
<td>“potential for an unwanted outcome resulting from an incident, event, or occurrence, as determined by its likelihood and the associated consequences”</td>
</tr>
<tr>
<td>Threat (T)</td>
<td>“natural or man-made occurrence, individual, entity, or action that has or indicates the potential to harm life, information, operations, the environment and/or property”</td>
</tr>
<tr>
<td>Vulnerability (V)</td>
<td>“physical feature or operational attribute that renders an entity open to exploitation or susceptible to a given hazard”</td>
</tr>
<tr>
<td>Consequence (C)</td>
<td>“effect of an event, incident, or occurrence”</td>
</tr>
</tbody>
</table>

Often, however, that level of data is unknown when conducting the assessment. A much simpler solution is to simply plan for the worst case scenario and the most likely scenario. The United States Marine Corps calls these the enemy’s Most Likely Course of Action (MLCOA) and the enemy’s Most Dangerous Course of Action (MDCOA). This is a much more subjective methodology, but has the advantage of being fast and reasonably accurate.

Table 1.5 Risk Assessment Example

<table>
<thead>
<tr>
<th>Breaker A</th>
<th>Breaker B</th>
</tr>
</thead>
<tbody>
<tr>
<td>5% Vulnerable to Hacker</td>
<td>3% Vulnerable to Hacker</td>
</tr>
<tr>
<td>40% Vulnerable to Employee</td>
<td>30% Vulnerable to Employee</td>
</tr>
<tr>
<td><strong>Threat</strong></td>
<td></td>
</tr>
<tr>
<td>Hacker</td>
<td>$40.00</td>
</tr>
<tr>
<td>Employee</td>
<td>$80.00</td>
</tr>
<tr>
<td><strong>Cost</strong></td>
<td><strong>Time</strong></td>
</tr>
<tr>
<td>$1,000</td>
<td>1 hr</td>
</tr>
<tr>
<td>80%</td>
<td>0.040</td>
</tr>
<tr>
<td>20%</td>
<td>0.080</td>
</tr>
</tbody>
</table>
In the prior example, the utility may believe that the most likely threat will come from an outside hacker attempting to control breaker A in order to cause chaos. This would be the MLCOA. The utility may also know that load A is a large population, but breaker B is at a remote substation and difficult to repair quickly. Additionally, the utility knows that many of its employees have physical and electronic access to the substation equipment, and so are much more likely to be successful if they attack. If the utility cares more about having minimal downtime than its bottom line, the MDCOA would be a disgruntled employee attacking breaker B.

Since each Core Function is considered continuously, it is likely that a utility may begin with a simple MLCOA/MDCOA assessment and, as more information is gathered, refine the assessment by quantifiably assessing risk. This, in fact, is precisely what the Department of Homeland Security (DHS) did when assessing the risk of a terrorist attack [16]. From 2001 to 2003, Risk was simply equal to population. Looked at simply, terrorists are more likely to attack a large population center, and the potential consequences for a large population center were greatest. Therefore, both the MLCOA and MDCOA were an attack on a large population center. In time, DHS refined their definition until they could more accurately quantify the total Risk.

A key component of the Identify function is to recognize one’s adversaries. Table 1.6 shows a list of potential candidates. These include external and insider threats, individuals and organizations, those motivated by politics, finances, or personal reasons, and unintentional threats. In short, the number and variety of threats are staggering. Despite this, it is extremely helpful to understand what threats may be most pertinent to the particular target, in order to develop the proper cybersecurity strategy.

Consider the example of a disgruntled employee. There are several unique concerns with this type of adversary. The first is the status as an insider threat. The disgruntled employee will be unidentified and unidentifiable. He or she will have access to internal networks and physical access to facilities. This adversary can cause immense harm based on knowledge of the utility’s inner workings.

There are several actions a utility may take just to mitigate the risk posed by a disgruntled employee. These include having a policy to delete accounts and passwords
when an employee is terminated, technical limitations requiring two employees to authenticate certain commands, and a badging system that keeps employees to physical areas where they are required to work and no others. These controls have great applicability to this adversary, yet almost no applicability to hackers, for example.

The previous section indicated a few actions that a utility may take to mitigate the risk posed by a particular adversary. In cybersecurity, any action taken to mitigate a risk is called a Control. They may be physical, technical, or administrative in nature. Physical controls include fences, locks, keying and badging systems, and security cameras. Technical controls include passwords, biometric identification, firewalls, and encryption. Administrative controls include a good cybersecurity strategy, policies, plans, and procedures.

Controls are how one executes the Core Functions of Protect, Detect, Respond, and Recover.

When implementing controls, one must ensure that they address the assessed risk and provide the proper levels of Confidentiality, Integrity, and Availability. Consider an interface between a transmission network and a distribution network. The utility assessed the system and determined that the required level of Confidentiality was Low, the required Integrity was High, and the required Availability was High. The MLCOA was a competitor would attempt to gain an unfair advantage by looking into the utility’s operations, and the MDCOA was a terrorist attack trying to bring down the transmission system and sending North America in to the Stone Age. Any controls used must address these risks and service levels. An overzealous security engineer may completely disconnect the interface from the network, requiring that a technician be physically located at the interface inside a locked room with an armed guard in order to defend against the potentially catastrophic MDCOA. Such a decision, however, would fail to meet the high level of Availability required.

Examples of controls include the following: [19]

- Inventory of authorized and unauthorized devices
- Inventory of authorized and unauthorized software
- Secure configurations for hardware and software on mobile devices, laptops, workstations, and servers
- Continuous vulnerability assessment and remediation
- Malware defenses
- Application software security
- Wireless access control
- Data recovery capability
- Security skills assessment and appropriate training to fill gaps
- Secure configurations for network devices such as firewalls, routers, and switches
- Limitation and control of network ports, protocols, and services
- Controlled use of administrator privileges
- Boundary defense
- Maintenance, monitoring, and analysis of audit logs
- Controlled access based on need to know
- Account monitoring and control
- Data protection
- Incident response and management
- Secure network engineering
- Penetration tests and red team exercises

The categories of controls (Physical, Technical, and Administrative) also apply to the different categories of attack vectors. An attack vector is simply the route by which an adversary can access a system. Physical vectors include doors, windows, and physical hardware such as laptops. A technical vector may be a flaw in computer code, a weak password, or an open port on a firewall. An administrative vector may be allowing an excessive number of users with administrator privileges or poor data backup procedures.
<table>
<thead>
<tr>
<th>Adversary</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nation States</td>
<td>“State-run, well organized and financed. Use foreign service agents to gather classified or critical information from countries viewed as hostile or as having an economic, military or a political advantage.”</td>
</tr>
<tr>
<td>Hackers</td>
<td>“A group of individuals (e.g., hackers, phreakers, crackers, trashers, and pirates) who attack networks and systems seeking to exploit the vulnerabilities in operating systems or other flaws.”</td>
</tr>
<tr>
<td>Terrorists/ Cyberterrorists</td>
<td>“Individuals or groups operating domestically or internationally who represent various terrorist or extremist groups that use violence or the threat of violence to incite fear with the intention of coercing or intimidating governments or societies into succumbing to their demands.”</td>
</tr>
<tr>
<td>Organized Crime</td>
<td>“Coordinated criminal activities including gambling, racketeering, narcotics trafficking, and many others. An organized and well-financed criminal organization.”</td>
</tr>
<tr>
<td>Other Criminal Elements</td>
<td>“Another facet of the criminal community, which is normally not well organized or financed. Normally consists of few individuals, or of one individual acting alone.”</td>
</tr>
<tr>
<td>Industrial Competitors</td>
<td>“Foreign or domestic corporations operating in a competitive market and engaged in illegal information gathering from competitors or foreign governments in the form of corporate espionage.”</td>
</tr>
<tr>
<td>Disgruntled Employees</td>
<td>“Angry, dissatisfied individuals with the potential to inflict harm on the Smart Grid network or related systems. This can represent an insider threat depending on the current state of the individual’s employment and access to the systems.”</td>
</tr>
<tr>
<td>Careless/ Poorly Trained Employees</td>
<td>“Those users who, either through lack of training, lack of concern, or lack of attentiveness pose a threat to Smart Grid systems. This is another example of an insider threat or adversary.”</td>
</tr>
</tbody>
</table>
1.4.3 Rural Electric Case Study

Consider the case of a rural electric provider using a legacy metering system. The utility operator has a good understanding of the metering system and its capabilities. The first thing the operator does is to assess the required levels of Confidentiality, Integrity, and Availability required by the system.

The consequences of unauthorized modification or destruction of the metering data could include mass theft of service and economic collapse for the company; therefore, High Integrity is required. The information contained in the system, however, would not be of any particular use to anyone except the utility. The consequences of unauthorized disclosure are limited, so the Confidentiality requirement is Low. Likewise, the utility only requires information from the meters once a month or so, with some flexibility as to when that information is accessed. A disruption of access would only have limited impact, so the required level of Availability is Low.

The utility has determined that two main adversaries exist: criminal elements attempting to steal power, and disgruntled employees attempting to damage the company. An initial assessment shows that the MLCOA is a power thief attacking one meter and stealing power for up to a year, and the MDCOA is a disgruntled employee bringing down the entire metering network for up to two weeks.

In addition to the MLCOA/MDCOA analysis, the company has collected incident data over the past 20 years. Using that historical data, the company determines that the Threat of a power thief is 2% and there is a 0.25% threat of a disgruntled employee. The probability of Vulnerability for a power thief is 1% and 25% for the disgruntled employee. The Consequences for a successful power thief are $500, and the Consequences for the disgruntled employee are up to $100,000. The quantified Risk for the power thief is therefore $0.10 and the quantified Risk for the disgruntled employee is $62.50.

When balancing Risk and the required levels of service, the utility decides that the Risk of the power thief does not warrant major controls. The utility decides to use Physical controls of sealed metal boxes at the meters and communications junctures, but not to use any specific Technical controls outside the data center, beyond the simple suite included in the communication equipment. In order to maintain the high levels of Integrity
required, though, the utility implements a robust storage and protection scheme on its servers, and has monthly reviews of the collected data to look for discrepancies. The boxes and limited security suite Protect the network, while random visual checks on meter boxes will Detect a problem. The data storage and protection scheme allows the utility to Respond to and Recover from an incident.

To address the relatively high risk of a disgruntled employee, the utility implements several policies to monitor network activity (Detect), require two-person authentication to perform any configuration changes on the metering network (Protect), limit administrator privileges (Protect), lock down unused ports (Protect), and to provide off-site backup data storage (Respond and Recover).

1.5 Cybersecurity Quantification [20]

Multiple sources identify quantification as a key question facing cybersecurity researchers. The Department of Homeland Security (DHS) states that Enterprise Level Metrics are a “current hard problem in INFOSEC Research,” and that “Defining effective metrics for information security (and for trustworthiness more generally) has proven very difficult, even though there is general agreement that such metrics could allow measurement of progress in security measures and at least rough comparisons between systems for security. Metrics underlie and quantify progress in all other roadmap topic areas.” [21] Yardley, et al. state that “The testing effort alone is challenging due to … a prescribed way to quantify security combined with the constantly evolving threat landscape. To close that gap, applied research and methods are needed to improve security quantification and rigorous security assessment, not only for single components but also for complex systems in which heterogeneous components constantly interact.” [4] The National Institute of Standards and Technology (NIST) says that “security is not absolute, and quantifying cybersecurity is already a hard problem.” [22] A group of senior corporate security officers identified security metrics as one of their top 3 security requirements. [23] In a whitepaper, RAND states that “most attempts to develop effective measures of cybersecurity have failed” [24]. Byres further elaborates characteristics that a security metric should have: a single metric that is easy
Johnson and Goetz identify “finding ways to effectively measure security and quantify if security is improving” as one of “three major areas in which security executives can make a significant impact in transforming their organizations” [26].

The following chapter provides a detailed review of cybersecurity quantification as discussed in technical literature.
CHAPTER 2 LITERATURE REVIEW

The most commonly published approach to quantification in literature is Probabilistic. Additionally, DHS categorizes other approaches into five general categories: Measures of effectiveness, Ideal-based metrics, Goal-oriented metrics, Quality of Protection, and Adversary-based metrics [21].

2.1 Probabilistic / Stochastic Methods

Many attempts to solve this problem have been published in the literature. The most common approach is Probabilistic. This approach is to model the state of the target system and address the probability of being in said state and the probability of transitioning to another state. The idea is that a user can thereby use those probabilities to determine the relative security of a system.

2.1.1 Mean Time To Security Failure

Madan, et al. propose a method of quantifying security, using the value of Mean Time To Security Failure (MTTSF). To understand this, one must look at the Markov chain and its embedded Discrete Time Markov Process, as shown in Figures 2.1 and 2.2. [27]

The authors do a thorough derivation, but the end result is

\[
MTTSF = \frac{h_G \cdot p_a + h_V \cdot p_a + h_A + p_m h_{MC} + h_{TR} (1 - p_m - p_a)}{1 - p_m}
\]  

(2.1)

Where: \(h_i\) is the mean sojourn time in state \(i\), and \(p_i\) is the probability of being in state \(i\).

This model relies on the various probabilities being determined experimentally. Once they are calculated, it is possible to display MTTSF as a function of \(\frac{h_G}{p_a}\), the ratio of
time spent in a known good state to the probability of attack. These are the two variables that will not be governed by system architecture.

Trivedi, et al used a probabilistic method to investigate SITAR (Scalable Intrusion Tolerant Architecture) an intrusion tolerant system [28] [29] [30].

Figure 2.1 State Transition Diagram for Intrusion Tolerant System [27]
Figure 2.2 Discreet Time Markov Process [27]
### 2.1.2 Probabilistic Risk Assessment

Probabilistic Risk Assessment (PRA) is a well-established method of quantifying risk that has been used in a variety of applications [31]. PRA is a five step process, and can be executed at three levels of rigor, as shown in Table 2.1.

Table 2.1 PRA Steps [31]

<table>
<thead>
<tr>
<th>Step</th>
<th>Description</th>
<th>Level 1 Accident Frequency</th>
<th>Level 2 Accident Progression and Source Term</th>
<th>Level 3 Offsite Consequence</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Accident frequency analysis</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>2</td>
<td>Accident progression analysis</td>
<td></td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>3</td>
<td>Source-term analysis</td>
<td></td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>4</td>
<td>Offsite consequence analysis</td>
<td></td>
<td></td>
<td>Yes</td>
</tr>
<tr>
<td>5</td>
<td>Risk calculation</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
</tbody>
</table>

Each step consists of a number of analyses, a thorough exploration of which would require the bulk of this paper. For example, Figure 2.3 shows the process of conducting a Level 1 PRA. The important application to the topic at hand is the introduction of event trees, which serve as a basis for determining the probability of a particular outcome. Each begins with an Initiating Event. The possible responses to that event determine the possible outcomes.

For example, consider an analysis of a pipe burst at a nuclear reactor. The burst pipe is the initiating event. The next question is if electric power is available, which is necessary for all responses. If electric power is available, the outcome falls to whether the Emergency Core Cooling System (ECCS) functions. If the ECCS functions, a problem can still occur if the fission product removal system fails. Even if it functions, a failure can occur if containment integrity breaks down. The corresponding severity of the spill that results can be characterized by which system failed, and a probability for
each result can then be calculated. This process can be summarized in the event tree below.

