11th Annual Symposium on Information Assurance (ASIA ’16)

General ASIA Chair:
Sanjay Goel
Information Technology Management, School of Business
University at Albany, State University of New York

Academic Track of 19th Annual NYS Cyber Security Conference
Empire State Plaza Albany, NY, USA
June 8-9, 2016
11th Annual Symposium on Information Assurance (ASIA ’16)

General ASIA Chair:
Sanjay Goel
Information Technology Management, School of Business
University at Albany, State University of New York

Academic Track of 19th Annual NYS Cyber Security Conference
Empire State Plaza Albany, NY, USA
June 8-9, 2016
MESSAGE FROM ASIA GENERAL CHAIR

Welcome to the 11th Annual Symposium on Information Assurance (ASIA’16)! This event complements the NYS Cyber Security Conference as its academic track with a goal of increasing interaction among practitioners and researchers to foster infusion of academic research into practice. For the last several years, ASIA has been a great success with excellent papers and participation from academia, industry, and government and well-attended sessions. We have excellent papers and very accomplished speakers in the program.

I would like to thank the talented technical program committee that has supported the review process for ASIA. In most cases, the papers were assigned to at least three reviewers who were either members of the program committee and experts in the field. We ensured that there was no conflict of interest during the review and selection process. Our goal is to keep the quality of submissions high as the symposium matures. There were multiple levels of quality control – first with the reviewers, then with the program chair, and then with the general chair. The committee serves a critical role in the success of the symposium and we are thankful for the participation of each member of the committee.

I would like to thank our Program Chair, Jaideep Vaidya from Rutgers University and our Review Chair, Justin Giboney from UAlbany in helping to manage the review process and setting up an excellent program for ASIA. I would also like to thank our proceedings chairs, Ersin Dincelli and Nic DePaula from UAlbany, who have worked diligently on editorial review of the articles to ensure a high quality standard for our proceedings. Damira Pon the publicity chair has done an excellent job in promoting the conference and keeping the website updated. Finally, we were fortunate to have extremely dedicated partners in New York State Information Technology Services. Deborah Snyder, our organization chair, represents NYITS on our board and has been instrumental in planning the conference.

NYS Office of Information Technology Services, the NYS Forum, and our partners in the event who have managed the logistics for the conference, allowing us to focus on the program management. We would like to thank the conference terabyte sponsor AT&T, the conference megabyte sponsor CISCO and Hewlett Packard Enterprise, the conference kilobyte sponsors ePlus Technologies, Presidio, Dell Security, Symantec, and the symposium dinner sponsor – the University at Albany’s School of Business for providing financial support for the symposium.

Most importantly, I would like to thank the authors for taking the time to prepare high quality manuscripts and respecting the deadlines imposed on them allowing us sufficient time for reviews and publication.

I hope that you enjoy the symposium and continue to participate in the future. In each of the subsequent years, we plan to have different themes in security-related areas. We also plan to hold the symposium next year. If you would like to propose a track or partner on a future symposium, please let us know. The call for papers for next year’s symposium will be distributed in the fall and we hope to see you again in 2017.

Sanjay Goel, ASIA General Chair
Director of FACETS
Professor & Chair, Information Technology Management Department, School of Business
# 11th Annual Symposium on Information Assurance (ASIA '16)

## DAY 1: Wednesday, June 8, 2016 (8:00am – 5:00pm)

### Registration & Visit Exhibitors – Base of the Egg (8:00 – 9:00am)

### Morning Session & Keynote (9:00 – 10:30am)

**Welcome Address:** Matthew Millea, *Deputy Director of State Operations, Office of the Governor*

**ASIA Welcome Remarks:** President Robert Jones, *University at Albany*

**Keynote:** "Focusing the Cyber Security Lens" Tim Brown, *Dell Fellow and CTO of Dell Security*

### Break / Visit Exhibitors – Base of the Egg (10:30 – 11:00am)

### Symposium Session 1: Information Security (11:00 – 11:50am)

**Chair:** Merrill Warkentin, *Mississippi State University, MS*

**Paper:** The Theory of Successful Behavior: A Theoretical Explanation of Successful Novice Cyber Security Practitioners

Justin S. Giboney and Adem Kotil, *University at Albany, SUNY, NY*

**Paper:** Countermeasures against Distributed Denial of Service – A literature Review

Manish Gupta, Gayathri Gopalakrishnan and Raj Sharman, *University at Buffalo, NY*

### Lunch / Visit Exhibitors – Base of the Egg (11:50am – 1:00pm)

### Symposium Session 2: Cyber Physical System Security & Cryptography (1:00 – 1:50pm)

**Chair:** Yuan Hong, *University at Albany, SUNY, NY*

**Paper:** Industrial Control Systems Security Testbed

Emrah Korkmaz, *Binghamton University and Turkish Military Academy*, Andrey Dolgikh, Matthew Davis, and Victor Skormin, *Binghamton University*

**Invited Paper:** FrankenSSL: Recombining Cryptographic Libraries for Software Diversity

Bheesham Persaud, Borke Obada-Obieh, Nilofar Mansourzadeh, Ashley Moni, and Anil Somayaji, *Carleton Uni.*

### Break / Visit Exhibitors – Base of the Egg (1:50 – 2:10pm)

### Symposium Session 3: Cyber Threats and Incident Analysis (2:10 – 3:00pm)

**Chair:** Damira Pon, *University at Albany, SUNY, NY*

**Paper:** A Sophistication Index for Evaluating Data Breaches

Nic DePaula and Sanjay Goel, *University at Albany, SUNY, NY*

**Invited Talk:** Information Sharing to Manage Cyber Incidents: Deluged with Data, yet Starved for Timely Critical Data


### Break / Visit Exhibitors – Base of the Egg (3:00 – 3:20pm)

### Symposium Session 4: Security Education (3:20 – 4:15pm)

**Chair:** Justin Giboney, *University at Albany, SUNY, NY*

**Paper:** Network Defense Exercises with Simulated Attacks

Delbert Hart, *SUNY Plattsburgh*

---

**END OF DAY 1**
DAY 2: Wednesday, June 9, 2016 (8:00am – 4:30pm)

REGISTRATION & VISIT EXHIBITORS – Base of Egg (8:00 – 9:30am)

MORNING SESSION & KEYNOTE (8:30 – 10:00am)
State of Cyber Security: Sanjay Goel, University at Albany, SUNY, NY
Hacking Demonstration: "How insecure are we: Look through the eyes of Darth Vader" UA Albany and NYSITS Hacking Team
Keynote: "Reducing Cyber Risk Through Diplomacy" Michele Markoff, U.S. Department of State
Plenary: "Lessons for a More Defensible Cyberspace" Jason Healey, Columbia University

BREAK / VISIT THE EXHIBITORS – Base of the Egg (10:00 – 10:30am)

SYMPOSIUM SESSION 5: Terrorism (10:30 – 11:20am)
Chair: Sanjay Goel, University at Albany, SUNY, NY

Paper: Towards Cognitive Immunization of Potential Criminals against Cyberterrorism
Merrill Warkentin, Mississippi State University, Joshua M Regan, University of New Haven, and Amin G Kosseim, New York City Police Department

Paper: Ranking Terrorist Nodes of 26/11 Mumbai Attack using Analytical Hierarchy Process with Social Network Analysis
Pankaj Choudhary and Upasna Singh, Defence Institute of Advanced Technology, Pune, India

BREAK / VISIT THE EXHIBITORS – Base of the Egg (11:20 – 11:40am)

SYMPOSIUM SESSION 6: Multimedia Forensics (11:40 – 12:30pm)
Chair: Fabio Auffant, University at Albany, SUNY, NY

Paper: Linguistic Features of Phone Scams: A Qualitative Survey
Judith L. Tabron, Hofstra University, NY

Paper: Facebook Forensics for Windows 10
Pankaj Choudhary, Upasna Singh, Nitesh Bharadwaj, and Bhupendra Singh, Defence Institute of Advanced Technology, Pune, India

LUNCH / VISIT THE EXHIBITORS – Base of the Egg (12:30 – 1:40pm)

SYMPOSIUM SESSION 7: Web Security (1:40 – 2:30pm)
Chair: Pradeep Atrey, University at Albany, SUNY, NY

Paper: Covert Channel in HTTP User-Agents
Susan Heilman, Jonathan Williams, and Daryl Johnson, Rochester Institute of Technology

Paper: Don’t See me, Just Edit Me: Towards Secure Cloud-based Video Editing
Odd-Arild Kristensen, University at Albany, SUNY, NY; Manoranjan Mhanty, New York University Abu Dhabi, and Pradeep Atrey, University at Albany, SUNY, NY

NETWORKING BREAK (2:30 – 2:50pm)

SYMPOSIUM SESSION 8: Security Legislation (2:50 – 3:45pm)
Chair: Victoria Kisekka, University at Albany, SUNY, NY

Paper: Cybersecurity Benefits of Executive Order 13,694
William Augustine, University at Albany, SUNY, NY

CLOSING REMARKS (3:45 – 4:00pm)
Sanjay Goel, General Chair
# TABLE OF CONTENTS

Day 1 Keynote  
**Focusing the Cyber Security Lens** .................................................................................................................. 1  
Tim Brown, *Dell Fellow and CTO of Dell Security*  

Session 1: Information Security  
**The Theory of Successful Behavior: A Theoretical Explanation of Successful Novice Cyber Security Practitioners** ............................................................................................................................... 2  
Justin S. Giboney and Adem Kotil, *University at Albany, SUNY, NY*  

**Countermeasures against Distributed Denial of Service – A literature Review** ................. 6  
Manish Gupta, Gayathri Gopalakrishnan, and Raj Sharman, *University at Buffalo, NY*  

Session 2: Cyber Physical System Security & Cryptography  
**Industrial Control Systems Security Testbed** ....................................................................................................... 13  
Emrah Korkmaz, *Binghamton University and Turkish Military Academy*, Andrey Dolgikh, Matthew Davis, and Victor Skormin, *Binghamton University*  

**FrankenSSL: Recombining Cryptographic Libraries for Software Diversity** ..................... 19  
Bheesham Persaud, Borke Obada-Obieh, Nilofar Mansourzadeh, Ashley Moni, and Anil Somayaji, *Carleton University, Canada*  

Session 3: Cyber Threats and Incident Analysis  
**A Sophistication Index for Evaluating Data Breaches** .......................................................................................... 26  
Nic DePaula and Sanjay Goel, *University at Albany, SUNY, NY*  

**Information Sharing to Manage Cyber Incidents: Deluged with Data, yet Starved for Timely Critical Data** .............................................................................................................................. 32  

Session 4: Security Education  
**Network Defense Exercises with Simulated Attacks** ....................................................................................... 33  
Delbert Hart, *SUNY Plattsburgh*
## TABLE OF CONTENTS, CONT’D.

