AVOIDIT: A Cyber Attack Taxonomy

Chris B. Simmons, Sajjan G. Shiva, Harkeerat Bedi, Dipankar Dasgupta
Computer Science Department
University of Memphis
Memphis, Tennessee, USA
{cbsmmons, sshiva, hsbedi, ddasgupta}@memphis.edu

***Abstract*—Cyber-attacks have greatly increased over the years, and the attackers have progressively improved in devising attacks towards specific targets. To aid in identifying and defending against cyber-attacks we propose a cyber attack taxonomy called AVOIDIT (Attack Vector, Operational Impact, Defense, Information Impact, and Target). We use five major classifiers to characterize the nature of an attack: classification by attack vector, classification by operational impact, classification by defense, classification by informational impact, and classification by attack target. Classification by defense is oriented towards providing information to the network administrator regarding attack mitigation or remediation strategies. Contrary to the existing taxonomies, AVOIDIT efficiently classifies blended attacks. We further propose an efficient cause, action, defense, analysis, and target (CADAT) process used to facilitate attack classification. AVOIDIT and CADAT are used by an issue resolution system (IRS) to educate the defender on possible cyber-attacks and the development of potential security policies. We validate the proposed AVOIDIT taxonomy using cyber-attacks scenarios and highlight future work intended to simulate AVOIDIT's use within the IRS.**

Keywords—Taxonomy; Cyber Attack Taxonomy; Vulnerability; Computer Security; Cyberspace; Issue Resolution System

I. INTRODUCTION

Cyber-attacks have created a global threat to local and global networks. Attacks are becoming more sophisticated and possess the ability of spreading in a matter of seconds. The World Economic Forum has established cyber-attacks as a global risk within its 2013 Global Risk report [18]. Therefore, it is essential to provide tools for educating cyber defense in an expected future of cyber warfare using an adaptable language. A taxonomy defines what data is to be recorded and how like and unlike samplings are to be distinguished [1], thus enabling a mechanism for representing attack vectors. Traditional security taxonomies classify vulnerabilities, computer and network attacks, security threats, and events. We build upon prior taxonomies through the use of attack vectors, which contain the path an attacker utilizes to gain access. Amer and Hamilton [7] stated that a complete secure solution considers more than one aspect. When security taxonomies are used to generalize attacks, they are prone to neglect subsequent attacks. For example, the Windows Server service Remote Procedure Call (RPC) stack buffer overflow vulnerability was utilized by both Gimmiv.A and Conficker attacks [14]; as a result a variation of an attack was used against a preexisting vulnerability. Variants of an attack are not captured in previous taxonomies.

A blended attack exploits one or more vulnerabilities to perform an attack against a target [3]. In general, little attention has been given to successfully classifying ubiquitous blended attacks. Although, a broad view attacks is essential for various personnel, providing a holistic approach for the entire organization utilizing attack details is pertinent to ensure appropriate classification. Current security related taxonomies are in need of improvement in accordance to current industry changes involving cyber-attacks. Research has suggested the importance of developing an improved taxonomy to encompass the changing terminology of cyber security. Moreover, previous taxonomies lack the ability to show temporal and causal relationships to display the evolution of cyber related attacks. We investigate the components of a comprehensive cyber attack taxonomy that employs attack vectors to classify blended attacks, and thus supports an improvement in devising complete security defenses. Previous taxonomies lack significant defense strategies that can be used in an application setting. This can be due to the enormous possibilities of defense strategies. Landwehr et al. [1] state a taxonomy is most useful when it classifies threats in scope that correspond to potential defenses. We believe coupling defenses within an attack taxonomy would enable not only understanding the exploit, but also deriving the strategy needed to mitigate and/or remediate auxiliary damages. Providing defense strategies within an attack taxonomy presents an invaluable approach in attempting to defend the network against cyber-attacks.

Organizations may question the impact a cyber attack has once its target is compromised. One approach to gaining insight into the attacker's target is to consider the attack paths, or combination of exploits [2]. This paper presents a guide intended primarily to aid an organization decipher attack characteristics using an attack taxonomy to classify attack vectors as they are made available for analysis and defense. Further, this paper provides an efficient cause, action, defense, analysis, and target process called CADAT. At a high level, our main contribution is an enhanced cyber attack taxonomy that facilitates the classification of attack vector information, which can be considered vulnerabilities or any means by which an attack exploits to conduct an attack. This taxonomy does not provide information to determine if the attack is successful, but rather classifies the attack vectors to gain insight for appropriate countermeasures. When applied within a knowledge management capacity, the attack taxonomy is beneficial to management, network administration, development, test, and support personnel.

In this paper, we present an intuitive approach to using the AVOIDIT taxonomy using a tree structure to mobilize the parent-child relationship. We propose AVOIDIT, a cyber attack taxonomy, as a solution addressing the deficiency in existing taxonomies. AVOIDIT is intended to provide a defender with attack vector details to what encompasses an attack and any impact the attack may have on a targeted system. AVOIDIT classifies blended attacks by labeling vectors of an attack in a tree structure. The tree structure allows attack detail dissemination for an appropriate flow of information using an issue resolution system (IRS). The use of the IRS provides a repository to label attack vector information as it is stored using an iterative process. The tree like structure is formalized using information extraction and data mining techniques to display the complete attack path within the issue resolution system. In future work, we provide a prototype implementation of AVOIDIT in the form of an issue resolution system capturing web application data in a virtual environment. We use a recent test bed of attacks to show AVOIDIT consistently classifies attack vectors to produce the full path to an attack. Our results show that the AVOIDIT taxonomy provides an intuitive classification mechanism for a functional system usage.

This paper is organized as follows: In Section 2 we survey previous attack taxonomies. In Section 3, we highlight requirements for a taxonomy and propose AVOIDIT. In Section 4, we explain the tree structure of AVOIDIT to classify elements of an attack and the CADAT process therein, followed by well-known attack examples used to demonstrate the utility of AVOIDIT. In Section 5, we illustrate how AVOIDIT can be applied as an organizational element within a network setting and Section 6 presents limitations. We conclude our work in Section 7, followed by future work in Section 8.

II. Brief Survey of Attack Taxonomies

In this section we provide a brief survey of existing taxonomies that assist with identifying attacks.