Lin, et al. and Yu, et al. applied this approach to cybersecurity for the power grid.

2.1.3 Vulnerability Graphs

Li, et al. introduce a graphical model called “Vulnerability Graphs” that works with a stochastic process to quantify security [34]. Vulnerability graphs have been previously explored in literature [35] [36] [37] [38] [39] [40], and essentially represent the results of a vulnerability scan of a system. The sample vulnerability graph in Figure 2.4 shows the evolution of a network from a secure state to a compromised state.

Figure 2.3 Process to Conduct a Level 1 PRA [31]
The authors proceed to overlay a stochastic process atop the graph, which represents the “probability that a randomly picked node is compromised when the system enters its steady state” (emphasis in original) [34].

As with MTTSF and many approaches to quantification, this one assumes historical or experimental data to determine state and transition probabilities. This is an inherent weakness of this approach; however, the only other option is to assume certain
values. Ye, et al used the approach to detect a cyber-attack, and found it to lack robustness for that reason [41].

The authors claim a number insights with practical significance; however, understanding their observations requires an understanding of mathematics that a typical user would not have. For example, they suppose that someone attempting to use this method would know what an “Erdos-Renyi random graph” is, without defining it. This limits its practicality. Probably the most useful of these observations is that, all other things equal, a node with fewer connections to other nodes is more secure than a node with more connections.

2.1.4 Mean Time to Compromise

Byres addresses the requirement for experimental data directly by proposing Mean Time To Compromise (MTTC) [25] [42] [43]. Additionally, the authors raise at least three key applications for a quantification methodology: deciding between security measures, justifying security measures to superiors, and comparing security between systems. They further state criteria that a quantification methodology should meet: understandable by management and users, represent a single number, aggregate separate sub-values, and be able to compare systems.

MTTC is an adaptation of the method that Underwriter Laboratories uses to rate the security level of safes [44]. A sample rating that a safe may receive is:

The authors propose quantifying the system based on the amount of time necessary for a beginner, intermediate, or expert adversary to conduct a successful attack. They define the steps to success by the following state-space model.
Figure 2.7 State-space Model for MTTC [25]

Table 2.2 shows example state dwell times for each state:

<table>
<thead>
<tr>
<th>Skill Level</th>
<th>B</th>
<th>P</th>
<th>C</th>
<th>I</th>
<th>A</th>
</tr>
</thead>
<tbody>
<tr>
<td>Expert</td>
<td>4.6</td>
<td>2.9</td>
<td>1.0</td>
<td>4.8</td>
<td>1.0</td>
</tr>
<tr>
<td>Intermediate</td>
<td>5.2</td>
<td>3.3</td>
<td>1.0</td>
<td>5.3</td>
<td>1.0</td>
</tr>
<tr>
<td>Beginner</td>
<td>9.5</td>
<td>7.3</td>
<td>1.0</td>
<td>10.3</td>
<td>1.0</td>
</tr>
</tbody>
</table>

Each state change from Launch to Success has a certain probability, much like in the Markov chains of MTTSF. From those state changes, the state dwell times, and the various path options from B to P, the user can calculate the MTTC.

2.1.5 Time To Compromise

McQueen, et al. present Time to Compromise (TTC), which has significant similarities to MTTC [45]. As with Byres, et al. in MTTC, they consider multiple skill levels of attackers each with the ability to exploit certain vulnerabilities. They consider a
fourth level, “Novice,” in addition to those in MTTC. They calculated the mean number of attempts necessary to compromise a system based on those attacker skill levels and compared to the total pool of vulnerabilities, as shown in Figures 2.8, 2.9, and 2.10.

Figure 2.8 TTC: Attempts vs. Vulnerability Pool [45]

Figure 2.9 TTC Probability of usable vulnerability [45]
2.2 Measures of Effectiveness

A measure of effectiveness functions by taking an action intended to reduce risk and measure the results. The only example of this approach in literature was conducted by the Institute for Defense Analyses as part of a DHS pilot program [46]. In this project, Simpson, et al. established fifteen Cyber Defense Configurations (CDCs), essentially scenarios around a network configuration. The CDCs are described in Figure 2.11.

The authors established three Threats. Threat 1 is a small group of adversaries with limited capabilities and time. Threat 2 still has limited capabilities, but has a large amount of time to conduct the attack. Threat 3 is a large, capable, coordinated effort such as a nation state. They then calculated the probability of a successful defense for each CDC against each threat, as displayed on the following page.

These probabilities were, effectively, educated guesses. The inherent subjectivity limits the application; however, the relative strengths should be of value. In short, this approach establishes a baseline network, applies certain defenses to the network, and determines the effect of those defenses on network security.
Figure 2.11 CDC Buildup [46]

Figure 2.12 Probability of Successful Defense [46]
2.3 Ideal Based Metrics

Ideal-based metrics compare the system to some ideal, such as a system with no vulnerabilities. Another example would be a system with no attack vectors or no adversaries conducting attacks.

2.3.1 Common Vulnerability Scoring System

An ideal-based metric with some level of industry acceptance is the Common Vulnerability Scoring System (CVSS) [47]. Each vulnerability has a variety of metrics described in Table 2.3.

Each metric has a value, generally ranging from 0 to 1, that corresponds to the levels associated with the metrics. A high value corresponds with a vulnerability that is easier to compromise.

For each metric, CVSS includes detailed instructions in how to assign it a level. Figure 2.13 is the flow chart to decide the attack complexity level, for example.

![Figure 2.13 CVSS Attack Vector Flow Chart [48]](image)

The metrics in Table 2.3 are combined in a variety of scores, summarized in Table 2.5. The base score ranges from 0 to 10 and evaluates the metrics of the vulnerability. The temporal score considers the availability of code or exploitation techniques. The environmental score considers the importance of the vulnerable asset.

The following example is for the DNS Kaminsky Bug (designated by NIST as CVE-2008-1447) [49] [50]. Table 2.4 summarizes the metrics of the vulnerability.
Table 2.3 CVSS Metrics [47]

<table>
<thead>
<tr>
<th>Variable</th>
<th>Metric</th>
<th>Levels</th>
</tr>
</thead>
<tbody>
<tr>
<td>( v )</td>
<td>Attack vector</td>
<td>Network, Adjacent Network, Local, Physical</td>
</tr>
<tr>
<td>( c )</td>
<td>Complexity</td>
<td>Low, High</td>
</tr>
<tr>
<td>( p )</td>
<td>Privilege required</td>
<td>None, Low, High</td>
</tr>
<tr>
<td>( u )</td>
<td>User interaction required</td>
<td>Yes, No</td>
</tr>
<tr>
<td>( i_c )</td>
<td>Impact on confidentiality</td>
<td>None, Low, High</td>
</tr>
<tr>
<td>( i_l )</td>
<td>Impact on integrity</td>
<td>None, Low, High</td>
</tr>
<tr>
<td>( i_a )</td>
<td>Impact on availability</td>
<td>None, Low, High</td>
</tr>
<tr>
<td>( e )</td>
<td>Exploitability</td>
<td>Unproven, Proof of Concept, Functional, High</td>
</tr>
<tr>
<td>( r )</td>
<td>Remediation level</td>
<td>Official Fix, Temporary Fix, Workaround, Unavailable</td>
</tr>
<tr>
<td>( rc )</td>
<td>Report confidence</td>
<td>Unknown, Reasonable, Confirmed</td>
</tr>
<tr>
<td>( R_c )</td>
<td>Security requirement for confidentiality</td>
<td>Low, Medium, High</td>
</tr>
<tr>
<td>( R_l )</td>
<td>Security requirement for integrity</td>
<td>Low, Medium, High</td>
</tr>
<tr>
<td>( R_a )</td>
<td>Security requirement for availability</td>
<td>Low, Medium, High</td>
</tr>
</tbody>
</table>

Using the values assigned in [47], the Base, Temporal, an Environmental values can then be calculated.

CVSS has been examined frequently in literature. Cheng et al examined how to scale CVSS from small systems to a larger network [51]. Mell et al discuss CVSS as a potential security standard within the software industry [52]. Premaratne, et al. point out a weakness in the applicability of CVSS to power systems, namely that it considers only vulnerabilities associated with computers and computer networks, not Intelligent Electronic Devices (IEDs) [53]. Lai and Hsia use CVSS to improve the security of an existing computer network [54].
<table>
<thead>
<tr>
<th>Metric</th>
<th>Value</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Attack Vector</td>
<td>Network</td>
<td>The attacker is sending the packets over the network.</td>
</tr>
<tr>
<td>Attack Complexity</td>
<td>High</td>
<td>The attacker must configure an authoritative source with a public IP to be routed to by the recursive server. The attacker must also beat a race condition to successfully exploit (regardless of how quick that race condition may occur).</td>
</tr>
<tr>
<td>Privileges Required</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>User Interaction</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Scope</td>
<td>Changed</td>
<td>The vulnerable component is the DNS server. The impacted component is the victim system who is unknowingly re-directed to unintended network locations based on the malicious DNS answers.</td>
</tr>
<tr>
<td>Confidentiality Impact</td>
<td>None</td>
<td>Any confidentiality is secondary.</td>
</tr>
<tr>
<td>Integrity Impact</td>
<td>High</td>
<td>The victim user has trusted a poisoned cache and is being directed to any destination the attacker wishes.</td>
</tr>
<tr>
<td>Availability Impact</td>
<td>None</td>
<td>Any availability impact is secondary.</td>
</tr>
<tr>
<td>Score</td>
<td>Conditions</td>
<td>Value</td>
</tr>
<tr>
<td>---------------</td>
<td>----------------</td>
<td>----------------------------------------------------------------------</td>
</tr>
<tr>
<td>IS&amp;C&lt;sub&gt;Base&lt;/sub&gt;</td>
<td>All</td>
<td>$1 - [(1 - i_c) * (1 - i_i) * (1 - i_a)]$</td>
</tr>
<tr>
<td>Impact</td>
<td>Scope unchanged</td>
<td>$6.42 * IS&amp;C&lt;sub&gt;Base&lt;/sub&gt;$</td>
</tr>
<tr>
<td></td>
<td>Scope changed</td>
<td>$7.52 * (IS&amp;C&lt;sub&gt;Base&lt;/sub&gt; - 0.029) - 3.25 * (IS&amp;C&lt;sub&gt;Base&lt;/sub&gt; - 0.02)^15$</td>
</tr>
<tr>
<td>Exploitability</td>
<td>All</td>
<td>$8.22 * v * c * p * u$</td>
</tr>
<tr>
<td>Base</td>
<td>Impact ≤ 0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Scope unchanged</td>
<td>min[(impact + exploitability), 10]</td>
</tr>
<tr>
<td></td>
<td>Scope changed</td>
<td>min[1.08 * (impact + exploitability), 10]</td>
</tr>
<tr>
<td>Temporal</td>
<td>All</td>
<td>Base * e * r * rc</td>
</tr>
<tr>
<td>IS&amp;C&lt;sub&gt;Modified&lt;/sub&gt;</td>
<td>All</td>
<td>$1 - [(1 - i_cR_c) * (1 - i_iR_i) * (1 - i_aR_a)]$</td>
</tr>
<tr>
<td>Modified Impact</td>
<td>Scope unchanged</td>
<td>$6.42 * IS&amp;C&lt;sub&gt;Modified&lt;/sub&gt;$</td>
</tr>
<tr>
<td></td>
<td>Scope changed</td>
<td>$7.52 * (IS&amp;C&lt;sub&gt;Modified&lt;/sub&gt; - 0.029) - 3.25 * (IS&amp;C&lt;sub&gt;Modified&lt;/sub&gt; - 0.02)^15$</td>
</tr>
<tr>
<td>Modified Exploitability</td>
<td>All</td>
<td>$8.22 * v * c * p * u$</td>
</tr>
<tr>
<td>Environmental</td>
<td>Impact ≤ 0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Scope unchanged</td>
<td>min[((modified impact + modified exploitability) * e * r * c, 10]</td>
</tr>
<tr>
<td></td>
<td>Scope changed</td>
<td>min[1.08(modified impact + modified exploitability) * e * r * c, 10]</td>
</tr>
<tr>
<td>Temporal</td>
<td>All</td>
<td>Environmental * e * r * rc</td>
</tr>
</tbody>
</table>
2.3.2 VEA-bility

Tupper, et al. propose VEA-bility to building on CVSS [53] [55]. VEA-bility considers a network’s relative security in relation to vulnerability (V), exploitability (E), and attackability (A). V is based on CVSS Impact and Temporal scores. E is based on the CVSS Exploitability score. A depends on attack graphs (an adversary-based metric). They are related as follows for a given vulnerability, \( v \):

\[
VEAbility = 10 - \frac{V + E + A}{3} \tag{2.2}
\]

\[
V = \min(10, \ln \sum e^{S(v)}) \tag{2.3}
\]

\[
S(v) = \frac{\text{Impact}(v) + \text{Temporal}(v)}{2} \tag{2.4}
\]

\[
E = \frac{\min(10, \ln \sum e^{\text{Exploitability}(v)}) \ast hs}{ns} \tag{2.5}
\]

where \( hs = \text{services in the system} \), and \( ns = \text{services communicating over the network} \), and

\[
A = \frac{10 \ast \sum \text{attack paths}}{\sum \text{network paths}} \tag{2.6}
\]

2.3.3 Other Ideal-based Metrics

CVSS is not the only approach using ideal-based metrics. Ganame and Bourgeois use the ideal of “no threat” and set three levels based on the deviation from that norm: Green, Orange, and Red, corresponding to “no threat,” “non-critical threat,” and “active intrusion” [56]. Hahn and Govindarasu use the ideal of “no exposure,” which is similar to “no vulnerability,” but also considers a state where vulnerabilities are not exposed to specific adversaries [57]. Tague, et al specifically compare the vulnerabilities of wireless networks to an ideal of “no vulnerabilities” [58]. Ten, et al specifically address vulnerabilities in SCADA systems. [59] Wang, et al propose a metric for handling zero-day software vulnerabilities, with an ideal of an infinite number of simultaneous vulnerabilities being necessary for system compromise [60]. Boyer and
McQueen discuss seven security ideals within the dimensions of the Security Group’s knowledge, Attack Group’s knowledge, Access, Vulnerabilities, Damage potential, Detection, and Recovery [61] [62] [63].

2.4 **Goal Oriented Metrics**

Goal-oriented metrics identify a goal and then measure progress toward that goal. Per [21], goal-oriented metrics are used in software engineering, and it appears to have been used for that application in the past [64] [65]. A thorough search of literature found no reference to goal-oriented metrics by that name relating to cybersecurity. For that reason, this section is only included to follow the framework laid out in [21].