Day 2 Keynote  
**Reducing Cyber Risk Through Diplomacy** ................................................................. 40  
Michele Markoff, *U.S. Department of State*

Lessons for a More Defensible Cyberspace ................................................................. 41  
Jason Healey, *Columbia University*

Session 5: Terrorism  
**Towards Cognitive Immunization of Potential Criminals against Cyberterrorism** .......... 42  
Merrill Warkentin, *Mississippi State University*, Joshua M. Regan, *University of New Haven*, and Amin G. Kosseim, *New York City Police Department*

**Ranking Terrorist Nodes of 26/11 Mumbai Attack using Analytical Hierarchy Process with Social Network Analysis** ................................................................. 46  
Pankaj Choudhary and Upasna Singh, *Defence Institute of Advanced Technology, Pune, India*

Session 6: Multimedia Forensics  
**Linguistic Features of Phone Scams: A Qualitative Survey** ............................................. 52  
Judith L. Tabron, *Hofstra University, NY*

**Facebook Forensics for Windows 10** .................................................................................. 59  
Pankaj Choudhary, Upasna Singh, Nitesh Bharadwaj, and Bhupendra Singh, *Defence Institute of Advanced Technology, Pune, India*

Session 7: Web Security  
**Covert Channel in HTTP User-Agents** ............................................................................... 68  
Susan Heilman, Jonathan Williams, and Daryl Johnson, *Rochester Institute of Technology*

**Don’t See me, Just Edit Me: Towards Secure Cloud-based Video Editing** ....................... 74  

Session 8: Security Legislation  
**Cybersecurity Benefits of Executive Order 13,694** ......................................................... 79  
William Augustine, *University at Albany, SUNY, NY*

Author Biographies ................................................................................................................. 84

Index of Authors .................................................................................................................... 90
Focusing the Cyber Security Lens

Tim Brown
Dell Fellow and CTO of Dell Security

2016 finds us in the middle of a perfect storm. Technology is advancing at an incredible rate, the adversary is adapting and adjusting their business models and the cyber security professional is stretched to the limit. This presentation will focus on how to adapt to today’s challenges and say yes to embracing new technologies and new opportunities. It will provide actionable information and knowledge enabling you to focus on the most important assets, prioritize security efforts and prepare for the future.

BIOGRAPHY

Tim Brown is at the front line of the most vexing challenge facing organizations today: IT security. As a former CTO, chief architect, distinguished engineer and director of security strategy, Tim deeply understands the challenges and aspirations of the person responsible for driving digital innovation and change. Tim has over 20 years of development experience in security technology, including identity and access management, security compliance, threat research, vulnerability management, encryption, managed security services and cloud security.

As a Dell Fellow and the CTO of Dell Security, Tim is developing the future of IT security, and it’s a future that is addressing customers' biggest challenges using a proactive, rather than a reactive, approach. His strong industry knowledge in finance, healthcare, education, manufacturing, retail and government enables Tim to provide relevant and actionable insights.

Nationally, his trusted advisor status has taken him from meeting with members of Congress and the Senate to the Situation Room in the White House. He has been on the board of the Open Identity Exchange and a member of the Trans Global Secure Collaboration Program driving advancements in identity frameworks and working with the US government on security initiatives. He is a member of the Advisory board for Clemson University and holds 20+ patents on security related topics.
The Theory of Successful Behavior: A Theoretical Explanation of Successful Novice Cyber Security Practitioners

Research-in-Progress

Justin Scott Giboney and Adem Kotil
Information Technology Management
University at Albany, SUNY
Albany, New York
jgiboney@albany.edu

I. INTRODUCTION

There are some good theories that explain why people attempt certain activities: for example the Theory of Reasoned Action (TRA) [1], the Technology Acceptance Model (TAM) [2], the Theory of Planned Behavior (TPB) [3], the Unified Theory of Acceptance and Use of Technology (UTAUT) [4]. However, none of those theories explain if the person attempts that activity whether they will engage in successful behavior during their activity. There are some theories that explain successful behavior in limited ways: for example Cognitive Fit Theory (CFT) [5], Task-Technology Fit (TTF) [6], and Delone and McLean IS Success Model [7]. However, these theories only explain a few antecedents of successful behavior. Self-Efficacy Theory (SET) [8]—and its application to computers; Computer Self-Efficacy (CSE) [9]—does predict specific successful computer behaviors by mediating most variables with a self-efficacy construct. However, there are many other variables, such as conceptual expertise of the phenomenon, predicted by other theories, that SET and CSE do not account for. There are also management theories about managing performance, but they do not try to predict successful behavior. There are goal theories and motivational theories. These types of theories do not account for skills. In short, there does not seem to be a comprehensive theory regarding successful behavior. A reason for the lack of scientific theory seems to be that competency is typically viewed as a professional or regulatory concern [10]. However, being able to explain why someone will engage in successful behavior in the future can help researchers better understand the antecedents of successful behavior and will help practitioners know which individuals to hire and what needs to be trained or developed in those individuals.

This paper expounds upon previous theories to construct a theory explaining engagement in successful behavior. We propose to test this theory in the context of novice cyber security professionals engaging in security practices. Thus, this paper adds to current literature by 1) creating an encompassing theory about successful behavior, 2) testing antecedents of successful behavior with cyber security novices, and 3) adding three contextual variables to typical behavioral models: low experience, high pressure, and team dynamics.

II. THEORETICAL DEVELOPMENT

This paper starts by introducing the propositional model of the Theory of Successful Behavior (see Figure 1). This paper will expound upon the logic for each of the paths and constructs in the model starting with the definition of successful behavior and the direct effects to successful behavior.

Fig. 1. Propositional model of the Theory of Successful Behavior

A. Successful Behavior

TRA [1], TAM [2], TPB [3], and UTAUT [4] all focus on a particular behavior or more specifically the intention to perform a particular behavior [11], [12]. However, what this paper tries to predict is not a specific behavior that can be encapsulated and measured as intentional or not (e.g., the use of a particular software) but the engagement in a set of behaviors, often unconsciously, that will cause a person to be successful at a certain task. Therefore, successful behavior is more akin to performance.

Performance is often defined as the speed and accuracy of the individual’s actions during the task [5]. Performance is a function of the task strategies or “methods [and] processes required to perform the task” [5, p. 220] used during the task. It is this definition of task strategies that is most closely aligned with the construct this paper is interested in predicting.
Unlike Vessey’s [5] definition, many tasks (e.g., chess) do not have “required” processes but they often have multiple processes that will lead to the desired outcome. Hence we do not seek to predict a single behavior, nor do we strive to predict the optimal (i.e., least resource intensive) behavior. Rather we seek to predict the engagement in any behavior that is likely to lead to a success outcome. Therefore, this paper defines successful behavior as the engagement in methods or processes that lead to successful performance of a task.

B. Self-Efficacy

Successful performance of a task is largely a function of previous performance [8]. Meaning that people are better able to perform a task the more they have successfully performed the task in the past. This concept has been colloquially stated as “perfect practice makes perfect.” Underlying this claim is the concept that people’s experiences are coded and retained as symbols or models in memory [13] that are retrieved and used to guide further action [8]. People’s mental models are refined through self-corrective adjustments as they perform the task [8]. Through reoccurring activity people lean towards beneficial actions and avoid punishing actions [8]. In a sense, people are recognizing positive and negative patterns from sequences of events as they perform an action many times [8].

People’s belief about their future performance is called efficacy expectations. Efficacy expectations are “convictions that [oneself] can successfully execute the behavior required to produce [an] outcome” [8, p. 193]. Efficiency expectations vary on magnitude, the difficulty level people can handle, generality, the number of contexts in which people can perform, and strength, how strongly people believe their expectations [8]. There are four major sources of efficacy expectations: enactive attainments, vicarious experiences, persuasion, and physiological states [8], [14]. Enactive attainments are past performances of the task that are either successful or unsuccessful. Successes will increase while failures will decrease efficacy expectations. Vicarious experiences are the observation of others succeed or fail through their efforts. Observing other people succeed will increase efficacy expectations due to social comparison and a mentality that if someone else can succeed, I must be able to achieve at least some improvement [8]. Education works off of the principle of observing others. Vicarious experiences are how support groups operate. While weaker than the previous two forms, verbal persuasion can also increase efficacy expectations. Verbal persuasion is the suggestion by others that a person can successfully accomplish a task [8]. Physiological states, specifically emotional arousal, also affect efficacy expectations as people use expected stress and anxiety as a measure of the effort that will have to be exerted to successfully perform the task. The more people expect stress, or are already stressed, the more effort they expect they will have to put into the task and the more likely they are to avoid the task altogether, and therefore they expect that they will less likely succeed [8].