Howard [8] provides an incident taxonomy that classifies attacks by events, which is an attack directed at a specific target intended to result in a changed state. The event involves the action and the target. He highlights all steps that encompass an attack and how an attack develops. The attack consists of five logical steps an attacker performs to achieve an unauthorized result. Those steps are: tools, vulnerability, action, target, and unauthorized result. The tool refers to the mechanism used to perform the attack; the vulnerability is the type of exploit used to perform attack. The action refers to the method used by the attacker to perform the attack (i.e. Probe, Scan, Authenticate, etc.). The target is the intention the attack is attempting to compromise, and the unauthorized result is the change state caused due to the attack.

Mirkovic and Reihner [10] offer a comprehensive taxonomy of Distributed Denial of Services (DDoS) attack and defense mechanisms in aim to classify attacks and defense strategies. This research highlight features of attack strategies, where the strategies are imperative in devising countermeasures. The taxonomy of DDoS attacks is categorized by Degree of Automation, Exploited Weakness, Source Address Validity, Attack Rate Dynamics, Possibility of Characterization, Persistent Agent Set, Victim Type, and Impact on Victim. In addition to classifying DDoS attacks, Mirkovic and Reihner developed a taxonomy of DDoS defenses consisting of Activity Level, Cooperation Degree, and Deployment Location.

Hansman and Hunt [6] proposed a taxonomy with four unique dimensions that provide a holistic classification covering network and computer attacks. Their taxonomy provides assistance in improving computer and network security as well as consistency in language with attack description. The first dimension being attack vector is used to classify the attack. The second dimension classifies the target of the attack. The third dimension consists of the vulnerability classification number, or criteria from Howard's taxonomy [9]. The fourth and final dimension highlights the payload or effects involved. Within each dimension various levels of information are provided to supply attack details.

Kjaerland [4] proposed a taxonomy of cyber-intrusions from Computer Emergency Response Team (CERT) related to computer crime profiling, highlighting cyber-criminals and victims. In this research, attacks were analyzed using facet theory and multidimensional scaling (MDS) with Method of Operation, Target, Source, and Impact. Each facet contains a number of elements with an exhaustive description. Kjaerland uses these facets to compare commercial versus government incidents. Kjaerland's taxonomy focuses on the motive of the attacker in an attempt to quantify why the attack takes place, and where the attack originated.

King, et al. [9] proposed a taxonomy for attacks against network log anonymization. This taxonomy is developed to capture attack pre-conditions, which is based on information necessary for the adversary to perform an attack. King, et al. organized the taxonomy by pre-conditions and specific log properties, which are Fingerprinting, Structure Recognition, Known Mapping, Data Injection, and Cryptographic. A graph is used to represent attack pre-conditions using nodes. Classifying attacks where two or more nodes share common pre-conditions, conditions are combined into a single node. King, et al. taxonomy presents a novel attack classification approach using network log anonymization

Amer and Hamilton [7] proposed a comprehensive up-to-date intrusion detection system (IDS) taxonomy, which provide insight into the technology of an IDS. They investigated the elements of an IDS being source of audit, layout technology, data processing, structure and arrangement, data collection, structure of IDS, detection paradigm, and time of detection. Amer and Hamilton stated the use of a single IDS may prevent an organization from being effective in security. The use of this taxonomy using the defined characteristics can assist organizations tailor an IDS which best suits security needs.