2.5 **Quality of Protection**

The Department of Homeland Security identifies Quality of Protection (QoP) as an approach to quantifying cybersecurity; however, references in literature dealing with QoP and cybersecurity are scant [21]. Gerstel and Sasaki discuss its application to optical networks [66]. Their approach is to prioritize traffic based on the level of protection required, similar to Quality of Service in network applications. These priorities called Reliability of Service (RoS) classes, described Table 2.6

<table>
<thead>
<tr>
<th><strong>RoS Class</strong></th>
<th><strong>Description</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Guaranteed Protection</td>
<td>99.999% protection in the transport layer</td>
</tr>
<tr>
<td>Best Effort Protection</td>
<td>Uses less protection bandwidth</td>
</tr>
<tr>
<td>Unprotected Traffic</td>
<td>No effort at the transport layer to protect the connection in case of failure</td>
</tr>
<tr>
<td>Preemtable Traffic</td>
<td>Traffic that uses protection bandwidth</td>
</tr>
</tbody>
</table>

A series of workshops were held to discuss the topic; however, they ceased in 2009 [67]. Atzeny and Lioy propose criteria to be used to evaluate a metric [68]:

- Clarity: the measurement should be easy to interpret
• Objectiveness: the measurement should not be influenced by the measurer
• Repeatability: the measurement should yield the same result in the same circumstances
• Easiness: using the measurement should not require excessive effort
• Succinctness: the measurement should only consider relevant information

In short, this approach appears to have been abandoned in literature.

2.6 Information and Game Theory

Information and Game Theory approaches to quantification are quite popular in academia; however, it is an approach with little potential for real-world usability. It will only be addressed briefly in this document. Amin, et al use game theory to estimate security risks to cyber-physical systems [69]. Law, et al apply game theory to bridge cybersecurity and Automatic Generation Control [70]. Zonouz, et al use game theory to respond and recover to intrusions [71]. Bloch, et al address security for wireless networks using information theory and determine average secure communications rates and outage probability [72]. Pasqualetti and Zhu use information theory to optimize system security [73]. Zonouz, et al use information theory to evaluate the relative level of importance of intrusions [74].

2.7 Adversary Based Metrics

Adversary-based metrics attempt to quantify how difficult it is for an adversary to complete some task. Examples include time, resources, steps, and attempts necessary.

2.7.1 Attack Graphs

Idika and Bhargava, among others, use attack graphs (which track how an attacker can leverage vulnerabilities) to determine the most relevant metric to a system [75]. A sample attack graph is shown in Figure 2.14.
The starting location for the attacker is s, and the target, compromised state, is g. v1, v2, and v3 denote vulnerabilities exploited and c1 is another host in the system.

There are a number of approaches to quantifying security using attack graphs. Per Idika and Bhargava, two of them do not analyze the path taken in the attack graph. Network Compromise Percentage (NCP), proposed by Lippmann, et al. which identifies what fraction of a network can be compromised (as defined by an adversary gaining permissions to access a host) [76]. Using the notation presented,

\[
NCP(G) = 100 \times \frac{\sum_{c \in C} c \cdot v}{\sum_{h \in H} h \cdot v}
\]  

(2.7)

where H is the set of hosts, C ⊆ H is the subset of hosts that the attacker can compromise, and v is some value associated with the host.
Other attack graph based metrics look the route taken. The shortest route between g and s is the Shortest Path metric [77] [78]. Using the example in Figure 2.14, the Shortest Path is s-v1-g, of length \( l = 2 \), so the shortest path metric, \( SP(G) = 2 \).

Variants on this approach have addressed the effort required to exploit vulnerabilities in terms of time and resources [79] [80] [81].

The Number of Paths metric considers, cleverly enough, the number of possible paths [78]. In the case of the example above, there are two paths: s-v1-g and s-v2-c1-v3-g, so the Number of Paths metric, \( NP(G) = 2 \).

The Mean of Path Lengths Metric is, simply, the average path length [75] [82]. In the example above, s-v1-g is of length 2 and s-v2-c1-v3-g is of length 4, so the Mean Path Length metric, \( MPL(G) = \frac{\sum l}{NP(G)} = 3 \).

Idika and Bhargava extend these approaches to add Normalized Mean Path Length, Standard Deviation of Path Length, Mode of Path Lengths, and Median of Path Lengths as metrics. They develop an algorithm to determine how to combine the metrics in the most useful way. This allows two attack graphs to be compared and to state which of the two situations is most secure [75].

Wang, et al. expand on this approach to consider unknown vulnerabilities [60].

2.7.2 Petri Nets

Chen, et al use Petri nets (a type of attack tree) to model attacks on the power grid [83]. A Petri net is a graphical and mathematical method of modeling a variety of systems. It includes places (P), transitions (T), flow relationships (F), weight functions (W), and an initial state (\( M_0 \)) [84]. Additionally, Petri nets use tokens to indicate the current state (M). The authors present a Petri net to describe a mix of cyber and physical transitions that occurred around the 2003 Northeast blackout.

For this example, a cyber transition is anything that occurs in the control system, and a physical transition is any physical occurrence.

\( P_1 \) is the initial state. \( T_1 \) represents the first generating unit tripping, which led to the generator being shut down in state \( P_2 \). \( T_2 \) is the failure of the alarm system to warn the operators, resulting in \( P_3 \), a condition of degraded situational awareness. \( T_3 \) is the
first transmission line tripping, leading to increased load on the remaining lines in state $P_4$. $T_4$ occurred when the subtransmission lines became overloaded and tripped. $P_5$ reflects the loss of those lines. $T_5$ is the overload and tripping of the final transmission line, leading to blackout in $P_6$.

Figure 2.15 Petri Net Example [83]

The authors modeled an attack on a smart meter using a variety of cyber and physical methods.

Figure 2.16 Petri Net for Attack on Smart Meter [83]
T₁ through T₅ and P₁ through P₄ represent the adversary gaining physical access to the smart meter. T₆ and T₇ give the adversary access to the smart meter’s firmware in P₅. The adversary cracks the firmware in T₈, gaining access to the smart meter’s cryptographic keys in P₆. The oval in Figure 2.16 highlights the process of actually exploiting the smart meter.

Unfortunately, the authors never validated the approach beyond a “proof-of-concept” program in Python.

Liu, et al. use attack graphs and a concept called Multiple Criteria Decision Making (MCDM) to conduct a security assessment of a power system [85]. MCDM is a collection of methods used to make decisions that balance multiple objectives. Liu, et al. use two of these methods: analytical hierarchy process (AHP) and Technique for Order of Preference by Similarity to Ideal Solution (TOPSIS). AHP prioritizes the various criteria [86]. TOPSIS states ideals for each criteria and compares the aggregate geometric distance for the criteria from the ideal [87]. Liu, et al. apply attack graphs to the problem of vulnerabilities in the system due to device interconnection, and MCDM to quantify the level of security.

The authors evaluated a case study of a transmission substation. They first evaluated the various security countermeasures (C1 to C6) implemented in the substation.

<table>
<thead>
<tr>
<th>Number</th>
<th>Security Countermeasure</th>
</tr>
</thead>
<tbody>
<tr>
<td>C1</td>
<td>Intrusion detection system on communications network</td>
</tr>
<tr>
<td>C2</td>
<td>Security expansion for internal networks</td>
</tr>
<tr>
<td>C3</td>
<td>PKI certificate and access control</td>
</tr>
<tr>
<td>C4</td>
<td>Remote connections via Virtual Private Network (VPN)</td>
</tr>
<tr>
<td>C5</td>
<td>Network scan and patches</td>
</tr>
<tr>
<td>C6</td>
<td>We client scan, patches, and authentication</td>
</tr>
</tbody>
</table>
They proceeded to develop a hierarchy to evaluate a number of attack methods, calculated the system vulnerability, and determined the impact of the resources available to an attacker to the vulnerability of the system.

![Vulnerability factor hierarchy](image)

**Figure 2.17 Attack Method Development [85]**

They developed three defense schemes and compared them to the baseline. Scheme 1 included C1, C2, and C4. Scheme 2 included C3, C5, and C6. Scheme 3 included all countermeasures, C1-C6. The results below show that, for a given amount of resources available to an attacker, the vulnerability decreases with the addition of the countermeasures.

### 2.7.3 Vulnerability Index

Srivastaga, et al. propose a vulnerability index for specific nodes and interconnections in the network [88]. The index looks at five criteria:

* **Discovery (d)**: (0=not discovered, 1=discovered) this shows whether an attacker can find the device or link in question. For example, for an Ethernet connection, it requires the attacker to penetrate the network’s security perimeter.
Access \((a)\): (0=not accessible, 1=accessible) this indicates whether an attacker can use the targeted device. It is dependent upon the attacker’s skill level and the specifics of the device.

Feasibility \((f)\): (0=not feasible 1=feasible) Feasibility requires that the attacker have knowledge of the device’s manufacturer and model number, that the device has vulnerabilities known to the attacker.

Detection Threat \((t)\): (0=no threat, 1=low threat, 2=high threat) this is the probability of an attack being detected, and is dependent upon the duration of penetration required to execute the attack and the type of communications link.

Communications Speed \((s)\): (0=less than 1,200bps, 1=2,400-19,200 bps, 2=19,200bps or faster) Faster communications links are more vulnerable than slower links.
The vulnerability index \( v \) is then calculated as:

\[
v = d * f * a * (s - t + 3)
\] (2.8)

This will give each node a vulnerability rating of 0 to 5. The higher the vulnerability rating, the more vulnerable it is to attack.

This method of “quantifying” security is not a truly quantitative evaluation, because it creates an index not a measurable value, but it does have good potential for evaluating individual systems. It gives the system administrator a good rubric for determining which nodes or devices need additional support.

### 2.7.4 Attack Trees

Ten, et al use attack trees to analyze the impact of a system penetration [89]. An attack tree is similar to an attack graph, but that combines multiple attack “leaves” [90]. Each leaf represents a specific vulnerability, and can be combined with AND or OR. The authors calculated vulnerability indices, as proposed by Srivastagva, et al. [88] for two leaves: port auditing and password strength, used the results to evaluate security improvements, and then presented a case study that modeled various intrusion scenarios. They then compared the results of those various scenarios.

In the attack tree, \( G_1 \) through \( G_{43} \) represent the various vulnerabilities to be exploited. Various countermeasures were applied to each vulnerability, leading to an improvement in Vulnerability Index. Figure 2.19 shows the attack tree used, and Figure 2.20 shows the results.

### 2.7.5 Weakest Link

The weakest link approach has been used extensively in literature. It basically views the security of the system as the security of the weakest link [36] [91] [92] [93] [94] [95] [96]. While heavily explored, this is a more aggregate approach that relies on an underlying quantification methodology to identify the weakest node in the network.
2.8 Combination approach

Abraham and Nair combine the Probabilistic Approach with attack graphs (an Adversary-based metric) and the Common Vulnerability Scoring System (CVSS), an Ideal-based metric [97]. The analytical model used to combine those approaches is in Figure 2.21.
The created a network with four targets nodes, each with vulnerabilities assigned from [49], and then overlaid an attack graph with the goal of exploiting node M4. This is described in Figures 2.22 and 2.23.

The authors then calculated a variety of attack graph metrics (as described previously).

Zhang, et al. use CVSS, TTC and attack graphs applied to SCADA systems [98]. An example of the attack graphs and TTC results on the various substations is shown in Figure 2.24.
Figure 2.23 Combination Approach Attack Graph [97]

Figure 2.24 Attack Graph and TTC for Substations [98]
Holm, et al conduct a statistical analysis of CVSS data to determine the TTC of various attacks [99]. The main contribution of this approach is the fact that it was applied to a real-world exercise, Baltic Cyber Shield. The importance of this cannot be understated, as it is possibly the only real-world effort to apply quantification methods to analyze cyber-attacks.

The authors used CVSS to rate the relative security of each node, and measured the real-world TTC experienced during the exercise. The results are interesting.

Hypothesis 1 stated that the CVSS of the weakest link nodes in each system would be negatively correlated with TTC. The results are shown in Table 2.8.

<table>
<thead>
<tr>
<th>Scenario</th>
<th>Correlation</th>
<th>p (two-tailed)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.036</td>
<td>0.838</td>
</tr>
<tr>
<td>2</td>
<td>0.166</td>
<td>0.349</td>
</tr>
<tr>
<td>3</td>
<td>-0.020</td>
<td>0.913</td>
</tr>
<tr>
<td>4</td>
<td>-0.176</td>
<td>0.320</td>
</tr>
<tr>
<td>5</td>
<td>-0.168</td>
<td>0.342</td>
</tr>
<tr>
<td>6</td>
<td>-0.178</td>
<td>0.312</td>
</tr>
</tbody>
</table>

Hypothesis 2 stated that the vulnerability exposure (as measured in days) would be negatively correlated with TTC. It found a correlation of -0.284 with a two-tailed p of 0.104.

Hypothesis 3 stated that the number of vulnerabilities present in a system, characterized by CVSS, would be negatively correlated with TTC. The results are shown in Table 2.9.

Hypothesis 4 stated that VEA-bility would be positively correlated with TTC. The authors found a correlation of 0.104 with a two-tailed p of 0.557.

Hypothesis 5 stated that the level of security estimated using CVSS as applied by Lai and Hsia would be negatively correlated with TTC [80]. The correlation was -0.288 with two-tailed p = 0.099
Table 2.9 Correlation Between TTC and Number of Vulnerabilities [99]

<table>
<thead>
<tr>
<th>Metric</th>
<th>Correlation</th>
<th>( p ) (two-tailed)</th>
</tr>
</thead>
<tbody>
<tr>
<td>High Vulnerabilities</td>
<td>-0.269</td>
<td>0.124</td>
</tr>
<tr>
<td>Medium Vulnerabilities</td>
<td>-0.240</td>
<td>0.172</td>
</tr>
<tr>
<td>Low Vulnerabilities</td>
<td>-0.061</td>
<td>0.733</td>
</tr>
<tr>
<td>All Vulnerabilities</td>
<td>-0.199</td>
<td>0.260</td>
</tr>
<tr>
<td>High Vulnerabilities with exploits</td>
<td>-0.279</td>
<td>0.110</td>
</tr>
<tr>
<td>Medium Vulnerabilities with exploits</td>
<td>-0.240</td>
<td>0.171</td>
</tr>
<tr>
<td>Low Vulnerabilities with exploits</td>
<td>-0.117</td>
<td>0.509</td>
</tr>
<tr>
<td>All Vulnerabilities with exploits</td>
<td>-0.264</td>
<td>0.132</td>
</tr>
</tbody>
</table>

Hypotheses 6 stated that McQueen’s method of calculating TTC would positively correlate with the real-world TTC [27]. The measured correlation was 0.242 with a two-tailed \( p = 0.167 \).

As stated, the results are interesting. No single correlation was significant at a significance level of \( p < 0.05 \). All six hypotheses were rejected. In other words: in the only published results of a real-world exercise where of a range of quantification methods for cybersecurity was used to predict TTC, no quantification method correlated with TTC with statistical significance. Clearly, there is room for more work on the topic.

### 2.9 Conclusion

The process of validating a quantification methodology requires a proposed idea, presenting the idea analytically, testing the idea with simulations, evaluating the idea experimentally or in real-world exercises, and gaining industry acceptance for the methodology. As discussed above, very few of the proposed methodologies have progressed beyond the analytical step. It is necessary, therefore, to take the best components of the various ideas, combine them into a new methodology, and go through the remaining steps of the validation process.
CHAPTER 3 PROPOSED MODEL

As demonstrated in Chapter 2, no consensus exists on how to quantify cybersecurity. Existing methods lack robustness, fail to correlate with statistical significance in real-world scenarios, and lack general acceptance within the industry [21] [41] [99].