Efficacy expectations influence people’s choices as they will choose activities and settings in which they think they will perform better as well as the amount of effort they will exert during the activity [8], [15]. A great example of this choice is the reluctance of many people to switch between operating systems due to their familiarity with one operating system and their fear of the other. People with high efficacy expectations will also set higher achievement goals and be firmer in their belief that they can accomplish those goals [15].


Magnitude in computer self-efficacy refers to people’s perception of the difficulty level they can obtain [9]. People with higher magnitude of computer self-efficacy believe they can accomplish more difficult tasks than people with low computer self-efficacy [9]. People with lower computer self-efficacy believe that they can only execute simple tasks [9].

Strength in computer self-efficacy refers to people’s confidence that they can perform the computer tasks given to them [9]. People with strong computer self-efficacy believe that they are more likely to succeed at a computer task than individuals with low strength [9]. The main difference between magnitude and strength is magnitude refers to belief that one can do complex computing and strength refers to one have greater confidence in the execution of computing tasks [9].

Generalizability in computer self-efficacy refers to people’s ability to apply a particular activity to other settings and activities [9]. Thus, a person with high computer self-efficacy generalizability can use their knowledge of one computer program or system to different computer programs and systems efficiently, whereas people with low computer self-efficacy generalizability perceive their capacity as limited to one computer program or system [9].

As self-efficacy determines efficacy expectations that determine successful behavior, computer self-efficacy determines successful behavior in computer tasks [16]. The relationship between expectations and successful behavior can occur at the specific activity level and at the general activity level and the two levels increase the successful behavior of the other level [16]. Thus we posit:

P1. Self-Efficacy increases successful behavior.

C. Conceptual Expertise

Conceptual expertise is “understanding [of a phenomenon] resulting from the accumulation of a large body of knowledge” [17, p. 167]. Conceptual expertise develops from the construction of a mental representation of a task that aids in decision-making [18]–[20]. People use these mental representations as guides for a solution path [21], [22]. The more someone experiences or learns about a phenomenon, the more comprehensive and generalizable their mental representations become [17]. The more people understand a given phenomenon, the more likely they are to have mental representations that will guide the person to successful behavior. For example, chess masters have accumulated enough conceptual expertise to recognize more ways to win that novices have. Therefore, we propose:

P2. Conceptual expertise increases successful behavior.
When people have increased conceptual expertise of a phenomenon, they understand whether they have the cognitive, social, or behavioral skills and subskills to perform a task [14]. When people know that they have the skills to perform a task they are more likely to believe they are going to be successful. Therefore, efficacy expectations mediate conceptual expertise [14]. Therefore we propose:

**P3. Conceptual expertise increases self-efficacy.**

### D. Ambition

Reinforcement of beneficial actions also leads to increased motivation to perform the task that in turn increases performance [8]. When people believe they will perform better, often because they have performed better in the past, they are more likely to put effort into performing and thus will realize enhanced performance [8]. When people are able to see the fruits of their labor they have increased enjoyment and pride from accomplishing their goals. These elated feelings encourage reoccurring performance that continues the cycle of self-corrective learning. Therefore, people’s belief in their own effectiveness will affect whether they will choose or avoid certain activities during their performance and how long they are willing to persist to overcome obstacles [8]. People that overcome obstacles will obtain self-corrective mental models whereas those that stop prematurely will retain the same debilitating skills and expectations [8].

**P4. Self-efficacy increases ambition**

**P5. Ambition increases performance**

### III. METHODOLOGY

#### A. Study Context

This article studies successful behavior in the national collegiate cyber defense competition (http://www.nationalccdc.org/). This competition is held annually in the United States of America since 2004. During the competition, teams of mostly undergraduate students (blue teams) attempt to defend themselves from competition hackers (red team). Winners are chosen based on how many services each blue team still has running by the end of the attacks. The competition happens in three rounds: qualifiers, regionals, and nationally. Qualifiers and regionals happen in ten regions (e.g., Mid-West Region). One team from each region goes on to compete nationally.

#### B. Participants and Demographics

Participants will be recruited from all people attempting to qualify for their regional competition. We will communicate to all regional organizers and attempt to obtain the list of competitors or have them forward our emails on to the competitors. In 2015 there were 29 qualifying teams for the Mid-Atlantic Region. Assuming there is at least 10 qualifying teams from each region, and at least 15 teams attempting to qualify, and twelve people per team (twelve is the max roster size, but only eight get to compete), the sample size can be up to 1,800 people. [Demographics will be reported after data collection]

#### C. Measurement

To collect our data, we will send out a survey to all of the participants attempting to qualify for the regional competitions. Our survey will contain measures for all of the constructs in our model except for the dependent variable. The dependent variable will be the level of success that the team achieves (0 = did not qualify, 1 = qualified for regions, 2 = qualified for regionals, 3 = won nationally).

### REFERENCES


Countermeasures against Distributed Denial of Service
A Literature Review
Manish Gupta, Gayathri Gopalakrishnan, and Raj Sharman
School of Management
State University of New York
Buffalo, NY, USA
{ mgupta3, g5, rsharman }@buffalo.edu

Abstract—A distributed denial-of-service (DDoS) attack is carried out by simultaneously by compromised systems against targets causing system and service unavailability. Regardless of industry and size, companies worldwide are increasingly becoming target of DDOS attacks. The sophistication and intensity of these attacks are exponentially rising due to increase in number of compromised systems, unpatched vulnerabilities and increased business impact. The paper reviews more than 200 research articles in the area of DDOS, of which 142 present countermeasures and mitigation against DDOS. The paper develops an ontological framework to classify the proposed mitigation methods under three layers of defense-in-depth — prevent, detect and respond. Research done each of these 3 pillars are further conceptualized based on underlying design and security principles. The paper also proposes alternate classification schemes based on placements of target components, while presenting qualitative analyses on research activities on DDOS.

Keywords—distributed denial of service, botnets, availability, network attack, traffic detection, bandwidth consumption, resource exhaustion

I. INTRODUCTION

An application server can provide many services to the clients. There are multiple users who access the system simultaneously. Regular service is provided when clients can access the service without any interruption. A service can be delayed due to various reasons. One reason could be that the resources of the server are insufficient (server features may not be designed well) or there could be flaws within the system. A denial of service happens when a client is denied of the access to a particular system or denied information or any service from a system. Then the question may arise whether the denial of service can be intentional or unintentional. It may happen in both ways. Imagine a situation when a disaster occurs; the service may be interrupted till the time the backups are restored. This is an example of an external factor which would interrupt the service. The internal factors can be due to reasons mentioned earlier like insufficient care given to details while designing the service architecture.

An intentional attack can be influenced by several factors like monetary gains, competence or it can be politically motivated. An attacker may adopt several ways to interrupt the services. An attacker can either directly attack the source with many requests or utilize the bandwidth of the communication channel. A good detection mechanism will look for balancing the server load at the same time keep a watch on the utilization of resources. A good defense mechanism will try to mitigate the attack as well to make sure the legitimate clients are served. But good security policies can help preventing the attacks most of the time although they cannot be completely avoided.

A distributed denial of service happens when there are multiple points of attack sources. The attacker can form the army of vulnerable systems in network to attack the server. A potential impact is business may find loss of revenue due downtime and a trace back is difficult due to innocent servers used as attackers. There can be attacks where IPs are spoofed as well. Typically, the DDoS attacks can be executed with these major components—an attacker, a proper mechanism, the victim/target server and the innocent intermediate servers. In case of DDoS, the bots or zombies participate in the attack unintentionally. This paper reviews literature on countermeasure suggested to mitigate risks from DDoS attacks and classifies these countermeasures in an ontological framework. A review of 142 papers provided key insights to categorize them using a 3-layered defense in depth model. The paper categorizes each paper into one of the categories. This provides a complete and holistic view of research on DDoS protection. We also compared our work against other review and survey papers on similar topic and found our contribution is unique and novel in terms of coverage and in ontological development. The details of each of the papers have not been included in this research due to lack of space. The contributions of the paper can help researchers and practitioners alike understand how DDoS protection research has evolved over last decade or more to be better informed in their decisions.

II. PRELIMINARIES

1. DDoS Attack

The DDoS attack is planned strategically. DDoS is not an uncommon event for more than a decade. The motivation could be malevolent aiming at the destruction of the entire system. It can be monetary gains or even politically motivated. Attacks against banking industry can be really devastating. The DDoS attacks start with initiating control on systems that can act as command and control servers. The command and control servers can give command to other vulnerable systems identified by scanning process. An army is formed in order to
launch the attack. A specific time is decided in order to launch the attack. This will ensure that all the nodes send packets to the target at the same time duration thus causing congestion at the bandwidth or the server memory level. The attacks may last for hours. The aftermath can be the target can be crashed due to overload and the service provided will be temporarily disrupted. The resilience depends on the timely identification of attack, tracing the attack or blocking the illegitimate traffic caused by the attack.

There is a continuous research process which aims at saving the business owners from negative and worst consequences of these attacks. There are many successful and innovative result oriented methodologies but there are few which may not have succeeded in the real world but can be a strong weapon in the future if they can be molded well.

There are some unique features to the DDoS attacks. They are distributed in nature. The impact is much high compared to an attack from a single source. Within a short span of time they can cause much disruption to the system. Over the years much paper has researched on DDoS attacks as well. But are they covering DoS defense as well? To answer this question, we need to look whether the defense mechanisms take into account the distributed nature of the attacks. Some of the common techniques to detect DDoS attacks are tracing the source, statistical analysis on the flow of traffic and detection of self-similarity in the attack traffic. Many of these apply for DoS as well. Although there can be an overlap, there are few defense mechanisms that can be unique to DDoS like analysis on similarity based on source information as there is only one source of attack in case of DoS.”.