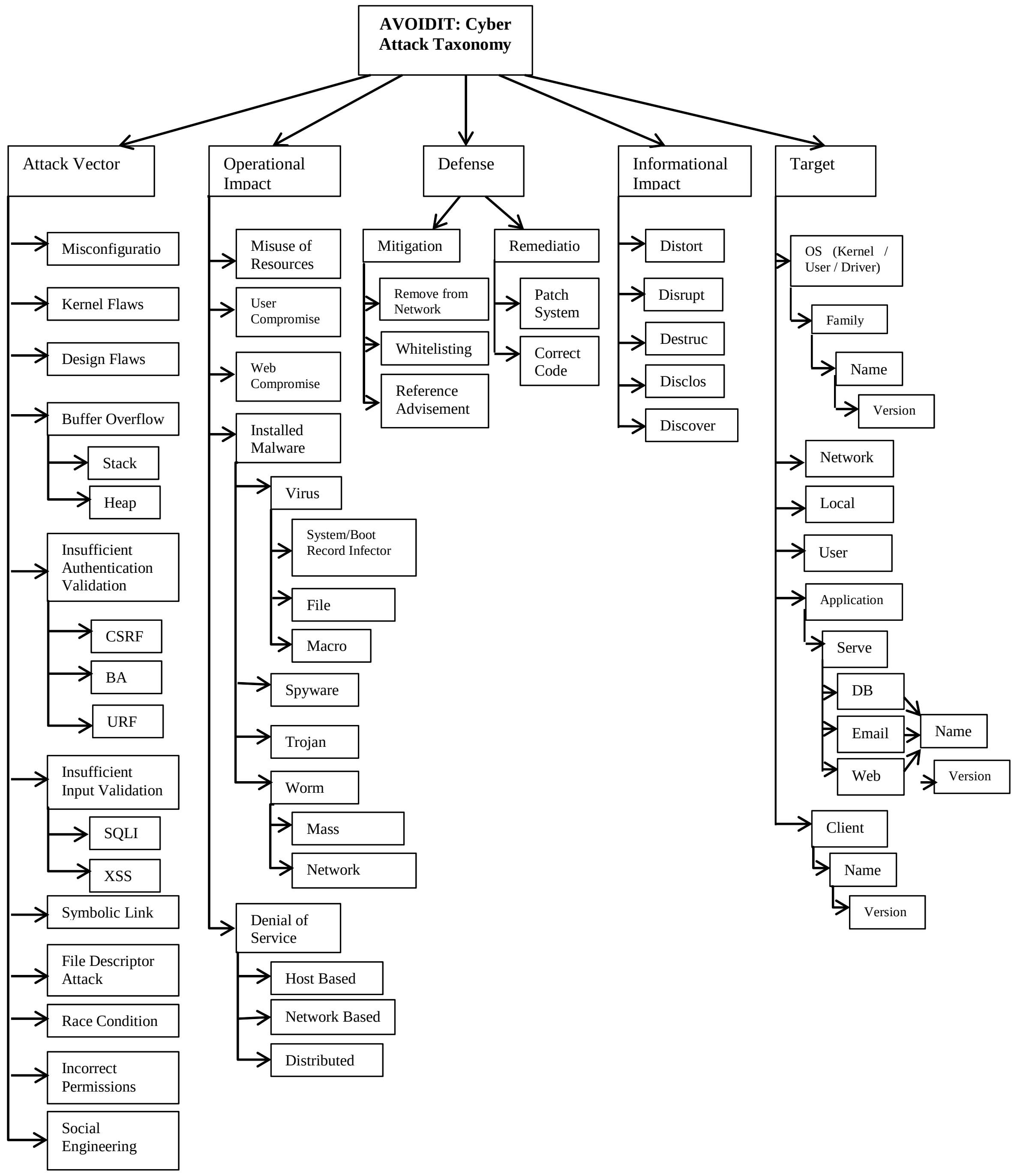


Fig. 2. AVOIDIT – A cyber attack

Meyers, et al. [15] proposed a cyber attack taxonomy similar to that of Hansman and Hunt, which provides nine classes of cyber-attacks. They focus on usage of the nine attack types to effectively classify attacks with multiple attacks using multiple attack classes, where mutually exclusiveness is not required in this research. Meyers, et al. taxonomy presents a continuous development of attack taxonomies. In this same body of work, Meyers, et al. included a taxonomy of cyber adversaries arranged in eight classes according to skill level.

III. PROPOSED CYBER ATTACK TAXONOMY: AVOIDIT

We explore the components of a comprehensive cyber attack taxonomy that employs attack vectors classifying blended attacks, and thus supports a profound improvement in devising complete security defenses. AVOIDIT couples defenses within an attack taxonomy to understand the exploit and derive strategies needed to prevent secondary damages.

A successful taxonomy should satisfy several requirements for its universal acceptance [8]. Typical requirements include the following:

- Accepted – builds on previous work that is well accepted.
- Mutually exclusive – each attack can only be classified into one category, which prevents overlapping.
- Comprehensible – clear and concise information; able to be understood by experts and those less familiar.
- Complete/exhaustive – available categories are exhaustive within each classification, it is assumed to be complete.
- Unambiguous – involves clearly defined classes, with no doubt of which class an attack belongs.
- Repeatable – the classification of attack should be repeatable.
- Terms well defined – categories should be well defined, and those terms should consist of established terminology that is compliant within the security community.
- Useful –use and gain insight into a particular field of study, particularly those having great interest within the field of study.

Although, these requirements remain complete to propose a successful taxonomy, we slightly update these requirements to include application. Application as a requirement further enhances the usefulness of a taxonomy, enabling the taxonomy to be used within a system repository permitting the use of knowledge capture.

Research has proven difficulty in developing a taxonomy that consistently satisfies mutually exclusiveness. Mutually exclusive regarding classifying cyber-attacks situate each attack into one category. We attempt to solve this critique by placing multiple associations of an attack in a tree structure allowing one category for each attack vector. We explain our tree structure further in Section 4. Applying these requirements for a complete taxonomy, we propose AVOIDIT. AVOIDIT provides, through application, a knowledge repository schema used by a defender to classify vectors that an adversary can use against a target. Fig. 1 provides an overview of our proposed taxonomy to support comprehending each attack classification and how a variety of attacks are represented in each category.

A. Classification by Attack Vector

When an attack takes place, there is a possibility it uses several vectors as a path to a full-blown cyber-attack. An attack vector is a path by which an adversary can gain unauthorized access to a host. This definition includes vulnerabilities, as it may require several vulnerabilities to launch a successful attack. The use of common vulnerabilities as a single attack vectors is a way to utilize the taxonomy as a tree structure to create an overall view of the attack vector and its impact involving a mature attack. Considering majority of attacks are not isolated events, the combination of attack vectors are used to depict the complete path of an attack.

1) Misconfiguration: An attacker can use a configuration flaw within a particular application to gain access to a network or personal computer to cause a variety of attacks. Settings that are improperly configured, usually default settings, are an easy target for an attacker to exploit [5].

2) Kernel Flaws: An attacker can use a kernel flaw within an operating system, which is the core code of an operating system, to gain certain privileges to exploit a vulnerability within the operating system.

3) Buffer Overflow: Buffer overflow is caused when a piece of code does not adequately check for appropriate input length and the input value is not the size the program expects. Cowan [11] describes a buffer overflow when a buffer with weak or no bounds checking is populated with user supplied data. An attack can exploit a buffer overflow vulnerability leading to a possible exploitation of arbitrary code execution, often of privileges at the administrative level with the program running [5]. Buffer Overflow can occur in both stack and heap memory locations. A buffer overflow constitute majority of attacks within software [11].

4) Insufficient Authentication Validation: A program fails to validate the authentication of an application and/or user sent to the program from a user. An attacker can exploit an insufficient authentication validation vulnerability and capture user credentials to impersonate a valid user, which commonly occurs within web applications.

a) Cross Site Request Forgery (CSRF): A program fails to validate the authentication of an application.

b) Broken Authentication: Authentication flaw within an application allowing the compromise of user related credentials to portray a valid user's identity. (ie. Keys, passwords, tokens, etc.) [16].

c) Unvalidated Redirects/Forwards: Validation flaw within web applications not properly validating the malicious redirects/forwards to malware sites or unauthorized pages [16].

5) Insufficient Input Validation: A program fails to validate the input sent to the program from a user [5]. An attacker can exploit an insufficient input validation vulnerability and inject arbitrary code, which commonly occurs within web applications.

a) SQL Injection (SQLI): Injection Flaws that are not properly validated and sent to an interpreter, usually due to some design flaw [16].

b) Cross-site Scripting (XSS): XSS flaws involve a design flaw not properly validated allowing malicious scripts to be executed against a vulnerable application in a web browser [16].

6) Symbolic Links: A file that points to another file [5]. An attacker can exploit a symbolic link vulnerability to point to a target file for which an operating system process has write permissions.

7) File Descriptor: A file that uses numbers from a system to keep track of files, as opposed to file names [5]. Exploitation of a file descriptor vulnerability allows an attacker the possibility of gaining elevated privileges to program related files.

8) Race Condition: Occurs when a program attempts to run a process and the object changes concurrently between repeated references allowing an attacker to gain elevated privileges while a program or process is in privilege mode [5].

9) Incorrect File/Directory Permission: An incorrect permission associated to a file or directory consists of not appropriately assigning users and processes [5]. Exploiting this vulnerability can allow a multitude of attacks to occur.