3.1 Contextual Framework

The approach presented here uses the best of the many approaches presented in Chapter 2, and combines them in a unique way. The objective is model that addresses the various failings of existing efforts in a way that is useful, reflects reality, has flexible applications, is secure in that its application does not give unnecessary information to adversaries, and is able to compare the levels of security of different systems.

Consider that a system has only two states: secured and compromised. If secured is the normal state, then certain criteria given below must be met in order to transition to the compromised state:

- The system must possess a vulnerability.
- The adversary must have a vector to conduct the attack.
- The adversary must possess the capability of exploiting that vulnerability.

If we consider vulnerabilities to be discrete items in a set, then we can consider a system to be compromised when:

\[ T \cap X \cap A \neq \emptyset \]  \hspace{1cm} (3.1)

where \( T \) is the set of vulnerabilities present in a system, \( X \) is the set of vulnerabilities with and attack vector, and \( A \) is the set of vulnerabilities that an attacker is capable of exploiting. Written more simply, a system is compromised when all the criteria above are met simultaneously.
3.2 Sets of Vulnerabilities

A more thorough analysis of this framework requires consideration of the full range of possible vulnerabilities. Consider $V$, the set of all possible vulnerabilities in any system. The size of set $V$ is $V$.

$V$ includes all vulnerabilities, even those not yet discovered or known to anybody. Some subset $K$ of $V$ is actually known, and is of size $K$.

![Figure 3.1 Set of All Potential Vulnerabilities](image1)

![Figure 3.2 Set of All Known Vulnerabilities](image2)

A subset $T$ of $V$ includes all of the vulnerabilities present in a target system, and is of size $T$. 

51
A subset $X$ of $T$ consists of all vulnerabilities in $T$ where there is an attack vector, and is of size $X$.

$A$, the intersection of $A$ and $X$, is the set of vulnerabilities present in a target system, where the adversary has an attack vector and the capability to exploit. It is of size $P$, as shown in Figure 3.6. Table 3.1 summarizes these sets.
Table 3.1 Summary of Vulnerability Sets

<table>
<thead>
<tr>
<th>Set Name</th>
<th>Set Size</th>
<th>Description of Vulnerabilities in Set</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>V</td>
<td>V</td>
<td>All in existence</td>
<td></td>
</tr>
<tr>
<td>K</td>
<td>K</td>
<td>All known</td>
<td>$K \subseteq V$</td>
</tr>
<tr>
<td>T</td>
<td>T</td>
<td>All present in a target system</td>
<td>$T \subseteq V$</td>
</tr>
<tr>
<td>X</td>
<td>X</td>
<td>All with an attack vector</td>
<td>$X \subseteq T$</td>
</tr>
<tr>
<td>A</td>
<td>A</td>
<td>All that an adversary can exploit</td>
<td>$A \subseteq K$</td>
</tr>
<tr>
<td>P</td>
<td>P</td>
<td>All in a system, with an attack vector, that an adversary can exploit</td>
<td>$P = A \cap X$</td>
</tr>
</tbody>
</table>
3.3 Cyber Potential

Based on the model in the previous section, the key attribute of a system that determines its level of vulnerability is $P$, the set of vulnerabilities present in the system, with an attack vector, and that an adversary is capable of exploiting. The size of this set, $P$, is the system’s Cyber Potential. It effectively represents the opportunity to conduct an attack through a particular vector. A large value for $P$ reflects a vector with more exploitable vulnerabilities.

Consider a simple scenario of a single target system, a single adversary, and a single attack vector, as shown in Figure 3.7. The adversary is capable of exploiting a set of vulnerabilities $A$. The target system has a set of vulnerabilities, $T$, of which some subset, $X$, the adversary can exploit through an attack vector. The cyber potential, $P$, is simply the size of the set created by the intersection of $A$ and $X$.

$$P = A \cap X$$

$X \subseteq T$

Figure 3.7 Simple Scenario

3.4 Cyber Resistance

Conceptually, cyber resistance is the amount of time necessary to conduct an effective attack. Consider $P$, a set of vulnerabilities that an adversary can exploit through a vector. Each vulnerability in $P$ can be characterized by the amount of time necessary to successfully exploit it, $t$. Recall that $P$ is a subset of $K$. Realistically, $t$ for each vulnerability in $K$ could and should be determined in a laboratory given standardized conditions, equipment, and procedures. This will be discussed in Chapter 5.

The unit of cyber potential is attacks.
For reasons that will be clear later in the chapter, R is simply the average value of t for all the vulnerabilities in K on that attack vector.

\[ R = \bar{t} = \frac{\sum x t}{X} \]  

(3.2)

The unit of cyber resistance is seconds.

### 3.5 Attack Rate

Attack rate, I, reflects the level of security of a specific attack vector. Conceptually, it can be viewed as:

\[ I = \frac{System\ vulnerability}{Difficulty\ to\ attack\ the\ system} \]  

(3.3)

Continuing the model presented thus far, if \( P_R \) is the number of vulnerabilities in P through a specific vector, then:

\[ I = \frac{P}{R} = \frac{P}{\bar{t}} \]  

(3.4)

I has units of attacks per second.

Figure 3.8 Simple scenario (continued)
3.6 Cyber Power

While attack rate reflects the level of security through a specific attack vector, cyber power reflects the overall level of security for the system. Consider the system in Figure 3.9, with multiple attack vectors:

![Figure 3.9 System with Multiple Attack Vectors](image)

Clearly, any measure of overall security must consider all of the attack vectors, as well as the overall vulnerability of the system. The simplest way of combining these values is:

\[ W = T \sum I \]  

(3.5)

where \( W \) is the cyber power, and \( T \) is the total number of vulnerabilities present in a system. The unit of cyber power is attacks² per second

3.7 Simplified Model

The worst case scenario assumes that an attacker has an attack vector and the capability of exploiting every vulnerability in a target system. Using this model:
\[ P = A = T = X \] \hspace{1cm} (3.6)

In this situation, a system engineer needs to only know the vulnerabilities present in a system and the number of attack vectors, \( n \). This information is available through standard penetration testing [100]. Simplifying everything yields:

\[ W = \frac{nT^2}{\bar{t}} \] \hspace{1cm} (3.7)

The concepts described above are summarized in Table 3.2.

<table>
<thead>
<tr>
<th>Concept</th>
<th>Description</th>
<th>Unit</th>
<th>Symbol</th>
<th>Mathematical Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cyber Potential</td>
<td>System vulnerability</td>
<td>Attacks</td>
<td>( P )</td>
<td>( P = A \cap X )</td>
</tr>
<tr>
<td>Cyber Resistance</td>
<td>Difficulty to attack through a vector</td>
<td>Second</td>
<td>( R )</td>
<td>( R = \bar{t} = \frac{\sum x_t}{X} )</td>
</tr>
<tr>
<td>Attack Rate</td>
<td>System vulnerability / Difficulty to attack</td>
<td>Attacks / Second</td>
<td>( I )</td>
<td>( I = \frac{P}{R} = \frac{p}{\bar{t}} )</td>
</tr>
<tr>
<td>Cyber Power</td>
<td>Quantified security</td>
<td>Attacks(^2) / second</td>
<td>( W )</td>
<td>( W = T \times \sum I )</td>
</tr>
</tbody>
</table>

3.8 Simple Example

Consider a very simple scenario, where vulnerabilities are represented by integers. There is a single target system with four adversaries conducting attacks, as in Figure 3.9. Table 3.3 describes the situation.

Additionally, consider that \( R_1 = 60s \), \( R_2 = 600s \), \( R_3 = 3600s \), and \( R_4 = 60s \).

Through vector 1, \( P_1 = 1 \) attack. The attack rate is \( I_1 = \frac{P_1}{R_1} = \frac{1 \text{ attack}}{60s} = 0.0167 \text{ att/s} \).

Through vector 2, the attack rate is \( I_2 = \frac{P_2}{R_2} = \frac{1 \text{ attack}}{600s} = 0.00167 \text{ att/s} \). Through vector 3,
the attack rate is $I_3 = \frac{P_3}{R_3} = \frac{0 \times \text{attack}}{3600s} = 0 \frac{\text{att}}{s}$. Through vector 4, the attack rate is $I_4 = \frac{P_4}{R_4} = \frac{2 \times \text{attacks}}{60s} = 0.0333 \frac{\text{att}}{s}$.

The overall level of security for the target system, Cyber Power, is $S = T \times \sum I = 6 \times (0.0167 + 0.00167 + 0.083) \frac{\text{att}}{s} = 0.31 \frac{\text{att}^2}{s}$.

### 3.9 Measures of Security

Cyber power is proposed as a method of quantifying security, that is, to define a value that can be used to accurately compare the relative security of various systems, can be used to make security decisions, must reflect reality, must be flexible enough to apply to a variety of vulnerabilities, must not reveal information about the system that can be used to penetrate it, and must apply to a variety of systems and situations. It is primarily a reflective value on a secure system.

Two values can be used to measure the security of a system in a laboratory environment or in a simulation. The first is Success Rate, $r_s$, which is the probability that an adversary is able to conduct a successful attack. The second is Difficulty, $\bar{t}$, which reflects the average amount of time consumed by an adversary while conducting an attack.

A given vulnerability can be characterized by three values: the probability that an attacker can exploit it, $p_A$, the probability that it exists in a system, $p_T$, and the difficulty of exploiting the vulnerability, $t$, for any vulnerability $V_k$.

These values can and should be determined in a laboratory using a standard attacker under standard conditions. For continuity and accuracy, a single entity should establish a pool of known vulnerabilities ($K$ essentially) and determine the appropriate characteristics for each vulnerability. This has a precedent in other types of security, namely Underwriter Laboratories method of rating and evaluating safes [44]. Such an effort requires resources beyond that currently available; however, a possible methodology will be discussed in Chapter 5.

Once those values are available, Success Rate and Cyber Power can be calculated.
Table 3.3 Example Sets

<table>
<thead>
<tr>
<th>Set Name</th>
<th>Set Size</th>
<th>Vulnerabilities in Set</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>V</td>
<td>V = 9</td>
<td>V = {1, 2, 3, 4, 5, 6, 7, 8, 9}</td>
<td></td>
</tr>
<tr>
<td>K</td>
<td>K = 8</td>
<td>K = {1, 2, 3, 4, 5, 6, 7, 8}</td>
<td>K \subseteq V</td>
</tr>
<tr>
<td>T</td>
<td>T = 6</td>
<td>T = {1, 2, 3, 4, 5, 6}</td>
<td>T \subseteq V</td>
</tr>
<tr>
<td>X₁</td>
<td>X₁ = 2</td>
<td>X₁ = {1, 2}</td>
<td>X \subseteq T</td>
</tr>
<tr>
<td>X₂</td>
<td>X₂ = 3</td>
<td>X₂ = {2, 3, 4}</td>
<td>X \subseteq T</td>
</tr>
<tr>
<td>X₃</td>
<td>X₃ = 1</td>
<td>X₃ = {5}</td>
<td>X \subseteq T</td>
</tr>
<tr>
<td>X₄</td>
<td>X₄ = 2</td>
<td>X₄ = {1, 6}</td>
<td>X \subseteq T</td>
</tr>
<tr>
<td>A₁</td>
<td>A₁ = 1</td>
<td>A₁ = {1}</td>
<td>A \subseteq K</td>
</tr>
<tr>
<td>A₂</td>
<td>A₂ = 1</td>
<td>A₂ = {4}</td>
<td>A \subseteq K</td>
</tr>
<tr>
<td>A₃</td>
<td>A₃ = 2</td>
<td>A₃ = {7}, {8}</td>
<td>A \subseteq K</td>
</tr>
<tr>
<td>A₄</td>
<td>A₄ = 4</td>
<td>A₄ = {1}, {2}, {6}, {8}</td>
<td>A \subseteq K</td>
</tr>
<tr>
<td>P₁</td>
<td>P₁ = 1</td>
<td>P₁ = {1}</td>
<td>P = A \cap X</td>
</tr>
<tr>
<td>P₂</td>
<td>P₂ = 1</td>
<td>P₂ = {4}</td>
<td>P = A \cap X</td>
</tr>
<tr>
<td>P₃</td>
<td>P₃ = 0</td>
<td>P₃ = \emptyset</td>
<td>P = A \cap X</td>
</tr>
<tr>
<td>P₄</td>
<td>P₄ = 2</td>
<td>P₄ = {1, 6}</td>
<td>P = A \cap X</td>
</tr>
</tbody>
</table>

Table 3.4 Vulnerability Characteristics

<table>
<thead>
<tr>
<th>Characteristic</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pₐ</td>
<td>For vulnerability Vₖ, the probability that Vₖ \in A.</td>
</tr>
<tr>
<td>pₜ</td>
<td>For vulnerability Vₖ, the probability that Vₖ \in T.</td>
</tr>
<tr>
<td>t</td>
<td>Amount of time necessary for an adversary to exploit vulnerability Vₖ</td>
</tr>
</tbody>
</table>

Recall that

\[ P = A \cap X \]  

(3.8)
To determine the cyber power without further knowledge of the adversary, we therefore need only consider the elements in $X$. Recall that for each element in $X$, $V_k$, $p_A$ is the probability that $V_k \in A$. Since the element is in $X$, it is also the probability that $V_k \in P$.

The success rate is simply the probability that $A$ includes at least one element in $P$. Using simple math,

$$r_s = 1 - \prod_{X} (1 - p_A) \quad (3.9)$$

The expected cyber potential, $P$, can be predicted analytically. As with $r_s$, consider only the elements in $X$, as other vulnerabilities that may be in $V$ are not of interest when determining $P$.

The expected value of $P$, then, is the probability that $A$ includes at least one element in $P$ multiplied by the size of $X$.

$$P = r_s \times X \quad (3.10)$$

The predicted cyber power can then be calculated as:

$$W = T \times \sum I = T \times \sum \frac{P}{t} = T \times \sum \frac{r_s \times X}{t} \quad (3.11)$$

Note that, for the special case of a single attack vector:

$$W = \frac{T \times r_s \times X}{t} \quad (3.12)$$

That is, in the case of a single attack vector,
$W \propto \frac{r_s}{t}$

(3.13)

Since $r_s$ and $\bar{t}$ are the measures of security, it holds that $W$ will reflect them.

Predicting the average time required, $\bar{t}$, is not deterministic without knowing $A$. Knowledge of $A$ is required to include the unknown elements of $A$ through which and adversary would need to iterate while attacking. Without more information, $\bar{t}$ when $P = \emptyset$ would be 0. When a successful attack occurs, $\bar{t}$ will not include the failed attempts that preceded the successful attack.

A simple approach is to assign a large number to $\bar{t}$, $t_L$, when $P = \emptyset$ to reflect the time wasted, and to add some average amount of expended time to the expected $\bar{t}$, $t_w$, during a successful attack.

In that case, an estimate for the approximate, expected amount of time necessary to conduct an attack is:

$$\bar{t}' = t_L(1 - r_s) + \sum p_A(t_w + t)$$

(3.14)

These values are summarized in Table 3.5.