2. State of System

A system can be visualized to have the following states. In healthy state, a system can provide services to its clients without any failure. A proactive measure can help the system retain its normal behavior. Figure 1 shows states of a network system. An attack traffic will cause disturbances in the normal performance of the client. The load is unbalanced. When a system detects an attack and knows how to mitigate, it moves to the defense state. On applying policies and recovering activities, system can move back to healthy state. While an attack happens, the survivability of system depends on its resilience. If the resilience is less, then the system may go to exhausted state. Several activities like clearing memory buffer, etc. would be required to regain a balanced state.

III. LITERATURE REVIEW

The following sources are used to collected the papers
1) ACM Digital Library
2) IEEE Xplore
3) ScienceDirect – Elsevier
4) Google Scholar

The keywords used to search are ‘Distributed Denial of service’, ‘Distributed Denial of service attacks and defense’.

The number of papers have increased over the last decade especially a high peak can be observed from 2011 to 2012 and 2013. This proves the fact that much focus is being given recently to defending against DDoS attacks. The attacks are really common with rented bots being available in the market. The Figure 2 shows this trend. The Figure 3 above shows the number of conference proceedings and journals in each year. The papers collected are broadly classified into defense techniques, Survey papers, DDoS Analysis and DoS.

Fig. 1. States of a Network System

Fig. 2. Total Number of Papers

Fig. 3. Distribution of Papers by Type
The defense techniques include the papers that focus on one or the other kind of mechanisms against DDoS discussed later in the paper. The defense techniques are classified and the papers are categorized accordingly. These papers usually observe a particular aspect of the attack traffic and then use this aspect to detect and track the attacks. Majority of the papers have simulated the techniques described using simulation tools like NS-2 or other experimental set ups. Table 1 shows this distribution. The practical aspects of implementation are not fully discussed in many of these papers. The survey papers have classified the defense techniques. There has been many base of classification. The third section of papers include those papers which cover the impact of DDoS attacks, the possibility of attacks on few architectures, simulation mechanisms for studying and analyzing the defense techniques. These papers also include different types of attacks, evolution of attacks. Few papers that completely focus on Denial of Service are also collected.

IV. Classification Scheme

A good classification scheme helps in analyzing the current trends of the research in the field. The classification scheme presented in this paper takes into account different aspects of the defense technique. The classification is based on an extensive review of collection of DDoS papers. All activities done to protect against DDoS can be broadly classified into three types – based on the action performed – Prevention techniques, Detection techniques and Reaction or responses. Although the mechanisms can be classified, they are a continuous process. An organization needs to do continuous improvement on these areas in order to sustain in the world of threats and attacks. Figure 4 shows the classification that we came up with based on literature review. The following sections discuss each of these categories (boxes), using the same numbering scheme as Figure 4.

![Table 1. Distribution of Papers by Type](image)

<table>
<thead>
<tr>
<th>Type of papers</th>
<th>Number of papers collected</th>
</tr>
</thead>
<tbody>
<tr>
<td>Defense Techniques</td>
<td>142</td>
</tr>
<tr>
<td>Survey papers</td>
<td>16</td>
</tr>
<tr>
<td>Analysis on DDoS</td>
<td>23</td>
</tr>
<tr>
<td>Papers focusing on DoS</td>
<td>20</td>
</tr>
</tbody>
</table>

The practical aspects of implementation are not fully discussed in many of these papers. The survey papers have classified the defense techniques. There has been many base of classification. The third section of papers include those papers which cover the impact of DDoS attacks, the possibility of attacks on few architectures, simulation mechanisms for studying and analyzing the defense techniques. These papers also include different types of attacks, evolution of attacks. Few papers that completely focus on Denial of Service are also collected.

IV. Classification Scheme

A good classification scheme helps in analyzing the current trends of the research in the field. The classification scheme presented in this paper takes into account different aspects of the defense technique. The classification is based on an extensive review of collection of DDoS papers. All activities done to protect against DDoS can be broadly classified into three types – based on the action performed – Prevention techniques, Detection techniques and Reaction or responses. Although the mechanisms can be classified, they are a continuous process. An organization needs to do continuous improvement on these areas in order to sustain in the world of threats and attacks. Figure 4 shows the classification that we came up with based on literature review. The following sections discuss each of these categories (boxes), using the same numbering scheme as Figure 4.

![Fig. 4. Classification of DDoS Countermeasures](image)

1. Prevent

Any proactive measure fall under prevention category. These can include stringent security policies that can even protect the system from intrusions, denial of service or any malware activities. The research in prevention techniques require more attention compared to the detection techniques as the attacks can be stopped earlier without much resource wastage. These types of techniques require immense knowledge of attacks, botnet formation and vulnerability detection in networks.

1.1. Tightened Security

The kind of prevention mechanisms are more into securing own systems or using stringent policies by ISP networks. Some of the key aspects in these papers are focused on security policies by ISPs (internet service providers) or
segmenting the network into domain and apply token systems. The concept is more focused to security in individual systems or network domains. The more secure systems are, the attackers find it difficult to capture vulnerable systems.

1.2. Prevent Botnet Formation
A DDoS attack is launched using bots or zombies. Attackers search for vulnerable systems. In case the systems are vulnerable enough to install a malware then they are used as C&C servers (Command and Control Servers). These C&C servers in turn command other zombies to perform the attack. The papers that focus on preventing botnet formation cover key concepts like preventing systems from being botnets, developing botnet detection strategies, etc. For example, building a tool to detect if a computer is a bot depending on the traffic flow. The key idea is that vulnerability of the system must not be exploited to form an army of bot network.

1.3. Source Verification
In source verification methods, the hosts or routers try to verify the source of the attack. The host ensures that the connection set up is attempted by a human and not a bot. These mechanisms use the packet header information and see if the packet can be trusted. In application level, the users can be sent a text based question or CAPTCHA puzzle to verify their authenticity. These type of mechanisms checks for the authenticity of a packet by checking its source. The packets may follow a certain path or contain unique ID that can be verified.

2. Detect
The alarm of DDoS has to be within shortest time possible and as accurate as it can be. A good separation from the legitimate packets is desired. This ensures that legitimate clients are served during attack.

2.1. Anomaly Based
Anomaly means deviation from a normal behavior. These type of defense mechanisms record the behavior of traffic and perform monitoring in order to detect any anomaly. Anomaly can be either volume based or content based.

2.1.1 Rate Anomaly
Statistical analysis/Computational
Some of the defense mechanisms apply methods like probabilities, entropy variations, etc. They use information like flow rate, source and destination IPs, traffic volume, open connections, time series data to analyze and then detect an attack. The papers usually suggest algorithms that can be implemented in router or victim and more of a comparison from a predicted behavior. These papers cover techniques that use neural networks, chaos theory or fuzzy logic to detect an anomaly. Based on the previous learning (when there is no attack), these mechanisms can be used to detect any deviation from normal behavior. A set of fuzzy rules can be built on data that are captured during normal traffic flow.

2.1.2 Volume Anomaly
A volume anomaly can be detected when there is an unusual volume of traffic in the network or in the servers. The papers that use volume anomaly as detection mechanism look at the packet counts, traffic congestion at edge routers or unusually large number of packets to a particular server. The main idea is that there can be large number of packets or heavy packets (with large packet length) in the traffic during an attack time.

2.1.2.1 Server Load
The papers that cover these type of anomaly detection look at the resource consumption in the server side. During an attack, much memory will be utilized due to large number of incoming packets. The flow may be less but the packet lengths could be high. The papers look in to those factors that can be detected in the server. For example, a regulator can be placed on the server side to check the traffic that are having much resource consumption.

2.1.2.2 Network Congestion
A network congestion can be detected locally or globally. Routers can detect for an anomaly and share the information with other routers. The bandwidth of the communication channel to the victim may be consumed due to the attack traffic. During an attack, areas of high network congestion can be identified based on the traffic flow rates.

2.1.3 Veracity Anomaly
These type of defense mechanisms check the source information of packet and see if they are legitimate. The veracity anomaly based techniques are much closely related to the reaction (traceback) mechanisms.

2.1.4 Other Anomaly
The content anomaly detection mechanisms look for any deviation in the packet contents. These can be the packet behavior or access behavior. A user can access for documents that are not commonly accessed.

2.1.4.1 Access Behavior
An access behavior may be recorded for the clients based on their common access behavior. This may include the way the users access the documents or requests that have heavy packets as replies or request for documents that are not commonly accessed. The papers that cover this type of mechanisms include the papers that compute the connection scores of the clients. The defense mechanisms to detect application layer attacks can be categorized here. An example would be frequency of page visits compared to normal behavior.

2.1.4.2 Incomplete Connections/Probing
Every protocol has a sequence of activities/rules and all the network components need to adhere with those rules. During an attack, it can be noticed that some of the packets or
connection requests do not completely obey these rules. These type of mechanisms look for such activities by the clients and detect DDoS attack. The symmetric behavior of protocols is used as the base to detect any suspicious behavior.

Incomplete connections
The type of mechanisms that check for incomplete connections are usually at the server side. For example, incomplete HTTP requests can be checked. In TCP SYN attack, the zombies do not reply for the SYN ACK acknowledgement (When a server receives a SYN packet, it replies with a SYN ACK and expects ACK packet from the client to complete the connection). Some mechanisms also check if the packets are paired. For example, if for every TCP SYN packet, there is an ACK packet. These methods are usually found in router level.

Probing
The papers under this category check for the legitimacy of packets by probing the delay in network or validating the membership in P2P networks.

2.2 Similarity Based
It can be observed that the attacks can be similar in nature either due to the fact that they are planned to be active at the same time, request for same information or originate from similar source. These type of defense mechanisms look for similarity in traffic features or packet requests.

Clustering
The papers classified under this category utilize the clustering to group the traffic that are similar. The main concept behind this is that the attack traffic can be similar to each other. Some of the similarities could be size of the packets, the interval of the packets.