10) Social Engineering: The process of using social interactions to acquire information about a victim or computer system. These types of attacks provide quick alternatives in disclosing information to assist an attack that in normal circumstances may not be available. From research we chose Social Engineering as an attack vector exhibiting the ability for a user to be coaxed into performing functions allowing further damage.

B. Classification by Operational Impact

Classification by Operational Impact involves the ability for an attack to culminate and provide high-level information known by security experts, as well those less familiar with cyber-attacks. We provide a mutually exclusive list of operational impacts that can be categorized and concisely presented to the public. This section presents classification by operational impact.

1) Misuse of Resources: An unauthorized use of IT resources [4]. We can extend this definition to consider any IT related function that require a certain privilege and those privileges are converted into an abusive action. This provides an effect to a function and/or user to launch an attack against a resource. We use misuse of resources in an attempt to remove the miscellaneous classification, as used in previous research, with a more formal definition.

2) User Compromise: A perpetrator gaining unauthorized use of user privileges on a host, as a user compromise [4].

3) Root Compromise: Gaining unauthorized privileges of an administrator on a particular host [4]. We shall extend this notion slightly by including any elevated privileges above a normal user including administrative and/or root level privileges to a particular system.

4) Web Compromise: A website or web application using vulnerabilities to further an attack [4]. An attack can occur through a web compromise, usually via cross-site scripting or sql injection. Web Compromise involves the use of a malformed website or web application an attacker exploits for gain.

5) Installed Malware: Exploiting some vulnerability an attack can be launched via user-installed malware, whether user installed or drive-by installation. Installed malware can allow an adversary to gain full control of the compromised systems leading to the exposure of sensitive information or remote control of the host.

a) Virus: A form of installed malware, where Hansman and Hunt[6] describes a virus as a piece of code that will attach itself through some form of infected files, which will self-replicate upon execution of program. Types of viruses include boot record infectors, file infectors, and macros.

b) Spyware: A type of malware program that is covertly installed and infects its target by collecting information from a computing system without owner's consent.

c) Trojan: A benign program to the user that allows unauthorized backdoor access to a compromised system. A common way to introduce a victim into a multitude of attacks.

d) Worms: A self-replicating computer program. A considerable threat to the internet today. Worms do not require human intervention to propagate as it is a self-replicating program that spreads throughout the network. Worms include mass mailing and network aware worms.

e) Arbitrary Code Execution: Involves a malicious entity that gains control through some vulnerability injecting its own code to perform any operation the overall application has permission [13].

6) Denial of Service: Denial of Service (DoS) is an attack to deny a victim access to a particular resource or service, and has become one of the major threats and rated among the hardest Internet security issues [13]. In this section we will provide details into the types of DoS attacks.

a) Host Based: A Host based DoS aims at attacking a specific computer target within the configuration, operating system, or software of a host. These types of attacks usually involved resource hogs, aimed at consuming up all resources on a computer; crashers, which attempts to crash the host system [6].

b) Network Based: A Network based DoS targets a complete network of computers to prevent the network of providing normal services [13]. Network based DoS usually occur in the form of flooding with packets [6], where the network's connectivity and bandwidth are the target [13].

c) Distributed: A Distributed Denial of Service (DDoS) is becoming more popular as an attacker's choice of DoS. A distributed denial of service uses multiple attack vectors to obtain its goal [10].

A. Classification by Defense

We extend previous attack taxonomy research to include a defense classification. This section highlights several strategies a defender can employ to remain vigilant in defending against attacks before and after its occurrence. We provide the possibility of using both mitigation and remediation when classifying attack defenses, as an attack could be first mitigated before a remediation can occur. There are many potential defenses using this classification. As new defenses become available the system in which the AVOIDIT schema is used, will have the ability of using the optimal defense solution for mitigation and/or remediation. We take vision of the work of Mirkovic and Reihner [10] by providing a taxonomy of defenses, which readily selects relevant defenses when attack vectors are identified. The goal is to provide the defender available solutions to defend against attack vectors thwarting a potential attack.

1) *Mitigation:* Prior to vulnerability exploitation or during an attack, there are several steps a defender can use to mitigate damage an attack has caused, or has the potential to cause. An example can involve an installation of a worm that propagate over the network, one instance could be to remove a set of hosts from the network and route traffic, while the administrator works on removal of the worm. Mitigation involves lessening the severity of the attack.

a) *Remove from Network:* The ability of an administrator to remove infected hosts preventing further damage. As the example described above, a particular worm may reside in a network and begins propagation.

b) *Whitelisting:* A list of permissible connections that are known to the defender. An attack could be directed at a particular software, which may reside on predetermined port.

c) *Reference Advisement:* Notes provided by the defender to mitigate an attack, or a vulnerability/vendor database reference number used to alleviate a vulnerability or attack.

2) *Remediation:* In the presence or prior to vulnerability exploitation, there are resolution steps that are available to a defender to prevent an attack. Remediation would involve taking the appropriate steps to correct the situation prior to or during an exploitation.

a) *Patch System:* Applying patches the vendor has released due to some vulnerability within software in use. When a vulnerability or attack is present, on various cases, a defender fails to utilize the patches a vendor provides.

b) *Correct Code:* Steps within an organization to release a code patch to a specific application that will close the potential for an attacker to exploit.

B. Classification by Information Impact

An attack on a targeted system has potential to impact sensitive information in various ways. A committed resource must be able defend information warfare strategies in an effort to protect themselves against theft, disruption, distortion, denial of service, or destruction of sensitive information assets [12]. This section classifies an attack's impact or the effect on information and defines the criteria used.

1) *Distort:* A distortion in information, usually when an attack has caused a modification of a file. When an attack involves distort, it is a change to data within a file, or modification of information from the victim [4].

2) *Disrupt:* A disruption in services, usually from a Denial of Service. When an attack involves disrupt, it is an access change, or removal of access to victim or to information [4].

3) *Destruct:* A destruction of information, usually when an attack has caused a deletion of files or removal of access. Destruct is the most malicious impact, as it involves the file deletion, or removal of information from the victim [4].

4) *Disclosure:* A disclosure of information, usually providing an attacker with a view of information they would normally not have access to. Kjaerland [4] describes disclosure as unauthorized disclosure of information, with the possibility of leading to other compromises.

5) *Discovery:* The discovery of information not previously known. For example, when a scanning tool probes for information, the information discovered can be used to launch an attack on a particular target.

C. Classification by Target

Various attacks target a variety of hosts, leaving the defender unknowingly susceptible to the next attack. The resources are operating system, a network, a local computer, a user's personal information and an application. Targeting one or a combination of these may aim the attack for fruition. This section classifies targets an attack will use to perform unauthorized privileges.