### 3.10 Practical example

Consider the marketing operations center at a utility with the following network, as shown in Figure 3.10.

During a security audit, the security manager discovers a number of attack vectors and vulnerabilities, as shown in Table 3.6.

Using this model, the network looks like the one shown in Figure 3.9.
<table>
<thead>
<tr>
<th>Concept</th>
<th>Description</th>
<th>Unit</th>
<th>Symbol</th>
<th>Mathematical Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Success rate</td>
<td>Probability of successful attack</td>
<td>None</td>
<td>$r_s$</td>
<td>$r_s = 1 - \prod_{X} (1 - p_A)$</td>
</tr>
<tr>
<td>Expected Cyber Potential</td>
<td>Expected system vulnerability</td>
<td>Attacks</td>
<td>$\bar{p}$</td>
<td>$\bar{p} = r_s \cdot X$</td>
</tr>
<tr>
<td>Cyber Power</td>
<td>Quantified security</td>
<td>Attacks$^2$ / Second</td>
<td>$W$</td>
<td>$W = T \cdot \sum I = T \cdot \sum \frac{\bar{p}}{t_R}$</td>
</tr>
<tr>
<td>Average difficulty</td>
<td>Predicted amount of time necessary to conduct an attack</td>
<td>Seconds</td>
<td>$\bar{t}$</td>
<td>Not deterministic</td>
</tr>
<tr>
<td>Approximate average difficulty</td>
<td>Approximate amount of time necessary to conduct an attack</td>
<td>Seconds</td>
<td>$\bar{t}'$</td>
<td>$\bar{t}' = t_L (1 - r_s) + \sum_{X} p_A (t_w + t)$</td>
</tr>
</tbody>
</table>

Table 3.5 Measures of Security
Figure 3.10 Example network

Table 3.6 Example vulnerabilities

<table>
<thead>
<tr>
<th>Vector</th>
<th>Vulnerability</th>
<th>Type</th>
<th>$\nu_v$</th>
<th>$\nu_r$</th>
<th>$t_k$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Web</td>
<td>File server</td>
<td>Technical</td>
<td>0.0001</td>
<td>0.5</td>
<td>1 year</td>
</tr>
<tr>
<td></td>
<td>Database server</td>
<td>Technical</td>
<td>0.001</td>
<td>0.5</td>
<td>1 month</td>
</tr>
<tr>
<td></td>
<td>Workstation</td>
<td>Technical</td>
<td>0.001</td>
<td>0.1</td>
<td>1 month</td>
</tr>
<tr>
<td></td>
<td>Weak password</td>
<td>Administrative</td>
<td>0.01</td>
<td>0.1</td>
<td>1 day</td>
</tr>
<tr>
<td>Workstations</td>
<td>Missing keys</td>
<td>Physical</td>
<td>0.01</td>
<td>0.5</td>
<td>1 day</td>
</tr>
<tr>
<td></td>
<td>Unsecured USB ports</td>
<td>Physical</td>
<td>0.1</td>
<td>0.5</td>
<td>1 hour</td>
</tr>
<tr>
<td>Server room</td>
<td>Single-person access</td>
<td>Administrative</td>
<td>0.01</td>
<td>0.1</td>
<td>1 hour</td>
</tr>
<tr>
<td>Employees</td>
<td>Corruption</td>
<td>Administrative</td>
<td>0.0001</td>
<td>0.01</td>
<td>1 year</td>
</tr>
<tr>
<td>No vector</td>
<td>Offline database</td>
<td>Technical</td>
<td>0.01</td>
<td>0.5</td>
<td>1 month</td>
</tr>
<tr>
<td></td>
<td>Offsite, offline storage</td>
<td>Physical</td>
<td>0.0001</td>
<td>0.5</td>
<td>1 year</td>
</tr>
</tbody>
</table>
From that information, it is clear that $T = 10$. For the four vectors, $X_1 = 4$, $X_2 = 2$, $X_3 = 1$, $X_4 = 1$. Cyber resistance along each vector is the average amount of time necessary to conduct an attack, so $R_1 = 9.2 \times 10^6 s$, $R_2 = 4.5 \times 10^4 s$, $R_3 = 3600 s$, $R_4 = 3.2 \times 10^7 s$. This omits the correction for a failed attack, since the numbers used are quite large.

Recall that:

$$r_s = 1 - \prod_{X} (1 - p_A)$$  \hspace{1cm} (3.9)

and

$$\bar{P} = r_s \times X$$  \hspace{1cm} (3.10)

Therefore, $\bar{P}_1 = 0.012$ attacks, $\bar{P}_2 = 0.109$ attacks, $\bar{P}_3 = 0.01$ attacks, and $\bar{P}_4 = 0.0001$ attacks.

Recalling that

$$I = \frac{P}{R} = \frac{P}{t}$$  \hspace{1cm} (3.4)

yields $I_1 = 1.30 \times 10^{-9} \text{ att/s}$, $I_2 = 2.42 \times 10^{-6} \text{ att/s}$, $I_3 = 2.78 \times 10^{-6} \text{ att/s}$, and $I_4 = 3.13 \times 10^{-12} \text{ att/s}$.

with

$$W = T \times \sum I$$  \hspace{1cm} (3.5)

the cyber power for the system, $W = 5.20 \times 10^{-5} \text{ att}^2/s$

Chapter 4 will implement this model and expand on the concepts presented in Chapter 3.
CHAPTER 4 MODELING, SIMULATION, AND RESULTS

The core hypothesis of this dissertation is that the methodology presented in Chapter 3 accurately quantifies cybersecurity in a manner that is useful, reflects reality, has flexible applications, is secure in that its application does not give unnecessary information to adversaries, and is able to compare the levels of security of different systems.

4.1 Evaluation Criteria

As discussed in Chapter 1, there is a no consensus as to a methodology for quantifying cybersecurity. As shown in Chapter 2, some criteria have been proposed; however, there is no indication that they have been widely adopted. For that reason, part of the process of developing the methodology presented in Chapter 3 requires the concurrent development of the criteria for evaluating its efficacy.

<table>
<thead>
<tr>
<th>Criterion</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accuracy</td>
<td>Any methodology that purports to quantify cybersecurity must, in fact, quantify cybersecurity.</td>
</tr>
<tr>
<td>Utility</td>
<td>Users of the methodology must be able to use it to make decisions that impact security.</td>
</tr>
<tr>
<td>Reality</td>
<td>The quantified values must be observable and measurable, not merely theoretical abstractions.</td>
</tr>
<tr>
<td>Flexibility</td>
<td>The methodology must be model a wide range of technical, administrative, and physical vulnerabilities.</td>
</tr>
<tr>
<td>Security</td>
<td>Quantifying a system must not reveal information that aids an adversary in conducting an attack.</td>
</tr>
<tr>
<td>Portability</td>
<td>The methodology must be able to compare the relative levels of security of different systems.</td>
</tr>
</tbody>
</table>
The concept of security requires some further elaboration. There is an inherent contradiction in the requirement to quantify a system’s level of security and the necessity to limit information available to an adversary. At a minimum, publishing a value that meets the other criteria will tell an adversary which systems are most susceptible to exploitation. For the sake of this analysis, a methodology that meets the security criterion will only reveal that minimum amount of information. It will not give any information to an adversary that will allow it to optimize its attacks.

4.2 Testing Methodology

Each evaluation criterion requires its own testing methodology as described below. All software was written using Python 3.4. Python is an object-oriented programming language that is available for free. Vulnerabilities can be characterized easily using objects, making this a suitable choice in programming languages. The Python code was generally used to output to a .csv (comma separated value) file, that could be manipulated in Excel for data analysis and presentation.

4.2.1 Accuracy

Hypothesis 1: Cyber Power will actively predict security; that is, it will correlate with attacker success rate and correlate negatively with difficulty.

Evaluation of the model’s accuracy is conducted in 5 steps.

Step 1: Model attacks in software.

The first step is to develop software to model an attack. Recall from section 3.1 that a successful attack occurs when:

- The system possesses a vulnerability.
- The adversary has a vector to conduct the attack.
- The adversary possesses the capability of exploiting that vulnerability.
- The adversary is actively conducting an attack.

In order to model this situation, consider a simple case where an adversary is capable of exploiting A vulnerabilities, and has only a single attack vector where it can
attack a single vulnerability present in the target system. For this step, assume that $T = X$, since it is only considering a single attacker, a single target, and a single vector.

Consider $A$ to be the integers in the range $1…A$, and $X$ to be a single, randomly determined, integer in $A$. The attack process consists of iterating through $A$ until $A_n \in X$, where $A_n$ is the element of $A$ under consideration. The important value here is the number of iterations necessary to reach that condition.

The code to conduct this test is found in Appendix B, Section 1. The code outputs to a file that can be used to display the data in another program.

The simulation was run 100 times each for $A$ in the range $2…100$. Using simple mathematics, the expected value for the number of iterations before $A_n \in X$ is $(A+1)/2$.

As is evident from Figure 4.2, the expected number of iterations before a successful attack matches closely with the prediction.

**Step 2: Create set of attacker capabilities**

Step 1 demonstrated a very simple method for conducting an attack in a very simple scenario. Step 2 yields a more realistic representation of $V$ and $A$ with the potential for multiple vulnerabilities with attack vectors to be present in $T$. For this step, assume that $T = X$, since it is only considering a simple scenario of a single attacker, a single target, and a single attack vector.
Figure 4.2 Test 1 Results

Figure 4.3: Step 2 Attack Scenario
In this case, \( A \) is a randomly generated set of integers in the range \( 1 \ldots V \). \( X \) is also a random integer in the range \( 1 \ldots V \). \( X \) is the set of integers in the range \( 1 \ldots X \). The attack is conducted when the adversary iterates through \( A \) until \( A_k \in X \).

This scenario has three possible outcomes, described in Table 4.2.

<table>
<thead>
<tr>
<th>Outcome</th>
<th>Successful attack?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>( P = A \cap X = \emptyset )</td>
<td>No</td>
<td>No successful attack can occur</td>
</tr>
<tr>
<td>(</td>
<td>P</td>
<td>=</td>
</tr>
<tr>
<td>(</td>
<td>P</td>
<td>=</td>
</tr>
</tbody>
</table>

In this case, there are two interesting values; whether an attack is successful and the number of iterations necessary to conduct the attack, both in relation to the cyber power and simplified Cyber Power. In both cases, it was assumed for simplification that \( \bar{t} = 1 \).

The code to conduct this test is found in Appendix B, Section 2.

The simulation was run 40 times each for \( V \) in the range \( 1 \ldots 40 \) and, for each value of \( V \), \( X \) in the range \( 1 \ldots V \) and \( A \) in the range \( 1 \ldots V \). Larger sample sizes required excessive time to compile. Figure 4.4 shows the cyber power vs. success rate.

Note that, above a certain value for cyber power, the success rate is 100%. Figure 4.5 is the same data, but only for values of cyber power under 100.

These graphs shows several interesting details.

1. As previously noted, for any value of cyber power over a certain value, the success rate is 100%. Effectively, any adversary can penetrate the system.
2. Even at very low levels of cyber power, some adversaries will successfully penetrate the system. These correspond to extremely capable adversaries. In the case of this simulation, a low number such as 1 or 2 is found in \( A \), even though \( A \) and \( X \) may be very small.
3. There is a distinctive logarithmic curvature to the scatter plot. Below a certain threshold, even a small change in cyber power can have a large effect on success rate.

Figure 4.4 Test 2 Cyber Power vs. Success Rate
Figure 4.5 Test 2 Cyber Power vs. Success Rate, Detailed View
Figure 4.6 shows the cyber power vs. difficulty, as measured in attempts.

A few observations:

1. As with cyber power vs. success rate, there is a distinctive logarithmic curve to the scatter plot.
2. Also above a certain value of cyber power, the number of attempts becomes quite low. Compared to success rate, however, the drop off is relatively gradual. This gradual degradation is due to the fact that, when $A$ is large and $X$ is not as large, the adversary must cycle through attacks that are not successful before conducting a successful attack.

**Step 3: Add a set of target vulnerabilities**

Step 2 created a set of attacker capabilities, $A$, but the set of target vulnerabilities with a vector, $X$, was simplified. Step 3 creates $X$ in a similar manner to the method used in Step 2 to create $A$. $X$ in Step 3 is a randomly generated set of integers in the range 1…$V$. The only difference between this step and step 2 is the method of generating $X$. For this step, assume that $T = X$, since it is only considering a simple scenario of a single attacker, a single target, and a single attack vector.

![Figure 4.7 Step 3 Attack Scenario](image)

The code to conduct this test is found in Appendix B, Section 3. The code outputs to a file that can be used to display the data in another program.
The simulation was run 40 times each for $V$ in the range $1...40$ and, for each value of $V$, $X$ in the range $1...V$ and $A$ in the range $1...V$. The following figures show the results, which are quite similar to those found in step 2.

![Graph](image)

Figure 4.8 Test 3 Cyber Power vs. Success Rate
Step 4: Describe the vulnerabilities

Step 3 created the shell of a realistic scenario, wherein there is a global pool of vulnerabilities, $\mathcal{V}$, a subset of which are present in a target, $\mathcal{T}$, a further subset of which have an attack vector, $\mathcal{X}$, and an adversary has the ability to exploit $\mathcal{A}$, a separate
subset of \( V \). Step 4 refines this model by actually describing each vulnerability, \( V_n \), using three characteristics:

<table>
<thead>
<tr>
<th>Characteristic</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Attacker probability, ( p_{A,n} )</td>
<td>Probability that ( V_n \in A )</td>
</tr>
<tr>
<td>Target probability, ( p_{T,n} )</td>
<td>Probability that ( V_n \in T )</td>
</tr>
<tr>
<td>Difficulty, ( t_n )</td>
<td>The amount of time some standard attacker would require to exploit the vulnerability under some standard conditions.</td>
</tr>
</tbody>
</table>

For this step, assume that \( T = X \), since it is only considering a simple scenario of a single attacker, a single target, and a single attack vector.

The attack scenario in Step 4 is nearly identical to that in step 3. The primary difference is that, instead of \( V \) being a set of integers, it is a set of objects. Generating \( A \) and \( X \) requires that \( p_A \) and \( p_T \) be considered. Conceptually, the elements of \( A \) are more likely to have high values of \( p_A \). The same relationship exists between \( X \) and \( p_T \).

This scenario has two values of interest: whether a successful attack occurs, and the amount of time necessary to conduct the successful attack. If an adversary iterates through \( A \) \( n \) times before conducting a successful attack, then the amount of time, \( t \), necessary to conduct a successful attack is:

\[
t = \sum_{i=1}^{n} t_i
\]

(4.1)

where \( t_i \) is the amount of time necessary to conduct each iteration.

The code to conduct this test is found in Appendix B, Section 4. The code outputs to a file that can be used to display the data in another program.