Correlation
These mechanisms look into similarity but use other correlation techniques like regression analysis to look for any correlation in time series of traffic.

3. React
After detecting an attack, the system need to take necessary actions to mitigate the attack. The reaction can be either traceback or filtering (mitigate the attack traffic) or creating an overlay or a ring or shield.

3.1 Traceback
The traceback mechanisms trace back the source of the path. These mechanisms could be based on the packet marking, static path of the packets or checking path fingerprint. For example, the TTL field (Time to Live) can be used to count the number of hops crossed by the packet. This information can be used to estimate the source of the packet. The papers under this category look into techniques to trace the source of the packets using either identifiers, information that can be added in headers of the packet or without any marking on the header. An example of the latter could be when an attack is detected, routers can collaborate, communicate and share the information about the packets send to victim. Irrespective of the type of method used, main idea is to trace back the source of the attack.

3.1.1 With Marking
The traceback mechanisms that use marking requires changes in the packet header. Marking helps to determine the path of the packet.

3.1.2 Traceback without marking
The traceback mechanisms can also work without any modification in the packet header fields. These are usually based on locally detected traffic rates by routers.

3.2 Filtering/Mitigation/Rate Limiting
Filtering can be done based on thresholds or attack signatures. Filtering can be performed at edge routers or firewalls. Some of the papers look into the concept of rate limiting at the routers. When an attack is detected, the routers can share information about the attack signature. Thus the traffic towards the victim can be filtered. Rate limiting is performed in order to reduce the traffic rate. The mechanisms can use statistical analysis before performing rate limiting. The common idea behind these papers are to limit the traffic flow toward the victim and save it from exhaustion.

3.3 Overlay network
A packet needs to travel through the network domain to reach the server. A secure overlay can be used to detect an illegitimate traffic. Agents can be placed in the network to detect any anomaly behavior. The papers that cover this concept discuss about placing a virtual tunnel in the domain or a collaborative mechanism by routers to look for any attackers. A common pattern would be collaboration rather than an autonomous system or a security policy.

V. ALTERNATE CLASSIFICATION SCHEME
1. Placement of the mechanisms
An alternate way to classify the attacks is based on placement of the defense mechanism. The defense technique can be either work independently or can work collaboratively with other nodes in the network. The attack can be detected earlier when the system is near the source but more accurate detection can happen near the victim as the server.
VI. COMPARISON WITH OTHER SURVEYS

We reviewed literature for other similar undertakings on literature review studies and we found several papers. But they have very different focus and coverage than our study. Next we present some of these studies with discussion on their coverage to express differences with our approach.

(Silva, Silva, Pinto & Raquel, 2013) [1]

This paper is a comprehensive survey of research done on botnets during 1999 - 2011. There were 205 papers reviewed in this study. The paper covers definition of botnet, few botnets, and major components of a botnet, characteristics, lifecycle and architecture of botnets. The paper also covers few of the detection mechanisms classified as honeynet-based, intrusion detection systems. IDSs are classified into signature-based and anomaly-based. Anomaly based can be classified into host-based and network-based (active monitoring or passive monitoring).

(Zargar, Joshi & Tipper 2013) [2]

The authors of this study collected 149 papers from 1994 to 2012. The motivation behind hackers are summarized, then the classification of the attacks (network/transport level and application level), botnet architecture and then types of defense mechanisms are covered. The paper points out the need for a comprehensive distributed and collaborative defense solution and also give an insight of the metrics that can be used to evaluate the defense techniques. Other topics include the cyber insurance policies and implementing a new type of mechanism. Types of attacks - 1) Network/transport level DDoS attacks - a) Flooding attacks b) Protocol exploitation flooding c) Reflection-based flooding d) amplification based flooding 2) Application level - a) Reflection/amplification based flooding b) HTTP flooding-session flooding, request flooding, asymmetric attacks, slow request/response attacks Types of Botnets- IRC-based, Web-based, P2P-based Types of defense mechanisms- source-based, destination-based, network-based and hybrid.

(Kumar & Sharma, 2013) [3]

This paper discusses about different security issues in cloud and the different types of attacks found in the cloud based on 21 papers published between 1999 and 2013. Some of the attacks seen in web can be - SQL Injection attacks, Cross Site Scripting attacks, Man in the Middle Attack, DNS attack, sniffer attacks, issues caused by reused IP addresses, BGP Prefix addresses, DoS, Backdoor attacks, cookie poisoning, DDoS, CAPTCHA breaking. The defense mechanisms of DDoS can be classified according to three aspects - detection, identification and filtration. IDS can be categorized into host based, network based, distributed and hybrid. The paper has also discussed about few mechanisms - 1) Collaborative peer-to-peer architecture to defend against DDoS attacks 2) Intrusion detection in the cloud 3) Cloud based attack Defense System (CLAD) 4) Confidence Based Filtering 5) Cloud based Intrusion Detection System 6) Approach based on SOA (Service Oriented Architecture).

(Beitolahi & Deconinck, 2012) [4]

The paper classifies the DDoS defense techniques into source-end, core-end, Victim-end, distributed defense techniques based on literature review of 78 papers published between 1996 and 2012. The defense techniques can be classified into survival, proactive or reactive based on reaction time. Different types of detection techniques include sequential change point detection, wavelet analysis, neural networks, and statistical techniques. Few of the survival techniques are multiple proxy servers, enlarging backlog queue, changing the timeout for connection requests or a combination of multiple techniques. The proactive techniques can be ingress/egress filtering, route-based distributed packet filtering, D-WARD, Internet indirection infrastructure, Secure overlay services, collaborative detection. Few of the reactive techniques are pushback, K-MaxMin, Hop count filtering, Anit-DDoS, traceback techniques, DefCOM, Divide and Conquer Strategy, Client-puzzle protocols, CAPTCHA puzzles, sharing beliefs, reflector attack detection and SYN cookies.

(Mansfield-Devine, 2011) [5]

This article has covered the DDoS attacks, the attack motivation and the type of attacks derived from 14 studies conducted between 2007 and 2011. Summarized results for different factors like DDoS based on type of attacks, affected
sites, country of attack sources and the mitigating mechanisms. A comprehensive summary is provided based on the data collected from different sources for the year 2011.

(Rejimol & Thomas, 2012) [6]

This study argues that the attacks can be classified into bandwidth depletion and resource depletion. Bandwidth depletion attacks are of two types - Flood attack and amplification attacks. There were 14 papers reviewed that were published between 2001 and 2011. Resource depletion attacks are of two types - Protocol Exploit attacks and Malformed packet attacks. Mitigation methods can be classified into different types - a) Network based - provide protection of network or its servers and isolate certain portions of traffic. b) Signature based - if some packets are malicious, then other packets with same signature are discarded. c) Server side mitigation - modify the server settings like scan TCP queue and drop half open connections. d) Client centric methods - puzzles and resource pricing schemes. This paper focuses on mitigation near server. Few of the mitigation methods are - using Swarm Network, Mitigation of TCP SYN flooding with IP spoofing, Adaptive history-based IP Filtering, Probabilistic approach and HCF method.

(Mirkovic & Reiher, 2004) [7]

This literature review paper suggests, based on degree of automation, the DDoS attacks can be classified as Manual, semi-automatic (with direct or indirect communications) and automatic. This paper reviewed 76 papers published between 1997 and 2003. Based on the exploited weakness, they can be classified into semantic and brute-force attacks. Based on source address validity, they can be classified into either spoofed (again classified as routable or non-routable and spoofing techniques) and Valid. The attack rate can be constant or variant (increasing or fluctuating). Attacks are also classified as characterizable (filterable or non-filterable) or non characterizable. Other basis for classification is persistency of agent set, victim type and impact on victim. The DDoS defense mechanisms are classified into preventive or reactive. Attack prevention involves securing target using system security and protocol security. DoS prevention can be resource accounting or resource multiplication. The detection strategies can be pattern, anomaly based or third party. Based on cooperation degree, the techniques are either autonomous, cooperative or interdependent. The deployment location can be either victim, intermediate or source network. Response strategy can be agent identification, rate limiting, filtering and reconfiguration.

VII. CONCLUSION

The breadth and intensity of threats to an organization in today’s age is increasing at an unprecedented rate. There are types of attack which exploit vulnerabilities on non-target systems (botnet) to use in an attack. DDoS uses similar attack vector has been around for many decades. This has been a constant threat to business continuity and service availability. This research looked at extant literature review for analyzed and proposed mitigation controls and countermeasures against DDoS attacks. A review of 142 papers provided key insights to categorize them using a 3-layered defense in depth model. The paper categorizes each paper into one of the categories. This provides a complete and holistic view of research on DDoS protection. We also compared our work against other review and survey papers on similar topic and found our contribution is unique and novel in terms of coverage and in ontological development. The details of each of the papers have not been included in this research due to lack of space. The contributions of the paper can help researchers and practitioners alike understand how DDoS protection research has evolved over last decade or more to be better informed in their decisions.

REFERENCES

Industrial Control Systems Security Testbed

Emrah Korkmaz\textsuperscript{1,2}, Andrey Dolgikh\textsuperscript{1}, Matthew Davis\textsuperscript{1}, and Victor Skormin\textsuperscript{1}

\textsuperscript{1}Electrical and Computer Engineering
Binghamton University, Binghamton, NY 13902 USA
Email: \{ekorkma1, adolgikh1, mdavis7, vskormin\}@binghamton.edu
\textsuperscript{2}Institute of Defense Science, Turkish Military Academy, Ankara 06654 Turkey

Abstract—Cyber-attacks on critical infrastructure have been a growing concern to government and military organizations. This paper aims to study the impact of cyber-attacks on a SCADA system. To perform this research, a cyber-physical testbed emulating power generation station is designed. It contains power generation units, real-time programmable logic controllers, drives, HMI and supervisory computers. The testbed implements the process monitoring/data collection, typical for an industrial power facility. This data facilitates the deployment and analysis of several approaches for exposing different attack types and the likely impact of cyber attacks on the testbed.