1) *Operating System (Kernel / User / Driver):* Responsible for the coordination of activities and the sharing of resources of a computer. An attack can be formulated to target vulnerabilities within a particular operating system.

2) *Network*: Target a particular network or gain access through a vulnerability within a network or one of the network protocols [6].

3) *Local:* An attack targeting a user's local computer.

4) *User*: An attack against a user is an attack to retrieve a user's personal information.

5) *Application*: An attack towards specific software. An application can be either client or server. A client application is software that is available to aid a user performing common tasks. A server application is software designed to serve as a host to multiple concurrent users.

IV. Taxonomy Classification Process and Example

In this section we highlight the structure and cause, action, defense, analysis, and target (CADAT) process used to classify attacks. We use AVOIDIT with a past computer attack and associated vulnerabilities. This will depict how our cyber attack taxonomy successfully classifies attack vector information and provides the defender with countermeasures that can be efficient in preventing or assuaging successful attacks.

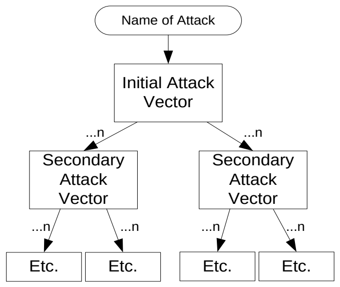


Fig. 2. AVOIDIT – Tree

A. CADAT Process

AVOIDIT benefits from classifying attacks in a tree structure, including the illusive blended attack. Using an attack tree provides a systematic method of characterizing a variety of attacks and enumerates the ways an attack could occur. Predecessors [4], [6] state that providing a tree structure is a solution to solving the blended attack, but claim this particular structure can become unorganized. We provide AVOIDIT in a tree structure to successfully classify common vulnerabilities and cyber-attacks to provide defenders with information required for defense. Each path on the attack tree represents a unique attack. Fig. 2 provides insight into how a tree structure can be converted into a searchable schema to classify a multitude of attacks. By using a parent-child relationship, AVOIDIT is able to display how multi-staged attacks can be captured, classified, and disseminated. Fig. 2 provides a small depiction of the initial attack vector being the parent is classified using a tree structure, including the remaining classifiers, which are omitted for space constraints. The secondary and following attack vectors are used as child nodes that comprise of how the attack was successful in its exploitation. Each node contains a unique id that helps identify the parent and child attack vectors. This feature aids in classifying blended attacks.

Once an attack is recognized, AVOIDIT uses the cause, action, defense, analysis, and target (CADAT) process to classify attacks against a particular target. The CADAT process consists of the following using AVOIDIT:

1) *Classify the Attack Vector:* Understand the cause in which the attack came to fruition.

2) *Classify the Operational Impact:* The type of action conducted resulting from the impact the attack vector enabled to take place (i.e. Trojan horse).

3) *Classify the Potential Defense:* Understand how to properly defend using preventative and/or reactive methods to a potential attack.

4) *Classify the Informational Impact:* Providing an analysis for reporting purposes to what damages have or may take place once the attack is successful.

5) *Classify the Target to which the Defense is Sought:* Properly identifying the target in which the attack seeks to compromise.

Although, the CADAT process is not final, due to different roles having various levels of understanding of an attack, it is a good start for research. For instance, support personnel may log the operational impact first, prior to involving the network administrator. The process for classifying attacks using CADAT enables the entire organization to gain common security knowledge to become vigilant in defending against attacks.

B. Taxonomy Example with SQL Slammer

SQL Slammer worm was able to perform 55 million scans per second and compromised ninety percent of vulnerable hosts in 10 minutes [3]. Continuing from our example in section 1, if the SQL slammer worm was activated on a network, when the ticket is opened and routed to the network administrator, notice of a considerable traffic amount located on UDP port 1434 is determined. The network administrator will immediately perform a scrupulous whitelist against IP addresses targeting port 1434. Table I classifies the SQL Slammer worm using AVOIDIT.

TABLE I. SLAMMER ATTACK CLASSIFICATION

Lough (Name)	**Improper Validation**	**Improper Exposure**	**Improper Randomness**	**Improper Deallocation**
Slammer	X	X		

Howard (Name)	**Tools**	**Vulnerability**	**Action**	**Target**	**Unauthorized Result**
Slammer	Script	Configuration, Design	Prob, Modify	Network	Corruption of Information

Hansman (Name)	**1st Dimension**	**2nd Dimension**	**3rd Dimension**	**4th Dimension**
Slammer	Network-Aware Worm	MS SQL Server 2000	CAN-2002-0649	Stack Buffer Overflow & UDP packet flooding DoS

AVOIDIT (Name)	**AV**	**OI**	**D**	**I**	**T**
Slammer	Misconfiguration	Installed Malware: Worm: Network Aware	Mitigation: Whitelist CVE-2002-0649	Discover	Network
Slammer	Buffer Overflow	Installed Malware: Worm: Network Aware	Remediation: Patch System	Distort	Application
Slammer	Denial of Service	Installed Malware: Worm: Network Aware	Remediation: Patch System	Disrupt	Application

The Slammer worm contained multiple vulnerabilities that provided the ability for a successful attack. In Table I, Lough's taxonomy is too general to provide useful information in describing the attack; Howard's taxonomy provides preliminary information. Hansman and Hunt's taxonomy is able capture more detail in comparison to Howard. However, AVOIDIT provides information on what caused the worm infection and possible defense strategies a network administrator can use to reduce the propagation of malware to cause further damage. Using AVOIDIT, if the first insertion was alleviated, the Slammer worm would not be able to spread.

C. Microsoft RPC Stack Overflow

In 2008, a Windows Server service Remote Procedure Call (RPC) stack buffer overflow vulnerability [14] was exploited and is currently "in the wild". This RPC service provides print support and network pipe sharing were other users were able to access services over a network. The notable Conficker or Downadup attacks use these vulnerabilities to perform attacks on vulnerable systems. Table II classifies the RPC buffer overflow and its subsequent attacks.