The simulation was run 40 times each for \( V \) in the range 1…40 and, for each value of \( V \), \( X \) in the range 1…\( V \) and \( A \) in the range 1…\( V \). Figures 4.10 and 4.11 display the results.
Test 4 shows the same pattern as tests 2 and 3, save that the logarithmic curve is less distinct on the success rate graph. This is due to the added consideration of attack difficulty. A few tests have $A$ containing attacks with very low values of $p_A$, $p_T$, and $\bar{r}$. That yields low values for success rate, but relatively average values for cyber power. This corresponds to an adversary having access only to vulnerabilities that occur rarely in target systems, most other adversaries do not have, and are relatively easy to use. An example in the physical world would be a copy of a key. It is improbable that it will work against any random lock, improbable that an adversary would have that particular key, but relatively easy to use in the odd chance that it matches the target lock.

![Figure 4.10 Test 4 Cyber Power vs. Success Rate](image-url)
Step 5: Determine correlation between cyber power, success rate, and difficulty.

This step considered a situation where $K$ consists of 27 possible vulnerabilities, and ten systems have some subset of those vulnerabilities. For simplicity, it is assumed that all of these vulnerabilities lie along a single attack vector; therefore, $T = K$. Table 4.4 describes the vulnerabilities.
Table 4.4 Test Vulnerabilities

<table>
<thead>
<tr>
<th>Vulnerability</th>
<th>Attacker Probability (p_A)</th>
<th>Target Probability (p_T)</th>
<th>Difficulty (t)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Vulnerability 1</td>
<td>0.001</td>
<td>0.1</td>
<td>100</td>
</tr>
<tr>
<td>Vulnerability 2</td>
<td>0.001</td>
<td>0.1</td>
<td>500</td>
</tr>
<tr>
<td>Vulnerability 3</td>
<td>0.001</td>
<td>0.1</td>
<td>1000</td>
</tr>
<tr>
<td>Vulnerability 4</td>
<td>0.001</td>
<td>0.3</td>
<td>100</td>
</tr>
<tr>
<td>Vulnerability 5</td>
<td>0.001</td>
<td>0.3</td>
<td>500</td>
</tr>
<tr>
<td>Vulnerability 6</td>
<td>0.001</td>
<td>0.3</td>
<td>1000</td>
</tr>
<tr>
<td>Vulnerability 7</td>
<td>0.001</td>
<td>0.5</td>
<td>100</td>
</tr>
<tr>
<td>Vulnerability 8</td>
<td>0.001</td>
<td>0.5</td>
<td>500</td>
</tr>
<tr>
<td>Vulnerability 9</td>
<td>0.001</td>
<td>0.5</td>
<td>1000</td>
</tr>
<tr>
<td>Vulnerability 10</td>
<td>0.01</td>
<td>0.1</td>
<td>100</td>
</tr>
<tr>
<td>Vulnerability 11</td>
<td>0.01</td>
<td>0.1</td>
<td>500</td>
</tr>
<tr>
<td>Vulnerability 12</td>
<td>0.01</td>
<td>0.1</td>
<td>1000</td>
</tr>
<tr>
<td>Vulnerability 13</td>
<td>0.01</td>
<td>0.3</td>
<td>100</td>
</tr>
<tr>
<td>Vulnerability 14</td>
<td>0.01</td>
<td>0.3</td>
<td>500</td>
</tr>
<tr>
<td>Vulnerability 15</td>
<td>0.01</td>
<td>0.3</td>
<td>1000</td>
</tr>
<tr>
<td>Vulnerability 16</td>
<td>0.01</td>
<td>0.5</td>
<td>100</td>
</tr>
<tr>
<td>Vulnerability 17</td>
<td>0.01</td>
<td>0.5</td>
<td>500</td>
</tr>
<tr>
<td>Vulnerability 18</td>
<td>0.01</td>
<td>0.5</td>
<td>1000</td>
</tr>
<tr>
<td>Vulnerability 19</td>
<td>0.1</td>
<td>0.1</td>
<td>100</td>
</tr>
<tr>
<td>Vulnerability 20</td>
<td>0.1</td>
<td>0.1</td>
<td>500</td>
</tr>
<tr>
<td>Vulnerability 21</td>
<td>0.1</td>
<td>0.1</td>
<td>1000</td>
</tr>
<tr>
<td>Vulnerability 22</td>
<td>0.1</td>
<td>0.3</td>
<td>100</td>
</tr>
<tr>
<td>Vulnerability 23</td>
<td>0.1</td>
<td>0.3</td>
<td>500</td>
</tr>
<tr>
<td>Vulnerability 24</td>
<td>0.1</td>
<td>0.3</td>
<td>1000</td>
</tr>
<tr>
<td>Vulnerability 25</td>
<td>0.1</td>
<td>0.5</td>
<td>100</td>
</tr>
<tr>
<td>Vulnerability 26</td>
<td>0.1</td>
<td>0.5</td>
<td>500</td>
</tr>
<tr>
<td>Vulnerability 27</td>
<td>0.1</td>
<td>0.5</td>
<td>1000</td>
</tr>
</tbody>
</table>
This data set includes vulnerabilities with relatively high, medium, and low values for each of the three vulnerability characteristics.

The simulation was run with ten test scenarios. The first is a base case where all 27 vulnerabilities are present. The remainder eliminate vulnerabilities with some characteristic.

Success rate, cyber power, and estimated difficulty were calculated using the method in Chapter 3. Recall that:

\[
    r_s = 1 - \prod_x (1 - p_A)
\]

(3.9)

\[
    W = T \sum I = T \sum \frac{\overline{P}}{t} = T \sum \frac{r_s * X}{t}
\]

(3.11)

and

\[
    \bar{t'} = t_L (1 - r_s) + \sum p_A (t_w + t)
\]

(3.14)

In this case, \( t_w \) is set to 1000 and \( t_L \) is set to 10,000. These values are arbitrary. The predicted values, scenario descriptions and calculated values are shown in Table 4.5.

These scenarios were run 500 times, using code modified from 4.3.1 Test 4. The code to conduct this test is found in Appendix B, Section 5. The results are displayed Table 4.6 and Figures 4.12 through 4.14.

Observationally, the data show that calculated and simulated values were generally close and cyber power correlates positively with success rate and negatively with difficulty. To prove Hypothesis 1, the correlations were calculated and a two-tailed t-test was run to determine significance. The results are shown in Table 4.7. Clearly, since \( p << 0.05 \) for all values and the correlations are all strong, the cyber power accurately reflects success rate and difficulty.
Table 4.5 Test Scenarios

<table>
<thead>
<tr>
<th>Scenario</th>
<th>Characteristic</th>
<th>Vulnerabilities Eliminated</th>
<th>Success Rate ( r_s )</th>
<th>Cyber Power ( W )</th>
<th>Estimated Difficulty ( t' )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Base</td>
<td>None</td>
<td>None</td>
<td>0.65</td>
<td>0.0329</td>
<td>5032</td>
</tr>
<tr>
<td>1</td>
<td>High</td>
<td>3,6,9,12,15,18,21,24,27</td>
<td>0.50</td>
<td>0.0300</td>
<td>5866</td>
</tr>
<tr>
<td>2</td>
<td>Medium</td>
<td>2,5,8,11,14,17,20,23,26</td>
<td>0.50</td>
<td>0.0164</td>
<td>6032</td>
</tr>
<tr>
<td>3</td>
<td>Low</td>
<td>1,4,7,10,13,16,19,22,25</td>
<td>0.50</td>
<td>0.0120</td>
<td>6165</td>
</tr>
<tr>
<td>4</td>
<td>High ((pT))</td>
<td>7,8,9,16,17,18,25,26,27</td>
<td>0.50</td>
<td>0.0169</td>
<td>6021</td>
</tr>
<tr>
<td>5</td>
<td>Medium ((pT))</td>
<td>4,5,6,13,14,15,22,23,24</td>
<td>0.50</td>
<td>0.0169</td>
<td>6021</td>
</tr>
<tr>
<td>6</td>
<td>Low ((pT))</td>
<td>1,2,3,10,11,12,19,20,21</td>
<td>0.50</td>
<td>0.0169</td>
<td>6021</td>
</tr>
<tr>
<td>7</td>
<td>High ((pA))</td>
<td>19,20,21,22,23,24,25,26,27</td>
<td>0.095</td>
<td>0.0032</td>
<td>9202</td>
</tr>
<tr>
<td>8</td>
<td>Medium ((pA))</td>
<td>10,11,12,13,14,15,16,17,18</td>
<td>0.62</td>
<td>0.0209</td>
<td>5194</td>
</tr>
<tr>
<td>9</td>
<td>Low ((pA))</td>
<td>1,2,3,4,5,6,7,8,9</td>
<td>0.65</td>
<td>0.0220</td>
<td>5018</td>
</tr>
<tr>
<td>Scen</td>
<td>Calc Power W</td>
<td>Simulated Power W</td>
<td>Calc Succ Rate, ( r_s )</td>
<td>Sim Success Rate, ( r_s )</td>
<td>Calc Est Diff ( \bar{t}' )</td>
</tr>
<tr>
<td>------</td>
<td>---------------</td>
<td>-------------------</td>
<td>----------------</td>
<td>----------------</td>
<td>----------------</td>
</tr>
<tr>
<td></td>
<td>Mean  ( \bar{t} )</td>
<td>( \sigma )</td>
<td>Mean  ( \bar{t} )</td>
<td>( \sigma )</td>
<td>Mean  ( \bar{t} )</td>
</tr>
<tr>
<td>Base</td>
<td>0.0329</td>
<td>0.0336</td>
<td>0.0240</td>
<td>0.65</td>
<td>0.664</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0.0300</td>
<td>0.0262</td>
<td>0.0298</td>
<td>0.50</td>
<td>0.436</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>0.0164</td>
<td>0.0149</td>
<td>0.0163</td>
<td>0.50</td>
<td>0.456</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>0.0120</td>
<td>0.0105</td>
<td>0.0119</td>
<td>0.50</td>
<td>0.436</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>0.0169</td>
<td>0.0146</td>
<td>0.0168</td>
<td>0.50</td>
<td>0.432</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>0.0169</td>
<td>0.0152</td>
<td>0.0167</td>
<td>0.50</td>
<td>0.450</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>0.0169</td>
<td>0.0151</td>
<td>0.0168</td>
<td>0.50</td>
<td>0.446</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>0.0032</td>
<td>0.0020</td>
<td>0.0079</td>
<td>0.095</td>
<td>0.058</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>0.0209</td>
<td>0.0205</td>
<td>0.0165</td>
<td>0.62</td>
<td>0.606</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>0.0220</td>
<td>0.0224</td>
<td>0.0160</td>
<td>0.65</td>
<td>0.664</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Figure 4.12 Test 5 Cyber Power

Figure 4.13 Test 5 Success Rate
Figure 4.14 Test 5 Difficulty

Table 4.7 Correlations

<table>
<thead>
<tr>
<th>Cyber Power Method</th>
<th>Evaluation Criteria</th>
<th>Correlation (r)</th>
<th>Significance (p &lt; 0.05)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Calculated Cyber Power</td>
<td>Calculated Success Rate</td>
<td>0.763</td>
<td>3.30 * 10^{-6}</td>
</tr>
<tr>
<td></td>
<td>Simulated Success Rate</td>
<td>0.769</td>
<td>1.38 * 10^{-5}</td>
</tr>
<tr>
<td></td>
<td>Calculated Difficulty</td>
<td>-0.784</td>
<td>6.26 * 10^{-8}</td>
</tr>
<tr>
<td></td>
<td>Simulated Difficulty</td>
<td>-0.785</td>
<td>3.58 * 10^{-7}</td>
</tr>
<tr>
<td>Simulated Cyber Power</td>
<td>Calculated Success Rate</td>
<td>0.792</td>
<td>3.15 * 10^{-6}</td>
</tr>
<tr>
<td></td>
<td>Simulated Success Rate</td>
<td>0.817</td>
<td>1.31 * 10^{-5}</td>
</tr>
<tr>
<td></td>
<td>Calculated Difficulty</td>
<td>-0.801</td>
<td>6.26 * 10^{-8}</td>
</tr>
<tr>
<td></td>
<td>Simulated Difficulty</td>
<td>-0.831</td>
<td>3.58 * 10^{-7}</td>
</tr>
</tbody>
</table>
4.2.2 Utility

As stated previously, users of this quantification methodology must be able to use it to make decisions that impact the security of the system. As security managers do not have infinite budgets, it is impractical to eliminate every potential vulnerability. A quantification method can be used to which vulnerabilities should be mitigated to maximize system security.

Reconsider Step 5 of 4.3.1. The situation presented offers an opportunity to demonstrate this. Suppose a security manager has the situation as described in the base scenario, that is, 27 vulnerabilities on a single attack vector with a variety of characteristics. The security manager should eliminate vulnerabilities based on the change in $W$, $\Delta W$. Table 4.5 shows that scenario 7 has the greatest impact on $W$, and therefore should have the greatest impact on $r_s$ and $\bar{t}$.

Scenario 7 has the greatest decrease in cyber power, the greatest decrease in success rate, and the greatest increase in difficulty. If the security manager implements scenario 7 based on calculated cyber power, it will have the greatest impact on the measures of security: success rate and difficulty. This demonstrates a simple example of the utility of this model.

4.2.3 Reality

Recall that

$$W = T * \sum l = T * \sum \frac{p}{\bar{t}} = T * \sum r_s * X$$ \hspace{1cm} (3.11)$$

Cyber power is, therefore, a function of four components: $T$, the number of vulnerabilities in a system; $X$, the number of vulnerabilities in a system with an attack vector, $r_s$, the probability of a successful attack, and $\bar{t}$, the mean difficulty of conducting an attack. Additionally, $r_s$ and $\bar{t}$ are the designated measures of security.

Proving that cyber power matches reality requires proof that those four components ($T$, $X$, $r_s$, and $\bar{t}$) reflect reality. Verifying $T$ and $X$ requires some standard
methodology of evaluating a system and counting the results. Fortunately, an entire industry exists to do this through penetration testing, and NIST and others have published standards to guide execution [101] [102] [103]. Among the outcomes of these tests are a list of vulnerabilities and attack vectors [100]. Therefore, T and X can be obtained through established testing methodologies, and can be said to reflect reality.

Proving that $r_s$, and $\bar{e}$ reflect reality is a greater challenge, and requires laboratory testing methodologies that have not been well established. The quantification methodologies MTTC and TTC propose these methodologies; however, they have not been well established or independently verified [25] [45].

A proper test bed to investigate whether cyber power reflects reality is shown in Figure 4.15.

![Figure 4.15 High-level Representation of Proposed Test Bed](image)

The test bed has three main components: the system under test, a collection of technical, physical, and administrative attack vectors, and a collection of standard adversaries. These adversaries should be structured to represent increasingly capable attackers, corresponding to the novice ($p_A = 50\%$), skilled ($p_A = 10\%$), capable ($p_A = 86$)
1%), expert \((p_A = 0.1\%)\), and elite \((p_A = 0.01\%)\) levels of skills and resources. Anything beyond that would be prohibitively expensive for practical testing. Tests can be run wherein a system under test has a specific vulnerability, the standard adversaries iterate through their tools and the vector toolbox, and a measurement is taken for the amount of time required to conduct the test. Given enough iterations, the average time can then be reported as \(\bar{t}\) and the skill level of the lowest-skill successful adversary can be reported as \(p_A\).

Recall that

\[
 r_s = 1 - \prod_{X} (1 - p_A)
\]  

(3.9)

Finding \(r_s\) depends only on \(X\) and \(p_A\), which are determined in a laboratory setting and can therefore be considered to reflect reality.