I. INTRODUCTION

Cyber-attacks against critical infrastructures have been an increasingly important concern when supervisory control and data acquisition (SCADA) systems are connected to the external or internal networks.

Many envision that SCADA systems may be operated through internal networks without being connected to enterprise networks. However, this assumption is not always true in cyber-physical systems, which obtain advanced computing services and communication information through enterprise networks [1]. Furthermore, cyber-security assessment for many critical infrastructures has not been paid necessary attention until recently. However, recent studies demonstrate that cyber attacks can create extremely dangerous circumstances for industrial control systems [1], [2], [3].

The 2015 Dell Security Annual Threat Report [4] shows worldwide SCADA attacks increased from 163,228 in 2013 to 675,186 in 2014. This report demonstrates that cyber-attacks on SCADA systems grew dramatically in 2014. Additionally, 51,258 attacks occurred in the US, which is the third utmost number of attacks per nation in the world. Thus, existing vulnerability of cyber-physical systems should be considered as a large security gap in the national infrastructure.

To prevent and mitigate cyber-attacks, security vulnerabilities of SCADA systems must be investigated. However, most SCADA attacks are unreported because the companies that were exposed to cyber attacks are unwilling to discuss their data breaches that typically contain proprietary information, including financial and personal data [4]. It also shall be realized that operational industrial facilities are not suitable for any experimentation, and many technical properties of their SCADA facilities are strictly proprietary. This reality, justifies development of completely secure and isolated testbeds offering a controlled environment for various experimental studies aimed at the detection and evaluation of cyber vulnerabilities.

In this research, a real-time cyber-physical testbed based on power generation is introduced. This testbed emulates technologies that could be observed in the critical infrastructure, and offers static and dynamic data enabling users to observe the phenomena indicative of normal operation of the facility and its operation under various cyber attacks.

This paper is organized as follows. In Section 2, we describe state-of-the-art testbeds recently built by other universities and laboratories. In Section 3, we explain testbed design and components in detail. In Section 4, our data collection process is presented. In Section 5, we introduce proposed experiments that can be deployed in our testbed. Finally in Section 6, offers the conclusion.

II. RELATED WORK

Cyber-physical systems are commonly available all over the world. They can be used in the various critical national energy infrastructures (e.g. nuclear, gas, oil plants, chemical). However, it is impractical to test real cyber-attacks on these systems without disrupting their routine operation. Thus, it is crucial to develop cyber-physical security (CPS) laboratories to investigate cyber-attacks. To make secure and robust CPS work spaces, several laboratories have been built by universities and national laboratories.

National SCADA Testbed (NSTB) is a national program that combines state-of-the-art systems to research and discover critical security vulnerabilities in the SCADA and distributed control systems (DCS)[5]. The NSTB is developed by miscellaneous labs and industrial control systems (ICS) vendors including Idaho Critical Infrastructure Test Range, Sandia Center for SCADA security, Pacific Northwest Electricity Infrastructure Operations Center, Oak Ridge, and Argonne.

As a part of NSBT, Idaho National Laboratory (INL) has been developed based on cyber-security assessments of nation’s DCS and electrical substation automation components [6]. To accomplish this goal, INL has carried out projects with an electric utility vendor to assess and analyze cyber-security vulnerabilities of electrical substations. Sandia National Laboratory (SNL) is also focused on CPS threats and impacts in electric power systems, programmable logic controllers, and communication systems for NSTB [7]. It has the ability to simulate cyber attacks and defend against them, as well as to develop security frameworks for ICS.

There are also different SCADA cyber-security testbeds developed by universities. In recent research, cyber-security assessment has been increasingly used by researchers on smart power grids. In [8], the Trustworthy Cyber Infrastructure for Power Grid (TCIP) was developed by the University of Illinois...
under the Information Trust Institute. The physical system was created virtually using the PowerWorld server which simulates a real-world power microgrid. The Real-time Immersive Network Simulation Environment (RINSE) is integrated to the CPS system. By using RINSE and realistic network simulations, cyber-attacks and cyber-defenses can be tested in this virtualized physical environment.

In [9] and [10], the authors developed a PowerCyber testbed, which is composed of industrial SCADA hardware and software, a virtual 9-bus power system, and networking components. This testbed has been trained to detect cyber-attacks. The Automatic Generation Control algorithm is utilized to investigate impacts on measurements and control directions when a cyber-attack is launched.

Chen et al. have the capability to research cyber security vulnerabilities on the smart power systems under the support of Texas A&M University [2], [11]. In this testbed, a power system simulation is created by using the Real-Time Digital Simulator, LabVIEW and PXI modules, and intelligent electronic devices (IEDs). The Testbed for Analyzing Security of SCADA Control Systems (TASSCS) has also been evaluated to investigate cyber-attacks [1]. This testbed, utilizing PowerWorld simulation and OPNET tools, was developed at the University of Arizona with the intent to detect and protect against cyber-attacks on a virtualized physical system.

A different testbed approach is mentioned, which is slightly similar to our design, by [12]. While many researchers develop virtual power systems by using PowerWorld, Favino et al. implemented real programmable logic controllers (PLCs), remote terminal units (RTUs), sensors, and actuators to the simulation platform. In this way, security problems based on the real integrated control systems can be detected and identified by using different simulated attacks.

Chen et al. have the capability to research cyber security vulnerabilities on the smart power systems under the support of Texas A&M University [2], [11]. In this testbed, a power system simulation is created by using the Real-Time Digital Simulator, LabVIEW and PXI modules, and intelligent electronic devices (IEDs). The Testbed for Analyzing Security of SCADA Control Systems (TASSCS) has also been evaluated to investigate cyber-attacks [1]. This testbed, utilizing PowerWorld simulation and OPNET tools, was developed at the University of Arizona with the intent to detect and protect against cyber-attacks on a virtualized physical system.

A different testbed approach is mentioned, which is slightly similar to our design, by [12]. While many researchers develop virtual power systems by using PowerWorld, Favino et al. implemented real programmable logic controllers (PLCs), remote terminal units (RTUs), sensors, and actuators to the simulation platform. In this way, security problems based on the real integrated control systems can be detected and identified by using different simulated attacks.

### III. Testbed Design and Components

Our research group at the Network Core, Binghamton University has several facilities ready to test industrial cyber-physical systems. The testbed consists of real industrial equipment (PLCs, motors, generators, sensors, etc.) and is equipped with necessary monitoring tools. Furthermore, if the need for processing of large volumes of raw data appears, the Binghamton University Compute Cluster (BUCC) can be utilized.

Particularly, components of our cyber-physical SCADA platform can be defined as follows:

1. **Programmable Controllers**: Programmable controllers vary from very small to quite powerful devices to meet the requirement of users [13]. In our research, we have tried to test the most commonly used and different types of PLCs found in the industrial control systems because they are all open targets to the cyber-attacks. Thus, two different types of PLCs have been set up in our testbed: Allen Bradley ControlLogix, which is an advanced controller, and Allen Bradley Micro850 PLC controller. The ControlLogix is quite new technology in modern automated systems that has the capability to house multiple controllers for the system or network. We utilized a 4-slot ControlLogix and inserted a 1756-L61 controller into the ControlLogix chassis to make multiple control decisions.

2. **Physical System**: Basically, the physical system on the testbed is created to detect, mitigate, and defend against cyber-attacks on power generation-based systems. In the testbed, two power generation units have been used as shown in Figure 2.

Firstly, a 0.25 HP 3-phase AC motor and a 0.33 HP permanent magnet DC motor are connected via a coupling tie-in shaft as shown in the right side of Figure 2. The DC motor is operated as a generator which has the capability to generate voltages approximately within the range of 0-400 Volts. The generated voltage is controlled with the PowerFlex 525 AC Drive. The PowerFlex enables the operators of the testbed to control motor parameters (i.e., voltage, speed, torque, position, etc.) on the programming stations [14]. Since the generated voltage is high, single-phase loads are connected to the DC motor to protect the motor from overload conditions.

The second power generation unit is created based upon air

![ICS Testbed Interface](image-url)
handling design as shown in the left side of Figure 2. It is established by using a 3-phase AC blower motor and a 12-volt DC blower motor. The AC motor is utilized to drive the DC motor so that it generates electricity. The air which is supplied by the AC motor, has been used directly to energize the DC motor. The second generation unit produces small voltages between 0 to 12 volts so any electrical load connection is not considered on the generation unit. Additionally, the PowerFlex 525 AC Drive is used to drive AC motor. In this way, the embedded EtherNet/IP on the Powerflex enables us to change and control desired control values throughout the SCADA network.

![Fig. 2. Motor pairs simulating physical process](image)

(3) **SCADA-HMI;** Human machine interface (HMI) systems allow operators to access the control system by using visualization devices. In the testbed, a private computer with LCD monitor has been used to supervise control tags. By using HMI, PID parameters of the 3-phase DC-motor are able to be tuned. Furthermore, the operator is able to adjust and observe real-time output voltages via the HMI screen. To build, monitor, and control the HMI system, the Proficy HMI/SCADA - iFIX software is used. This supervisory monitoring software enables us to leverage more reliability, flexibility, and scalability in our system [15].

(4) **Communication Infrastructures and Software;** The communication between the controllers, PowerFlex, and HMI Desktop is established using Rockwell Automation Company’s EtherNet/IP Modules which can be used on the ControlLogix chassis. Moreover, communication of all networked devices has been bridged by RSLinx Enterprise [16]. The RSLinx Enterprise by Rockwell Automation is a well-known and commonly used communication server in critical infrastructures that enables us to share networking data between the physical system, controllers, and network devices on the testbed. This network traffic between components of the testbed is monitored by Wireshark network packet analyzer.