TABLE II. MICROSOFT RPC STACK OVERFLOW CLASSIFICATION

Lough (Name)	Improper Validation	Improper Exposure	Improper Randomness	Improper Deallocation
MS RPC Stack Overflow	X	X		

Howard (Name)	Tools	Vulnerability	Action	Target	Unauthorized Result
MS RPC Stack Overflow	Script	Design	Modify	Process	Increased Access

Hansman (Name)	1st Dimension	2nd Dimension	3rd Dimension	4th Dimension
MS RPC Stack Overflow	Stack Buffer Overflow	Windows Server	CVE-2008-4250	Corruption of Information

AVOIDIT (Name)	AV	OI	D	I	T
MS RPC Stack Overflow	Buffer Overflow : Stack	Installed Malware : ACE	Distort	Mitigation : RA VU#827267 Remediation: Patch System	OS: Windows Server
Gimmiv.A	Buffer Overflow : Stack	Installed Malware : Trojan	Disclosure	Mitigation : RA	OS: Windows: Srvr
Conficker	Buffer Overflow : Stack	Installed Malware : Worm	Disrupt	Mitigation : RA	OS: Windows: Srvr, 2000, XP

Lough or Howard's taxonomy was unable to fully classify the details of the buffer overflow, thus lacks the ability to aid in defending against the vulnerability exploit. Hansman and Hunt's taxonomy was able to classify the attack, but insufficient regarding the possible blended attack or subsequent vulnerability the various attacks exploited. With this particular vulnerability exploitation, you can view AVOIDIT as being able to thoroughly classify the vulnerability, potential blended attacks, and attack variations that specifically exploited the Windows buffer overflow vulnerability.

V. AVOIDIT LIMITATIONS

Attacks have become increasingly present in the cyber world, and being able to prevent all attacks is extremely difficult. In this section we highlight two distinct limitations of AVOIDIT. Other limitations are available for discussion, but the authors chose to omit considering the below has an immediate impact on the proposed taxonomy.

D. Lack of Defense Strategies

The defense strategies in AVOIDIT present a defender with an appropriate starting point to mitigate and/or remediate an attack. The plausible defenses are enormous, so the proposed taxonomy provides a high level approach to cyber defense. Although AVOIDIT is extensible, more research is needed to provide an exhaustive list of possible defense strategies for each attack vector exploited. This can be accomplished using a knowledge base to track attack vectors and respective defenses. Additional discussion is provided in the future works section.

E. Physical Attack Omission

Physical attacks are an important aspect in achieving security. While it is necessary to understand physical attacks, the AVOIDIT IRS focuses on cyber-attacks. Further research can be done to include the physical aspect of cyber security, which may include the end hosts of an attack. For example, cyber attacks that begin with use of a USB drive are not present in the current state of the AVOIDIT Taxonomy, but can be aims for future endeavors

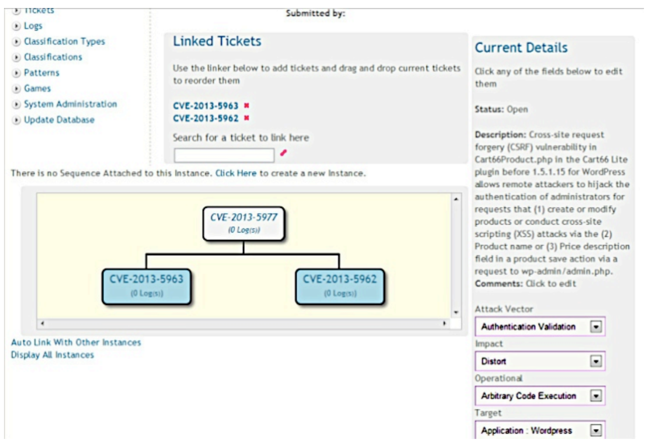
VI. CONCLUSION

This paper provides a cyber attack taxonomy AVOIDIT that classifies attack components by attack vectors, operational impact, defense, informational impact, and target. This classification scheme will aid a defender in protecting their network by disseminating vital attack information to advise and construct defense strategies. Most of the research involved in developing an attack taxonomy only offer information for individuals knowledgeable in security. A significant defense involves organizations aptitude for all to take precautions against cyber threats. Previous taxonomies lack the structure of useful information to classify attacks through vectors that can be utilized in a local application setting. Former taxonomies are absent of pertinent details practical for knowledge bodies, such as CERT. Hansman et al. [6] mentioned the need to improve classifying blended attacks, which is also a limitation of prior taxonomies. Using AVOIDIT, we have demonstrated a straightforward CADAT process to pinpoint the classification of vulnerabilities, attacks, and security threats to benefit cyber security awareness in an organization. We provided attack examples presented in a tree structure to neatly classify attack vectors used to launch cyber-attacks. Using this structure we encapsulate classifying blended attacks. This paper aims to increase an organization's attack resiliency in all functional areas, as opposed to specific individuals.

We are aware of the possibility of new attack manifestations; therefore AVOIDIT is extensible to include new categories within each classification. AVOIDIT will provide a defender with the appropriate information to make an educated decision in defending against cyber-attacks. Creative approaches to defending attacks will become available and providing an extensible taxonomy able to classify new defenses associated to an attack is imperative for defense. We believe AVOIDIT provides a foundation for the cyber security community and the capability to continuously grow as attacks and defenses become more sophisticated.

VII. Future Work

This paper is proposed to continuously evolve the dynamics of taxonomies with respect to cyber security. Further development is ongoing in the development of our issue resolution system, where an AVOIDIT schema will be used to enumerate and classify attack vector discovery for optimal defense. Fig. 3 depicts our proposed IRS where a user is able to view CVE related incidents and associated log instances, as well as attack vector information and the attack tree of the potential attack.



Use the linker below to add tickets and drag and drop current tickets to reorder them

CVE-2013-5963
CVE-2013-5962

Search for a ticket to link here

There is no Sequence Attached to this Instance. Click Here to create a new Instance.