In sum, proving that cyber power reflects reality requires laboratory verification, which requires resources not available at this time.

4.2.4 Flexibility

A flexible quantification scheme is one that can apply to a variety of technical, administrative, and a physical vulnerabilities. The criteria used to evaluate vulnerabilities \((p_A, q_T, \text{and } \bar{t})\) were deliberately chosen to be measurable in a laboratory. Consider three vulnerabilities: a password of a given length (administrative), a locked door (physical), and unpatched software (technical). Some percentage of adversaries will be able to crack the password, open the door, or exploit the code in the software. Some percentage of target systems will have those vulnerabilities. Exploiting each vulnerability will require some amount of time.

The methodology described in 4.3.2 is applicable regardless of the nature of the vulnerability. Adversaries with increasing levels of capability can be established in each domain: physical, technical, administrative. Sample vectors can be established for each.
As with Reality, demonstrating flexibility requires laboratory experimentation; unfortunately, the resources are not currently available.

4.2.5 Security

To prove security, it is necessary to prove that stating a system’s cyber power will not reveal details that an adversary can use to ease exploitation. Obviously, simply stating cyber power will advertise which nodes or systems are more or less secure; however, the information cannot be used for an adversary to identify specific vulnerabilities or vectors.

Recall that

\[ W = T \sum l = T \sum \frac{p}{\bar{l}} = T \sum \frac{r_s \cdot X}{\bar{l}} \]  \hspace{1cm} (3.11)

Demonstrating security requires that demonstrating that \( W \) can have the same value for various values of \( T, X, r_s, \) and \( \bar{l} \), as well as the number of attack vectors, \( n \). Consider the following ten scenarios, using simplified values for each component.

<table>
<thead>
<tr>
<th>Scenario</th>
<th>( T )</th>
<th>( n )</th>
<th>( X )</th>
<th>( \bar{l} )</th>
<th>( r_s )</th>
<th>( W )</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>10</td>
<td>0.5</td>
<td>0.2</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td>2</td>
<td>1</td>
<td>3</td>
<td>0.1</td>
<td>0.2</td>
</tr>
<tr>
<td>3</td>
<td>4</td>
<td>1</td>
<td>3</td>
<td>24</td>
<td>0.4</td>
<td>0.2</td>
</tr>
<tr>
<td>4</td>
<td>5</td>
<td>2</td>
<td>3</td>
<td>1.5</td>
<td>0.01</td>
<td>0.2</td>
</tr>
<tr>
<td>5</td>
<td>6</td>
<td>4</td>
<td>1</td>
<td>12</td>
<td>0.12</td>
<td>0.2</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td>4</td>
<td>0.06</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>2</td>
<td>2</td>
<td>0.005</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>2</td>
<td>48</td>
<td>0.08</td>
<td></td>
</tr>
</tbody>
</table>
Note that the cyber power is the same in all 5 scenarios; however, no two values for success rate, difficulty, or number of vulnerabilities in the system are the same. A large value for cyber power can result from a large number of vulnerabilities, vulnerabilities that many adversaries can exploit, vulnerabilities that require little effort to exploit, or some combination. Reporting cyber power does not reveal any other information.

4.2.6 Portability

Demonstrating portability requires demonstrating that, for any two systems, the system with the lower value of cyber power is more secure. The challenge is the unpredictable nature of adversaries. An extremely secure system could be attacked by an extremely capable adversary, whereas an effectively unsecured system may never be attacked. The important values to compare relative security are the average success rate and the average time.

Proving portability requires demonstrating that the average success rate, $\bar{r}_s$, and the average difficulty, $\bar{t}$, meet the following criteria for all values of $W$:

$$\bar{r}_{s_1} \leq \bar{r}_{s_2} \quad \text{if} \quad W_1 < W_2 \quad \text{and} \quad \bar{t}_1 \geq \bar{t}_2 \quad \text{if} \quad W_1 < W_2$$

Consider the data in Test 4 above, and smooth the results using a 50 sample moving average. The results are shown in Figures 4.16 and 4.17.
Figure 4.16 Portability Cyber Power vs. Success Rate
This methodology would meet these criteria if the graphs had no local maxima or minima. Clearly, that standard is not met. Using a large moving average and a larger data set would likely eliminate the local minima and maxima; however, it would have the net result of camouflaging the data to reflect an untrue state.

The limited portability is likely a problem inherent to any methodology that relies on probabilities. Additionally, the introduction of T and X into the equation potentially impacts the portability. As shown in 4.3.1, however, W does correlate strongly with \( r_s \) and \( \bar{r} \); therefore, there is some value to the methodology to when used to compare
systems. It cannot be said, however, with 100% certainty, that a system with a higher value of $W$ is less secure than a system with a lower value.
CHAPTER 5 SUMMARY AND CONCLUSIONS

Chapter 4 demonstrated the analytical validity of the model and supported that with simulations. Experimental and empirical evaluation are necessary before industry acceptance can be pursued.

5.1 Results

Recall from Chapter 4 that the following criteria were used to evaluate the quantification methodology.

<table>
<thead>
<tr>
<th>Criterion</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accuracy</td>
<td>Any methodology that purports to quantify cybersecurity must, in fact, quantify cybersecurity.</td>
</tr>
<tr>
<td>Utility</td>
<td>Users of the methodology must be able to use it to make decisions that impact security.</td>
</tr>
<tr>
<td>Reality</td>
<td>The quantified values must be observable and measurable, not merely theoretical abstractions.</td>
</tr>
<tr>
<td>Flexibility</td>
<td>The methodology must be model a wide range of technical, administrative, and physical vulnerabilities.</td>
</tr>
<tr>
<td>Security</td>
<td>Quantifying a system must not reveal information that aids an adversary in conducting an attack.</td>
</tr>
<tr>
<td>Portability</td>
<td>The methodology must be able to compare the relative levels of security of different systems.</td>
</tr>
</tbody>
</table>

Chapter 4 demonstrated that this methodology clearly meets the criteria for accuracy, utility, and security. Verifying that it meets reality and flexibility requires
laboratory facilities that are not currently available. The methodology does not meet a high standard for portability, but a correlation does exist.

The initial results are promising, especially in comparison to previously published efforts as described in Chapter 2. Further work is necessary to validate Reality and Flexibility, to establish a set of vulnerabilities evaluated by $p_A$, $p_T$, and $\bar{t}$.

This dissertation has had two major impacts on the current state of research into cybersecurity quantification. The first is that it presents a new approach that takes the best features of existing methodologies, and combines them in a way that is useful and mathematically correct. The second is that it presents techniques for modeling cyber attacks mathematically using simple software that can be used to evaluate this and other approaches. Less significantly, this document aggregates the results of dozens of other attempts to quantify security, addresses in detail the problem of cybersecurity in the context of power systems, and presents a path for future research that can have impact in academia, the utility sector, the national security sector, industry, and government.

5.2 Future work

The possible applications of this methodology are great, ranging from utility in designing and running systems, to developing standards, to providing topics for countless masters’ theses. In the short term, several steps should be taken.

First, Reality and Flexibility must be verified in a laboratory. To do this, a well-organized test bed must be developed. The loose sketch of one is provided in 4.3.3. The core capability of this test bed is the ability to determine $p_A$, $p_T$, and $\bar{t}$ for a wide range of vulnerabilities.

Recall from Chapter 2 that Byres, et al. propose Mean Time To Compromise (MTTC) as a security metric. While it does not meet all the evaluation criteria established in this document, it does propose an approach that can be adapted to assist with this model [25] [42] [43]. MTTC is based on the method used by Underwriters Laboratories to rate commercial safes [44]. A sample rating that a safe may receive is shown in Figure 5.1.
This rating includes the resources used to penetrate the safe and the amount of time that it lasted before it was compromised. Those values can be easily related to $p_A$ and $t$. $p_A$ will depend on the resources required to exploit the vulnerability, and $t$ relates directly. A laboratory could set up a test bed with 6 levels of capability, corresponding to $p_A = 100\%$, $p_A = 10\%$, $p_A = 1\%$, $p_A = 0.1\%$, $p_A = 0.01\%$, and $p_A = 0.001\%$.

Similarly, the vulnerability can go through tests at each level and a rating can be given for how long it lasts under tests. For simplicity, these values can be grouped and named, as in Table 5.1.

![Figure 5.1 Sample safe rating](image)

Each level will have increased amount of resources, as measured by personnel and equipment, designed to correspond roughly to capability of a similar adversary. Vulnerabilities can then be tested at each level of capability, and the vulnerability can be given a rating that matches the results of the test. For example, if the vulnerability cannot be exploited in any reasonable time in the Class A, B, or C laboratory, but fails in 2 days in the Class D laboratory, then the vulnerability can be rated a D-3 vulnerability.

When a network undergoes penetration testing, the vulnerabilities present can be identified and rated. The security manager can then use those ratings to make decisions that will secure the network.

Another major step is to establish $K$. A good starting point is the National Vulnerability Database published by NIST and updated daily. As of 17 April 16, it includes a database of 76,067 common vulnerabilities [49]. Unfortunately, it has two limitations. First, it only addresses technical vulnerabilities. Metrics are needed for physical and administrative vulnerabilities as well. Second, it only deals with computer
networks, and does not include vulnerabilities for industrial control and automation systems. Establishing and managing such a database is the stuff of government agencies.

**Table 5.1 Vulnerability Ratings**

<table>
<thead>
<tr>
<th>Class Name</th>
<th>$p_A$</th>
<th>Time Rating</th>
<th>$t$</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>100%</td>
<td>1</td>
<td>1 minute</td>
</tr>
<tr>
<td>B</td>
<td>10%</td>
<td>2</td>
<td>1 hour</td>
</tr>
<tr>
<td>C</td>
<td>1%</td>
<td>3</td>
<td>1 day</td>
</tr>
<tr>
<td>D</td>
<td>0.1%</td>
<td>4</td>
<td>1 week</td>
</tr>
<tr>
<td>E</td>
<td>0.01%</td>
<td>5</td>
<td>1 month</td>
</tr>
<tr>
<td>F</td>
<td>0.001%</td>
<td>6</td>
<td>1 year</td>
</tr>
</tbody>
</table>

The most critical area of future research is application to real-world exercises. To date, only one attempt to apply cybersecurity quantification methods to real-world events has been published [99]. As discussed in chapter 2, the results were underwhelming. The ability to accurately measure security in a useful manner in an operational environment is the holy grail of this research area.

The final element of future work to be discussed is the need to establish some level of industry acceptance. This requires soft skills of networking and marketing, but establishes the bridge between theory and reality.
REFERENCES CITED


Available:
democrats.energycommerce.house.gov/sites/default/files/image_uploads/09141
1%20EP%20The%20American%20Energy%20Initiative%2012%20-%
%20Impacts%20of%20the%20Environmental%20Protection%20
Agency%27s%20New%20and%20Proposed%20Power%20Sector%20Regula-
ons%20on%20El. [Accessed 19 May 2014].

2013. [Online]. Available:
democrats.energycommerce.house.gov/sites/default/files/documents/Report-

dire-threat-of-cyberattack.html?_r=0. [Accessed 19 May 2014].


Stackpole, 1985, p. 28.

[16] "The Department of Homeland Security's Risk Assessment Methodology:


2014].


### APPENDIX A CYBER SECURITY STANDARDS

This appendix contains a list of cybersecurity standards for the power grid.

**Table A.1 Selected Power Grid Cybersecurity Standards**

<table>
<thead>
<tr>
<th>Publisher</th>
<th>Standard Number</th>
<th>Standard Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>NERC</td>
<td>CIP-002-5.1</td>
<td>Cybersecurity — BES Cyber System Categorization</td>
</tr>
<tr>
<td></td>
<td>CIP-003-5</td>
<td>Cybersecurity - Security Management Controls</td>
</tr>
<tr>
<td></td>
<td>CIP-004-5.1</td>
<td>Cybersecurity — Personnel &amp; Training</td>
</tr>
<tr>
<td></td>
<td>CIP-005-5</td>
<td>Cybersecurity - Electronic Security Perimeter(s)</td>
</tr>
<tr>
<td></td>
<td>CIP-006-5</td>
<td>Cybersecurity - Physical Security of BES Cyber Systems</td>
</tr>
<tr>
<td></td>
<td>CIP-007-5</td>
<td>Cybersecurity - System Security Management</td>
</tr>
<tr>
<td></td>
<td>CIP-008-5</td>
<td>Cybersecurity - Incident Reporting and Response Planning</td>
</tr>
<tr>
<td></td>
<td>CIP-009-5</td>
<td>Cybersecurity - Recovery Plans for BES Cyber Systems</td>
</tr>
<tr>
<td></td>
<td>CIP-010-1</td>
<td>Cybersecurity - Configuration Change Management and Vulnerability Assessments</td>
</tr>
<tr>
<td></td>
<td>CIP-011-1</td>
<td>Cybersecurity - Information Protection</td>
</tr>
<tr>
<td>Publisher</td>
<td>Standard Number</td>
<td>Standard Name</td>
</tr>
<tr>
<td>-----------</td>
<td>----------------</td>
<td>---------------</td>
</tr>
<tr>
<td>ISO/IEC 27001</td>
<td>Information security management systems -- Overview and vocabulary</td>
<td></td>
</tr>
<tr>
<td>ISO/IEC 27002</td>
<td>Code of practice for information security controls</td>
<td></td>
</tr>
<tr>
<td>ISO/IEC 27003</td>
<td>Information security management system implementation guidance</td>
<td></td>
</tr>
<tr>
<td>ISO/IEC 27004</td>
<td>Information security management -- Measurement</td>
<td></td>
</tr>
<tr>
<td>ISO/IEC 27005</td>
<td>Information security risk management</td>
<td></td>
</tr>
<tr>
<td>ISO/IEC 27006</td>
<td>Requirements...audit and certification of information security management systems</td>
<td></td>
</tr>
<tr>
<td>ISO/IEC 27007</td>
<td>Guidelines for information security management systems auditing</td>
<td></td>
</tr>
<tr>
<td>ISO/IEC 27008</td>
<td>Guidelines for auditors on information security controls</td>
<td></td>
</tr>
<tr>
<td>ISO/IEC 27010</td>
<td>Information security management for inter-sector and inter-organizational communications</td>
<td></td>
</tr>
<tr>
<td>ISO/IEC 27011</td>
<td>Information security management guidelines for telecommunications organizations…</td>
<td></td>
</tr>
<tr>
<td>ISO/IEC 27014</td>
<td>Governance of information security</td>
<td></td>
</tr>
</tbody>
</table>
### Table A.3 Selected Power Grid Cybersecurity Standards

<table>
<thead>
<tr>
<th>Publisher</th>
<th>Standard Number</th>
<th>Standard Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>ISO/IEC</td>
<td>ISO/IEC 27019</td>
<td>Information security management guidelines based on ISO/IEC 27002 for process control systems specific to the energy utility industry</td>
</tr>
<tr>
<td>ISO/IEC</td>
<td>ISO/IEC 27032</td>
<td>Guidelines for cybersecurity</td>
</tr>
<tr>
<td>ISO/IEC</td>
<td>ISO/IEC 27033</td>
<td>Network security</td>
</tr>
<tr>
<td>ISO/IEC</td>
<td>ISO/IEC 27035</td>
<td>Information security incident management</td>
</tr>
<tr>
<td>IEC</td>
<td>IEC 60870</td>
<td>Telecontrol equipment and systems</td>
</tr>
<tr>
<td>IEC</td>
<td>IEC 61850</td>
<td>Communication networks and systems for power utility automation</td>
</tr>
<tr>
<td>IEC</td>
<td>IEC 62351</td>
<td>Power systems management and associated information exchange</td>
</tr>
<tr>
<td>IEC</td>
<td>IEC 62443</td>
<td>Industrial communication networks - Network and system security</td>
</tr>
<tr>
<td>IEC</td>
<td>IEC 62746</td>
<td>System interfaces and communication protocol(s)...for systems connected to the Smart Grid</td>
</tr>
<tr>
<td>Publisher</td>
<td>Standard Number</td>
<td>Standard Name</td>
</tr>
<tr>
<td>------------------------------------------------</td>
<td>-----------------</td>
<td>---------------------------------------------------------</td>
</tr>
<tr>
<td>International Society of Automation (ISA)</td>
<td>ISA 99 (AKA</td>
<td>Industrial Automation and Control Systems Security</td>
</tr>
<tr>
<td></td>
<td>ANSI/ISA 62443</td>
<td></td>
</tr>
<tr>
<td>National Institute of Standards and Technology (NIST)</td>
<td>NISTIR 7628</td>
<td>Framework for Improving Critical Infrastructure Security</td>
</tr>
<tr>
<td></td>
<td>NIST SP 800 Series</td>
<td>Guidelines for Smart Grid Cybersecurity</td>
</tr>
<tr>
<td>Institute of Electrical and Electronics Engineers (IEEE)</td>
<td>1686-2013</td>
<td>Intelligent Electronic Devices Cybersecurity Capabilities</td>
</tr>
<tr>
<td></td>
<td>PC37.240</td>
<td>Cybersecurity Requirements for Substation Automation, Protection and Control Systems</td>
</tr>
</tbody>
</table>
APPENDIX B CODE

This appendix contains the code used in Chapter 4.