(5) **Lab Network Testbed;** Cyber-physical attack-defense scenarios, penetration testing of the testbed technology, and overall level security can be qualified with the help of the Binghamton University computer security testbed (BU-Testbed). The experimental BU-Testbed for information security research was created at BU under two Air Force grants [17]. The network testbed offers a secure, controlled environment for experimental analysis of the efficiency of various intrusion detection/mitigation and computer forensics systems. It allows for staging large scale experiments with real malware on thousands of interacting heterogeneous nodes. The testbed provides effective ways to collect data representing the network and software operation and it facilitates secure time sharing of the hardware among different research projects. Its enhanced security is achieved by separation and hardening of the core services. To provide necessary physical security the facility is installed in a secure isolated room at the campus datacenter.

**IV. DATA COLLECTION**

The testbed is aimed at fusing the statically available environment information (such as electronic data sheets, electric plans, PLC program code, etc.) and the logging information available at run-time. The final result of the technology is the ability to auto-generate and enforce the security policy for the manufacturing floor.

Most industrial manufacturing systems have and produce an abundance of structured and unstructured data that can be used to improve the security of the system. We can classify such source data into two types: first, static data representing the design of the system (equipment description, wiring diagrams, etc.); second, dynamic data representing current running instance of the industrial system with all its parameters (motor/generator temperatures, PLC state, network traffic, etc.). From the formalization point of view these two types of data require different handling.

In our case, we have identified the data collection process according to the availability of data in the testbed. This process can be represented through static and dynamic data compilation.

**A. Static Data**

Static data is usually stored in organized human readable text, machine readable text, and binary files. Machine readable file formats do not represent a significant challenge to read and provide a substantial amount of information about the corresponding manufacturing facilities. Below we list types of static data that are acquirable from our testbed (See Figure 3).

- **Device datasheets;** Many industrial manufacturers want to fabricate user-friendly devices. Hence, critical infrastructure control device datasheets are freely available on the internet to help clients. Datasheets of programmable controllers, motor drives, and Ethernet/IP Modules are utilized to reach the static data of cyber-physical testbed. For example, the PowerFlex datasheet allows us to reach manufacturer related information such as output voltage range, frequency range, and overload capabilities of each of the motors.
- **Electric diagrams;** This information can be reachable through the electric diagram of the cyber-physical testbed, which is connected to the lab outlets.
- **Network diagrams;** In the cyber-physical testbed, network diagrams between the controllers, HMI computer, and PowerFlex facilitate us for appropriate data collection.
- **Functional Specifications;** This data type represents the requirements of the power generation process which are basically in human readable format. We have specified how the power generation process should be proceeded.
- **PLC code;** Controllers could be thought of as mini-computers. Thus, uploaded, downloaded, and created PLC
code is easily reversible into original source code and can be utilized for cyber-attacks.

B. Dynamic Data

Dynamic data can be intercepted from the running machinery in a variety of computer readable formats. Below is the list of dynamic data observed through PLC code modifications.

- **Industrial network protocol messages;** This information is obtained through current status or changes in state of the hardware or software of the testbed. Messages describe changes in the status of individual hardware units, actions performed by individual PLCs, warnings and errors, etc.
- **Network activity logs;** These logs are built from the control network traffic monitoring. The following information can be mined from these logs: network identity and connectivity graphs, as well as industrial protocol control message dependency graphs.
- **Tracking of changes;** All changes introduced by an authorized user to control equipment are recorded and reported to the system.

V. PROPOSED EXPERIMENTS

As mentioned in the introduction, traditional SCADA systems are not designed with the highest cyber-security priority. Many industrial control systems are designed to provide a high level of system availability and operational resilience. Low cost has an important role in this process. In addition, other operational requirements, which are environmental priorities and human safety, are considered as the main priorities. [18]. However, building SCADA systems based on cyber-security assessment is not a main concern. Due to this negligence in developing secure cyber-physical system designs, the SCADA systems could be exposed to cyber-attacks.

We designed a testbed that enables us to perform risk analysis of possible cyber-attacks towards real PLCs, drives, operator’s (HMI) computer or programming station, or network. Vulnerability assessment based on a data collection process can be performed by simulating and emulating real world cyber-attack scenarios. In this section, practicable cyber-physical attack scenarios on the testbed are explained. These scenarios can be classified into four different categories:

A. Network Monitoring and Attacks on Networks

**Man-in-the-middle attacks (MitM);** MitM is a well known attack type that can be done by eavesdropping of a network. The attacker captures transferred exchange packets between sender and receiver devices through sniffing network traffic. The attacker can poison the Address Resolution Protocol (ARP) caches, change sequence number of packets, and ultimately can keep the network connection synchronized while injecting packets [19].

In order to carry out successful MitM attack on our SCADA system, the attacker computer is deployed on the testbed environment and the MitM attack can be penetrated into the network between different testbed components. MitM attack can performed based on poisoning of ARP packets between operator computer, programmer station, PLC, and PowerFlex drives. As a result of a MitM attack, the attacker might cause the following changes on the testbed.

- Static data collection process demonstrated to us that the requirement specifications of every physical component can be forced to change in the testbed. These modifications in the testbed can lead to physical damage of the
equipment. For instance, it is demonstrated that motors on the testbed have a nominal frequency operating limit. This frequency limit can easily be manipulated and increased dramatically through a MitM attack so the augmented frequency causes an increase in the speed of motor shaft or blower motor and alters the generated voltage. Basically, this means that the increased speed causes dangerous overvoltage production and may lead to a crash of the system.

- We have demonstrated that the attacker can easily halt power generation in the testbed. This problem can be thought of as an insignificant stop; however, even a short breakdown in the power generation units could engender serious catastrophic consequences. If the testbed could be thought of as a simplified small part of a power grid distribution system, this fleeting power-cut would cause blackouts and an extreme loss of economy for industrial power plants and consumers.

Local DNS poisoning: The new technological progressions currently enable system operators to use IP based mobile devices (laptops, smart phones, tablet, etc.) on the industrial control systems. While these remote controlled devices have a lot of benefits, such as preventing SCADA systems from sudden failures, they might become a target of cyber-attacks over the internet.

The testbed network also would be facilitated to carry out local DNS poisoning based on phishing attacks. Such attacks are performed by poisoning of local DNS through a rogue access point, when an operator or user types a URL address in the computer’s internet browser [20]. These attacks on critical infrastructures generally are constrained to within the SCADA network because internal SCADA networks have no access to the external internet. However, due to the unauthorized internet access by programming station users or operators, a computer can connect to a malicious web portal.

In this way, the attacker can create two consequences on the testbed [12]:
- Attacker can provide a fake picture of the controlled PID parameters and desired output voltage on the HMI computer screen.
- Attacker can acquire credentials of the operator that enables direct access to the testbed.

B. Network Congestion and Delays Affects

Denial of Service (DOS) attack: Our experimental setup could be used to analyze the behavior of the SCADA network when Denial of Service attacks are carried out. These attacks can likely be successful for critical infrastructures that have connection to the external internet. For instance, power generation and transmission infrastructures which use wide area monitoring, protection, and control (WAMPAC) systems to improve grid stability can be the target for this attack. With this attacking technique, SCADA infrastructures can lose connection to the external network and therefore the attacker may cause potential disruption or stoppage of the SCADA system.

In the testbed, the attacker could increase data-packet-flow severely on the SCADA server using fake service requests which include victim IP addresses. Basically, the SCADA server tries to reply to and overwhelming number of packets that come from attacker side. As addresses are forged, the SCADA server cannot obtain any particular information and thus connection will stop with a timeout [21].

C. Attacks on Controllers, Sensors, and Drives

The attacker can achieve unprecedented levels of control over the SCADA system by subverting the controllers, sensors, and drives. The attacker can achieve this scenario in two different ways over the network: (1) By injecting of malicious software on internal or external network. (2) By creating firmware modification attacks.

Malicious software injections: These attacks aim to disable process control systems by causing computer epidemics [17]. To launch those attacks, computer malwares such as worms, viruses, trojans, etc. can be injected into the industrial control systems.

Such malwares on the testbed could infect specific SCADA components that may include:
- ControlLogix/Micro850; so that viral infection disables SCADA controllers.
- PowerFlex-525 devices; so that power generation can be aborted via specific malwares.

With this attack type, the attacker can cause dangerous circumstances on the testbed. Based on the data collection process, specific limits of the PowerFlex drive can be obtained and relevant malwares could be written. For example, the PowerFlex drive has a voltage regulator that prevents users from undesirable overvoltage issues. However, the device could not be stopped when actual deceleration times is longer than commanded deceleration times [14]. The attacker can utilize this information and program malware to carry out on the drive.
In the testbed, the ControlLogix back-plane has firmware so that the device provides communication with other controllers as well as I/O modules. By modifying the ControlLogix firmware, malware could be created. As result of malware injection into the testbed controllers, significant system disruptions and motor or controller failures can be seen.

D. Attacks on HMI and Programming Stations

The critical infrastructures can be hijacked effectively by injecting malware on HMI computers or programming station computers. Many SCADA computers do not have any effective anti-virus protection. Even if anti-virus programs are installed on the computers, they are vulnerable to cyber-attacks because these computers’ update cycles are extremely long. Thus, it is possible that the control process network can be vulnerable to malware injections.

Process network malware injection; The process network malware can infiltrate through the external network or internal network. While on the internal network the injection could be achieved by removable storage devices (such as CDs, DVDs, floppy disks, as well as USB drives). Network-based attack techniques might be carried out on the external network. As an example, for network based attack the Sandworm was carried to hack Cimplity HMI Solution Suite by researchers [23]. The results show that by using such malware, the HMI or programming computer could be used to manipulate actual SCADA data or network.