Click any of the fields below to edit them

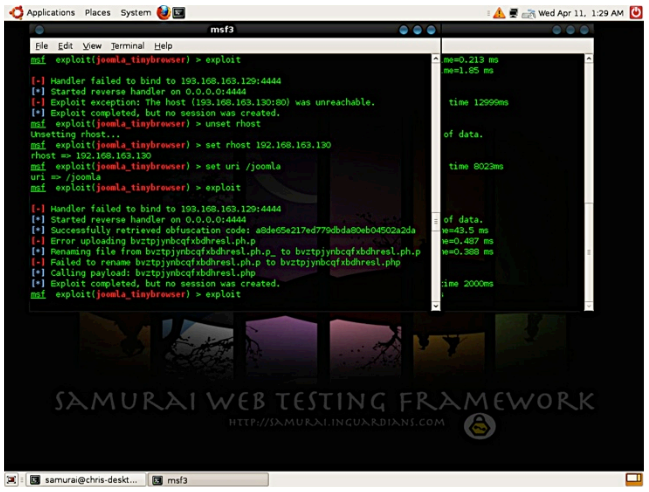
Status: Open

Description: Cross-site request forgery (CSRF) vulnerability in Cart66Product.php in the Cart66 Lite plugin before 1.5.1.15 for WordPress allows remote attackers to hijack the authentication of administrators for requests that (1) create or modify products or conduct cross-site scripting (XSS) attacks via the (2) Product name or (3) Price description field in a product save action via a request to wp-admin/admin.php.
Comments: Click to edit

Attack Vector: Authentication Validation
Impact: Distort
Operational: Arbitrary Code Execution
Target: Application : Wordpress

Fig. 3. IRS user interface

AVOIDIT applied within the IRS provides a common way to represent security knowledge for an entire organization to benefit in all disciplines. Fig. 4 highlights a classified CVE report using AVOIDIT (depicted on the right) retrieved by the IRS involving a WordPress application, which was deemed a monitored asset by the organization. The IRS has the ability to link tickets representative of attack vectors in a tree structure. This enables the IRS to display to the defender attack alternatives and complete paths of an attack.

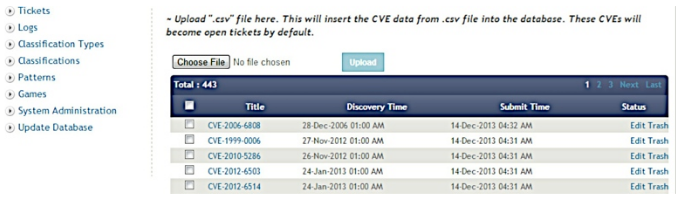


```
msf  exploit(joomla_tinybrowser) > exploit

[-] Handler failed to bind to 193.168.163.129:4444
[*] Started reverse handler on 0.0.0.0:4444
[-] Exploit exception: The host (193.168.163.130:80) was unreachable.
[*] Exploit completed, but no session was created.
msf  exploit(joomla_tinybrowser) > unset rhost
Unsetting rhost...
msf  exploit(joomla_tinybrowser) > set rhost 192.168.163.130
rhost => 192.168.163.130
msf  exploit(joomla_tinybrowser) > set uri /joomla
uri => /joomla
msf  exploit(joomla_tinybrowser) > exploit

[-] Handler failed to bind to 193.168.163.129:4444
[*] Started reverse handler on 0.0.0.0:4444
[*] Successfully retrieved obfuscation code: a8de65e217ed779dbda80eb04502a2da
[-] Error uploading bvztpjynbcqfxbdhresl.ph.p
[*] Renaming file from bvztpjynbcqfxbdhresl.ph.p_ to bvztpjynbcqfxbdhresl.ph.p
[-] Failed to rename bvztpjynbcqfxbdhresl.ph.p to bvztpjynbcqfxbdhresl.php
[*] Calling payload: bvztpjynbcqfxbdhresl.php
[*] Exploit completed, but no session was created.
msf  exploit(joomla_tinybrowser) > exploit
```

```
me=0.213 ms
me=1.85 ms

 time 12999ms

of data.

 time 8023ms

of data.
e=43.5 ms
e=0.487 ms
e=0.388 ms

ime 2000ms
```

Fig. 4. Virtual machine test

More specifically, the IRS takes the system's behavior graph and real time sensor readings as inputs. It gives a probability distribution over the next possible states as output. Computationally, this leads to identifying an underlying NFA (Non-deterministic Finite Automaton) in the IRS. The Automaton operates on the system similar to the probability attack models in [17]. The state space includes some states, which correspond to a secure status of the system while others correspond to the system's compromised status. With these states being vertices & possible transitions between the states being the edges this state space corresponds to the system's graph. The transitions of the system between the sub-graphs with only secure states into that with compromised states will be our prime interest, whose formal structure we intend to investigate in future work. Fig. 4 gives an example of our tree-structure highlighted in section 4, which will be converted into attack graphs as described herein. Focus will be placed on web application related vulnerabilities and attacks.



~ Upload ".csv" file here. This will insert the CVE data from .csv file into the database. These CVEs will become open tickets by default.

Total : 443

	Title	Discovery Time	Submit Time	Status
☐	CVE-2006-6808	28-Dec-2006 01:00 AM	14-Dec-2013 04:32 AM	Edit Trash
☐	CVE-1999-0006	27-Nov-2012 01:00 AM	14-Dec-2013 04:31 AM	Edit Trash
☐	CVE-2010-5286	26-Nov-2012 01:00 AM	14-Dec-2013 04:31 AM	Edit Trash
☐	CVE-2012-6503	24-Jan-2013 01:00 AM	14-Dec-2013 04:31 AM	Edit Trash
☐	CVE-2012-6514	24-Jan-2013 01:00 AM	14-Dec-2013 04:31 AM	Edit Trash

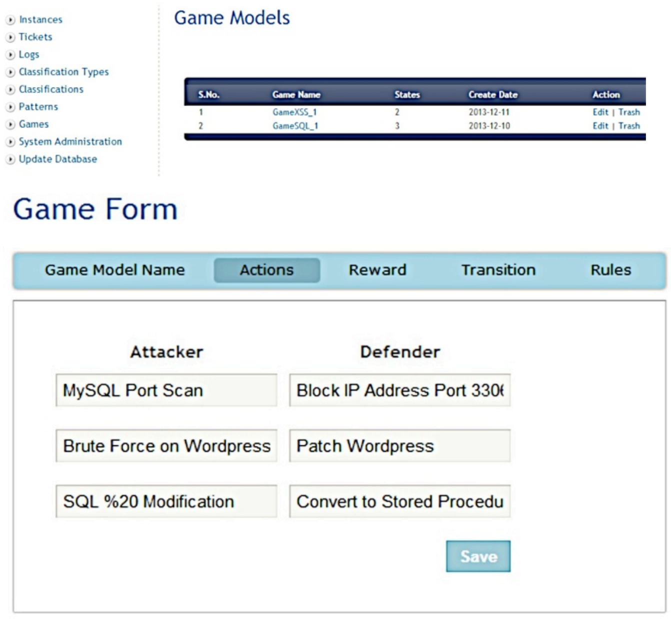
Fig. 5. IRS user interface

We set up the IRS in a virtual attack test environment as highlighted in Fig. 5. The test environment consists of the Ubuntu based UltimateLAMP VMWare image containing multiple vulnerable web applications, such as Joomla, Drupal, and WordPress. The test environment will also consist of the Metasploitable Ubuntu 8.04 server install on a VMWare 6.5 image, which mainly consists of networking vulnerabilities. Fig. 4 depicts an attack against a Joomla web application. We use the Metasploit joomla_tinybrowser module to conduct the attack and exploit a vulnerability in Joomla 1.5.12 tinybrowser plugin. This particular attack is used to perform a file upload and execute arbitrary code on the targeted system. This

Metasploit module connects to the target server and allows the attacker to upload a file without logging in. Once successfully connected to the server an attacker modifies the file name to prevent detection with the option to conduct further damage.