Section 1: Step 1 code

1. `from random import randint`
2. `from statistics import stdev`
3. `from statistics import mean`
4. `import csv`
5. `# initialize sets`
6. `As = []`
7. `means = []`
8. `output = []`
9. `# run 100 iterations at each value A = 2...100`
10. `i = 100`
11. `a = 2`
12. `A = 100`
13. `while a <= 100:`
14. `result = []`
15. `As.append(a)`
16. `for k in range(i): #conduct i attacks`
17. `v = randint(1, a)`
18. `n = 1`
19. `while n <= a: # determine when an attack is successful`
20. `if n == v:`
21. `result.append(n)`
22. `
23. `break`
24. `n += 1`
25. `if len(result) > 1:`
26. `m = mean(result)`
27. `means.append(m)`
27. a += 1

28.

29. # output to CSV file
30. b = open('test1.csv', 'w')
31. x = csv.writer(b)
32. output = []
33. data00 = ['A:']
34. data01 = ['Mean number of attempts:']
35. for item in As:
36.   data00.append(item)
37. for item in means:
38.   data01.append(item)
39. output.append(data00)
40. output.append(data01)
41. x.writerows(output)
42. b.close()
Section 2: Step 2 code

1. `from random import randint`
2. `from statistics import stdev`
3. `from statistics import mean`
4. `import csv`
5. `output = []`
6. `i = 40 #iterations`
7. `V = 40 #pool of vulnerabilities`
8. `k = 0`
9. `#clear output file`
10. `output = []`
11. `#conduct test`
12. `data00 = ['Test A']`
13. `data01 = ['Test X']`
14. 
15. `data03 = ['Simplified Power']`
16. `while k < i: #conduct i scenarios`
17. `data02 = ['Power']`
18. `data04 = ['Attempts']`
19. `data05 = ['Success']`
20. `k += 1`
21. `A = 0`
22. `while A < V: #iterate through A possibilities`
23. `A += 1`
24. `a = 0`
25. `attacks = [] #empty set for the attacker's capabilities`
26. `vectors = [] #empty set for potential attack vectors`
27. `while a < A: #generate set of attacker capabilities`
28. `z = randint(1, V)`
29. `if z not in attacks:`
attacks.append(z)
a += 1
for x in range(V): #generate size of X
    X = x + 1
power_count = sum(1 for t in range(A) if attacks[t] <= X) #calculate power
power = power_count * X
simp_power = X * X #calculate simple power
if k == 1: #append to data file
data00.append(A)
data01.append(X)
data03.append(simp_power)
data02.append(power)
if min(attacks) > X: #determine if successful attack possible
data05.append(False)
data04.append(A)
else:
    attempts = 0
    for t in range(A): #conduct attacks
        attempts += 1
        if attacks[t] <= X:
            data05.append(True)
data04.append(attempts)
brea
if k == 1:
    output.append(data00)
    output.append(data01)
    output.append(data03)
    output.append(data02)
output.append(data04)
output.append(data05)
b = open('test2.csv', 'w')
add = csv.writer(b)
add.writerows(output)
b.close()
Section 3: Step 3 code

1. `import` numpy, scipy.io
2. `import` csv
3. `from` random `import` randint
4. `from` random `import` uniform
5. `from` random `import` shuffle
6. `from` statistics `import` stdev
7. `from` statistics `import` mean
8. `output` = []
9. `i` = 40  #iterations
10. `V` = 40  #pool of vulnerabilities
11. `k` = 0
12. #clear output file
13. `output` = []
14. #conduct test
15. `data00` = ['Test A']
16. `data01` = ['Test X']
17. `data03` = ['Simplified Power']
18. `while` `k` < `i`:  #conduct i scenarios
19.     `data02` = ['Power']
20.     `data04` = ['Attempts']
21.     `data05` = ['Success']
22.     `k` +=1
23.     `A` = 0
24. `while` `A` < `V`:  #iterate through A possibilities
25.     `A` += 1
26.     `a` = 0
27.     `attacks` = []  #empty set for the attacker's capabilities
28. `while` `a` < `A`:  #generate set of attacker capabilities
29.     `z` = randint(1, `V`)
if z not in attacks:
    attacks.append(z)
    a += 1

for x in range(V):  # generate size of X
    X = x + 1
    simp_power = X * X  # calculate simple power
if k == 1:  # append to data file
    data00.append(A)
    data01.append(X)
    data03.append(simp_power)
y = 0
vulnerabilities = []  # empty set for potential attack vectors
while y < X:  # generate set of vulnerabilities
    w = randint(1, V)
    if w not in vulnerabilities:
        vulnerabilities.append(w)
        y += 1
vectors = set(attacks).intersection(vulnerabilities)  # generate vectors (P)
power_count = len(vectors)  # calculate power
power = power_count * X
data02.append(power)
if not vectors:  # determine if successful attack possible
    data05.append(False)
data04.append(A)
else:
    attempts = 0
    for t in range(A):  # conduct attacks
        attempts += 1
        if attacks[t] in vectors:
            data05.append(True)
data04.append(attempts)

break

if k == 1:
    output.append(data00)
    output.append(data01)
    output.append(data03)
    output.append(data02)
    output.append(data04)
    output.append(data05)

b = open('test3.csv', 'w')
add = csv.writer(b)
add.writerows(output)
b.close()
Section 4: Step 4 code

1. `import` numpy, scipy.io
2. `import` csv
3. `from` random `import` randint
4. `from` random `import` uniform
5. `from` random `import` shuffle
6. `from` statistics `import` stdev
7. `from` statistics `import` mean
8. `output = []`
9. `i = 40` #iterations
10. `V = 40` #size of vulnerability pool
11. `k = 0`
12. #clear output file
13. `output = []`
14. #create vulnerability pool
15. `class` Vulnerability(object):
16.     `def` `__init__`(self, name, rarity, ability, difficulty):
17.         `self.name = name`
18.         `self.rarity = rarity` #probability that a vulnerability is present
19.         `self.ability = ability` #probability that an attacker has this capability
20.         `self.difficulty = difficulty` #amount of time needed to conduct attack
21. `class` Result(object):
22.     `def` `__init__`(self, size_X, size_A, success, time, power, power_simp):
23.         `self.size_X = size_X`
24.         `self.size_A = size_A`
25.         `self.success = success`
26.         `self.time = time`
27.         `self.power = power`
28.         `self.power_simp = power_simp`
29. `V_set = []` #empty set of vulnerabilities
```python
for v in range(V):
    name = 'Vulnerability_'
    name += str(v)
    Vuln = Vulnerability(name, randint(1,99) / 100, randint(1,99) / 100, randint(1,1000))
    #uniformly distributed random probabilities between 0.01 and 0.99,
    #uniformly distributed random time between 1s and 1000s (to be modified)
    V_set.append(Vuln)

#conduct test
data00 = ['Test A']
data01 = ['Test X']
data03 = ['Simplified Power']
while k < i:  #conduct i scenarios
    data02 = ['Power']
data04 = ['Time']
data05 = ['Success']
k += 1
A = 0
while A < V:  #iterate through A possibilities
    A += 1
    a = 0
    attacks = []  #empty set for the attacker's capabilities
    while a < A:  #generate set of attacker capabilities
        shuffle (V_set)
        for item in V_set:
            z = uniform(0,1)
            if item.ability >= z and item not in attacks:
                attacks.append(item)
                a += 1
        break
```

121
for x in range(V):  #generate size of X
    X = x + 1
	times_calc = []
y = 0
vulnerabilities = []  #empty set for potential attack vectors
while y < X:  #generate set of vulnerabilities
    shuffle(V_set)
    for item in V_set:
        w = uniform(0,1)
        if item.rarity >= w and item not in vulnerabilities:
            vulnerabilities.append(item)
            y += 1
            break
vectors = set(attacks).intersection(vulnerabilities)  #generate vectors (P)
power_count = len(vectors)  #calculate power
time = 0
if not vectors:  #determine if successful attack possible
    for item in attacks:
        time += item.difficulty
    data05.append(False)
data04.append(time)
power = 0
else:
    shuffle(attacks)
    for item in attacks:  #conduct attacks
        time += item.difficulty
        if item in vectors:
            data05.append(True)
data04.append(time)
break
for item in vectors:

times_calc.append(item.difficulty)

times = mean(times_calc)

power = power_count * X / times  #calculate simple power"

simp_power = X * X / times

data02.append(power)

if k == 1:
    data00.append(A)
    data01.append(X)
    data03.append(simp_power)
    output.append(data00)
    output.append(data01)
    output.append(data03)
    output.append(data02)

b = open('test4.csv', 'w')
add = csv.writer(b)
add.writerows(output)
b.close()
9. i = 500  #iterations
10. V = 27  #size of vulnerability pool
11. k = 0
12. #clear output file
13. output = []
14. #create vulnerability pool
15. class Vulnerability(object):
16.    def __init__(self, name, rarity, ability, difficulty):
17.       self.name = name
18.       self.rarity = rarity  #probability that a vulnerability is present
19.       self_ability = ability  #probability that an attacker has this capability
20.       self.difficulty = difficulty  #amount of time needed to conduct attack
21. class Scenario(object):
22.    def __init__(self, name, vulnerabilities):
23.       self.name = name
24.       self.vulnerabilities = vulnerabilities
25. class Result(object):
26.    def __init__(self, name, power, success, time):
27.       self.name = name
28.       self.power = power
29.       self.success = success
30.       self.time = time
31. V_set = []  #empty set of vulnerabilities
32. Base = []
33. for v in range(V):
34.    name = 'Vulnerability_'
35.    vu = v + 1
36.    name += str(vu)
37.    Vuln = Vulnerability(name, randint(1,500) / 1000, randint(1,500) / 1000 , randint(1,1000))
if vu <= 9:
  Vuln.ability = 0.001
elif 10 <= vu <= 18:
  Vuln.ability = 0.01
elif 19 <= vu <= 27:
  Vuln.ability = 0.1
if vu % 3 == 0 and vu < 28:
  Vuln.difficulty = 1000
elif vu % 3 == 1 and vu < 28:
  Vuln.difficulty = 100
elif vu % 3 == 2 and vu < 28:
  Vuln.difficulty = 500
if 1 <= (vu % 9) <= 3 and vu < 28:
  Vuln.rarity = 0.1
elif 4 <= (vu % 9) <= 6 and vu < 28:
  Vuln.rarity = 0.3
elif not 1 <= (vu % 9) <= 6 and vu < 28:
  Vuln.rarity = 0.5
if vu <= 27:
  Base.append(Vuln)
V_set.append(Vuln)
#create scenarios
scenarios = []

vt = 0
while vt <= 9:
  vuln_set = []
  if vt == 0:
    name = 'Base'
scen = Scenario('Base', Base)

else:
    name = 'Scenario_'
    name += str(vt)
    for item in Base:
        if vt == 1 and item.difficulty < 1000:
            vuln_set.append(item)
        if vt == 2 and not item.difficulty == 500:
            vuln_set.append(item)
        if vt == 3 and item.difficulty > 100:
            vuln_set.append(item)
        if vt == 4 and item.rarity < 0.5:
            vuln_set.append(item)
        if vt == 5 and not item.rarity == 0.3:
            vuln_set.append(item)
        if vt == 6 and item.rarity > 0.1:
            vuln_set.append(item)
        if vt == 7 and item.ability < 0.1:
            vuln_set.append(item)
        if vt == 8 and not item.ability == 0.01:
            vuln_set.append(item)
        if vt == 9 and item.ability > 0.001:
            vuln_set.append(item)
    scen = Scenario(name, vuln_set)
    scenarios.append(scen)
    vt += 1

#conduct test
while k < i: #conduct i scenarios
    data00 = ['Scenario']

126
data01 = ['Power']
data02 = ['Success']
data03 = ['Time']
k += 1
shuffle (V_set)
attacks = []
results = []

for item in V_set:
z = uniform(1,1000) / 1000
if item.ability >= z and item not in attacks:
attacks.append(item)
break

#Conduct tests
for item in scenarios:
    vuln_set = item.vulnerabilities
    vectors = set(attacks).intersection(vuln_set) #generate vectors (P)
    power_count = len(vectors) #calculate power
    name = item.name
    if name == 'Scenario_1':
        power = power_count * 18 / 300
    elif name == 'Scenario_2':
        power = power_count * 18 / 550
    elif name == 'Scenario_3':
        power = power_count * 18 / 750
    elif name == 'Base':
        power = power_count * 27 / 533
    else:
        power = power_count * 18 / 533
    time = 0
if not vectors: #determine if successful attack possible
success = False

time = 10000

else:
    for item in attacks: #conduct attacks
        time += item.difficulty
        success = True
        time = time + 1000
        res = Result(name, power, success, time)
        results.append(res)

    for item in results:
        data00.append(item.name)
        data01.append(item.power)
        data02.append(item.success)
        data03.append(item.time)

    if k == 1:
        output.append(data00)
        output.append(data01)
        output.append(data02)
        output.append(data03)

        b = open('test5.csv', 'w')
        add = csv.writer(b)
        add.writerows(output)
        b.close()