We intend to use the IRS within this test environment, as illustrated, to assist with classifying pertinent discovery information towards defense notification prior to a full attack. We will conduct multiple attacks using the Metasploit attack modules against various web applications. Conducting this task will allow the assessment of the IRS. We also use Selenium [20], an automation tool, to create automated scripts that allow for persistent session management.

In a preliminary study of the IRS's capability to classify vulnerability information, 500 CVEs were used to test the IRS. The IRS successfully retrieved 443 specific CVEs associated to WordPress. Of the 443, 414 were correctly classified using the knowledge levied by security experts. This provides a recall, or true positive rate, of ninety-three percent (83%), and a precision rate of (93%). This is reflective of a system, which provides the administrator pertinent information associated to the systems of interest. Fig. 5 highlights the associated CVEs retrieved by the IRS relative to the application installed. Future work is in progress to correlate application log instances from our test environment with CVE information to afford a system administrator the capacity of quickly mitigating problematic instances in a system, and prevent subsequent attack vectors.



S.No.	Game Name	States	Create Date	Action
1	GameXSS_1	2	2013-12-11	Edit | Trash
2	GameSQL_1	3	2013-12-10	Edit | Trash

Attacker	Defender
MySQL Port Scan	Block IP Address Port 330(
Brute Force on Wordpress	Patch Wordpress
SQL %20 Modification	Convert to Stored Procedu

Fig. 6. SQL injection game

Further, we will use the issue resolution system within a Game Inspired Defense Architecture (GIDA) to investigate the applicability of the AVOIDIT schema classifying attack vector information towards determining the action space of the attacker. The IRS will determine the type of attack. GIDA will use the information to assess candidate game models identified by the IRS to select the optimal game model for defense [19]. Fig. 6 presents an example of game models used within the IRS.

Currently, the IRS stores the actions for both attacker and defender along with associated probability and rewards. Fig. 6 highlights the SQL Injection candidate game information, which will be used by GIDA to assess the optimal game model for a SQL Injection Attack.

REFERENCES

[1] S. William, "A Taxonomy of Computer Program Security Flaws, with Examples". ACM Computing Surveys, 26,3 (Sept. 1994).

[2] S. Noel, S. Jodie, B. Cowberry, M. Jacobs, "Efficient Minimum-Cost Network Hardening via Exploit Dependency Graphs". in Proceedings of the 19th Annual Computer Security Applications Conference, Las Vegas, Nevada, December 2003.

[3] D. Moore, V. Paxson, S. Savage, C. Shannon, S. Staniford, and N. Weaver, "Inside the slammer worm". In IEEE Security and Privacy, volume 1, 2003.

[4] M. Kjaerland, "A taxonomy and comparison of computer security incidents from the commercial and government sectors". Computers and Security, 25:522–538, October 2005.

[5] K. Scarfone, M. Souppaya, et al., "Technical Guide to Information Security Testing and Assessment". NIST (Sept. 2008) http://web.nvd.nist.gov/view/vuln/detail?execution=e7s1

[6] S. Hansman and R. Hunt, "A taxonomy of network and computer attacks". Computer and Security (2005).

[7] S. Amer and J. Hamilton, "Intrusion Detection Systems (IDS) Taxonomy – A Short Review". Defense Cyber Security, 13 (2), June 2010.

[8] J. D. Howard and T. A. Longstaff, "A Common Language for Computer Security Incidents". Technical report, Sandia National Laboratories, 1998.

[9] J. King, K. Lakkaraju, and A. Slagell, "A taxonomy and adversarial model for attacks against network log anonymization". In ACM symposium on Applied Computing (SAC), 2009.

[10] J. Mirkovic and P. Reiher, "A Taxonomy of DDoS Attack and DDoS Defense Mechanisms". In ACM CCR (April 2004).

[11] C. Cowan, P. Wagle, C. Pu, S. Beattie, J. Walpole, "Buffer Overflows: Attacks and Defenses for the Vulnerability of the Decade". Foundations of Intrusion Tolerant Systems (OASIS'03), pp. 227, 2003.

[12] B. Cronin and H. Crawford, "Information warfare: Its Application in military and civilian contexts". Information Society, volume 15, pp. 257-263, 1999.

[13] C. Douligeris and A. Mitrokotsa, "DDoS Attacks and Defense Mechanisms: Classification and State-of-the-art". Comp. Networks, vol. 44, pp. 643–66, 2004.

[14] P. Porras, H. Saidi, and V. Yegneswara, "An Analysis of Conficker's Logic and Rendezvous Points". Malware Threat Center. SRI International Technical Report, February 2009.

[15] C. Meyers, S. Powers, and D. Faissol, "Taxonomies of cyber adversaries and attacks: a survey of incidents and approaches," Technical Report, Lawrence Livermore National Laboratory, 2009.

[16] J. Williams and D. Wichers. OWASP Top 10 - 2010. https://www.owasp.org/index.php/Top_10_2010-Main, 2010.

[17] V. Mehta, C. Bartzis, H. Zhu, E. Clarke, and J. Wing, "Ranking attack graphs," In Recent Advances in Intrusion Detection, 2006.

[18] The Global Information Risk Report. available at www.weforum.org/reports, Retrieved January 5, 2013.

[19] S. Shiva, H. Bedi, C. Simmons, and V. Shandilya, "A Game Inspired Defense Architecture," Poster presented at the Third Conference on Decision and Game Theory for Security, Budapest, Hungary, 2012.

[20] SeleniumHQ Browser Automation Tool, Retrieved June 24, 2013 http://docs.seleniumhq.org/.