Our cyber-physical testbed enables us to see the consequences of process network malware. If the malware is injected in the programmer or HMI computer, high-impact results are observed because these computers are thought of as the main workstation of the testbed. By implementing malware to the programming station or HMI computer, the attacker can directly affect the power generation units and consequently the functionality of the testbed totally could change.

VI. CONCLUSIONS

In this work, a cyber-physical testbed based on power generation is explained. Two different power generation units are used with two different design approaches; first, blower motors are connected to each others and second an AC motor is directly coupled to a permanent magnet DC motor. Power generation units are controlled with IP based real-time intelligent electronic devices, in this way the system transformed into a cyber-physical system. We separately discuss intelligent electronic devices and their connection (controllers, HMI devices, drives, and network components) on the testbed.

Furthermore, the testbed is structured to gather static and dynamic information available. Data collection on the testbed has demonstrated that static data is usually seen in a human readable format, while dynamic data is acquired in a computer readable format.

Based on the data gathering process, the testbed architecture is evaluated in terms of the probability and availability of likelihood cyber-attacks on the SCADA systems. Proposed experiments in this paper demonstrate that the testbed is an effective tool to expose cyber-attacks that may cause massive amounts of damage to the system and enables us to study them.

ACKNOWLEDGMENT

The authors would like to thank the Office of Naval Research (ONR) for their support under the award number 15114581.

REFERENCES

FrankenSSL: Recombining Cryptographic Libraries for Software Diversity

(Invited Paper)

Bheesham Persaud, Borke Obada-Obieh, Nilofar Mansourzadeh, Ashley Moni, Anil Somayaji
School of Computer Science, Carleton University
Ottawa, Ontario, Canada
{bheesham, borke, nilofar, ashley, soma}@ccsl.carleton.ca

Abstract—Many security vulnerabilities arise from protocol implementation flaws. Software diversity can reduce the impact of such flaws; however, in practice there are relatively few implementations of important protocols due to the challenge of making interoperable, reliable, and efficient implementations. One strategy for increasing the number of variants is to mix and match components from different implementations. Just as biological systems recombine DNA from parent organisms to create children, we could mix and match code from different protocol implementations to get thousands of variants. To achieve this goal, two fundamental challenges must be overcome: we need to demonstrate that these variants can be created while preserving functionality, and we need to show that these variants are not all susceptible to the same vulnerabilities. As a step towards this goal we are developing a method for recombining implementations of Transport Layer Security (TLS), specifically OpenSSL, LibreSSL, and BoringSSL. In this paper we report on our progress to date.

Keywords—software diversity, software recombination, library security

I. INTRODUCTION

Each year, large numbers of computer security vulnerabilities are reported as well as strategies to exploit them. To address such discoveries, most Internet-connected platforms have extensive resources devoted to developing and deploying patches for software vulnerabilities. Despite such efforts, however, many systems remain unpatched, and even patched systems are subject to exploits of zero-day vulnerabilities. While formal methods hold the promise of provably secure systems, they are reported as well as strategies to exploit them. To address such discoveries, most Internet-connected platforms have extensive resources devoted to developing and deploying patches for software vulnerabilities. Despite such efforts, however, many systems remain unpatched, and even patched systems are subject to exploits of zero-day vulnerabilities. While formal methods hold the promise of provably secure systems, they have had success with the automatic recombination of program fragments of other programs [10]. Further, in past work in our group we have had success with the automatic recombination of code from different sources would most likely result in a non-functional program. The entire field of genetic programming, however, is based on creating new programs by recombining fragments of other programs [10].

The dangers of “software monocultures,” or the lack of software diversity, have been noted by many researchers [4], [5]. Almost all work in software diversity, however, has been focused on the automatic creation of software variants at levels not specified in program source code [6], [7]. This style of diversity can be effective against attacks that make use of regularities in program memory layout and behavior (such as buffer overflows). Yet there exists many vulnerabilities which do not make use of such regularities but are very exploitable. Whether it be incorrect authentication checks, race conditions, protocol implementation errors, or even more idiosyncratic exploitable mistakes, such vulnerabilities cannot be mitigated though semantic-preserving changes in how code is compiled or run because the problems are in the programs’ semantics.

A diversity-style approach to mitigating such vulnerabilities thus must make use of different implementations of the same functionality. Creating independent implementations of critical functionality is a practice long followed in the fault-tolerance community [8], [9]. Today there exist multiple open-source implementations of critical Internet infrastructure, server applications, and even many desktop applications such as web browsers. While the presence of these variants provides a small amount of diversity, an attacker still gets a lot of benefit from exploiting popular variants because they are installed on millions of systems.

Thus to improve security through diverse implementations we need more than a handful of variants; instead, we need enough variants that it is not feasible for attackers to study and develop exploits for each one. We cannot hope to create so many variants manually, and we cannot create them by simply randomizing the behavior of individual implementations (as has been done in previous work). The key insight of this work is that we could combine portions of implementation with one another to create diversity. Done properly, we would get a combinatorial explosion of variants that would be challenging for an attacker to characterize, let alone exploit.

Differences in program structure and function would suggest that this would be a fool’s errand, as any arbitrary mixing of code from different sources would most likely result in a non-functional program. The entire field of genetic programming, however, is based on creating new programs by recombining fragments of other programs [10]. Further, in past work in our group we have had success with the automatic recombination
of regular program binaries at the object-file level [11], [12]. To get significant improvements in diversity that have measurable improvements in security, however, we need to maximize implementation diversity while also preserving higher-level semantics, particularly at the network and user interface level, and do so with existing programs—something past approaches cannot do.

Rather than recombining portions of entire programs, here we propose to recombine portions of security-critical libraries. Libraries present a stable API to applications yet can differ in their internal semantics. So long as libraries implement the same API, they may be interchanged. Thus, so long as we can create library variants that implement the same API we can get most of the benefits of implementation diversity at much lower cost.

As a first test case, we are working on recombining libraries implementing standard SSL/TLS functionality, as such libraries are both critical for security-related functionality and have been shown to have many exploitable vulnerabilities. Here we report on our progress on recombining elements of OpenSSL [13] with two OpenSSL forks, BoringSSL [14] and LibreSSL [15], [16]. As we will explain, recombination can be challenging even between libraries that have recently diverged from each other. Nevertheless, results to date indicate that our approach has promise.

The rest of this paper proceeds as follows. We first describe background and related work in Section II. We discuss our approach in more detail in Section III. Sections IV & V present details of our design and implementation progress, respectively, while Section VI reports on our preliminary results. Section VII discusses limitations and plans future work; Section VIII concludes.

II. BACKGROUND
A. Software Diversity for Security

The idea of software diversity was proposed in fault tolerant systems as far back as the late 1970s, in the form of N-version programming [8]. In 1993, Cohen et al. [4] also recognised the importance of software diversity for software security and reported potential dangers of carrying out “software monocultures.”

While using teams of developers to create independent implementations has been proposed in the context of security [17], the prohibitive costs of doing so have generally precluded this approach. Instead, most work on software diversity for security has followed the direction described in 1997 by Forrest et al. [6] where compile time and runtime diversity (really, randomization) are added in ways that preserve the source code level semantics of the application [7]. Most notably, address-space layout randomization (ASLR) [1] is now a common feature of modern operating systems; however, randomization approaches have also been proposed at virtually all levels of program behavior including the instruction set [13], [19], memory address [20], function layout [21], and system calls [22], [23]. Randomization-based defenses are often criticized as they only provide probabilistic protection; however, through N-variant programming [24] where the execution of variants are compared at runtime, it is even possible to get deterministic guarantees.

Because these transformations preserve source code-level semantics, they are primarily useful in detecting memory corruption attacks. Randomization at higher levels is also possible, however, in order to mitigate attacks such as SQL attacks [25] and cross-site scripting attacks [26]. But again, such defenses cannot mitigate vulnerabilities that arise from more general classes of implementation mistakes or design errors.

B. Software Recombination & Patching

While most code is created manually by programmers, there is a large body of work on automated ways to create code. Compilers and assemblers are of course automated code translators and generators; further, any declarative programming system directly or indirectly generates code. By their nature, however, these types of systems produce systems with relatively little diversity.

Natural evolution has proven to be very good at automatically producing diverse systems, so in considering ways to automatically generate diversity it makes sense to look to systems for evolving code. Most work in genetic programming (GP)—the main community concerned with evolving code—has focused on the evolution (really, searching in a pre-defined space) of code which satisfies various fitness functions [10]. While there is work on diversity in genetic programming, such work is mostly concerned with improving the quality of GP search rather than creating diverse solutions for their own sake [27], [28], [29]. While most work on genetic programming only works with S-expressions, not the source or object code of standard programming environments, there has been progress in using GP to automatically create patches to repair security vulnerabilities [30].

The work closest to ours here is that of Foster and Somayaji’s ObjRecombGA (2010) [11], [12]. ObjRecombGA used a genetic algorithm to search for ways to successfully recombine object files between two closely related programs to create (with the help of a specialized linker) a new program that combines the functionality of the two “parent” programs. This approach was successful in combining functionality between variants of open source programs including a UNIX shell program (GNU sed), a web browser (Dillo), and a game (Quake). While object-level recombination may hold promise for creating variants for security purposes, ObjRecombGA tends to create variants that are mostly one version of the program with small additions from the other parent. It does this in order to minimize the linking problems caused by symbol, data structure, and function declaration mismatches between object file variants.

Thus, to create a larger number of variants, we need to study and develop techniques for managing the incompatibilities that arise between divergent code bases.
C. OpenSSL Security

The OpenSSL library started in 1998 as an implementation of TLS. Over time, however, it has become a comprehensive library implementing a wide range of cryptographic primitives and protocols that a developer might require. While OpenSSL is widely used, its security record has been problematic at best. In 2014 and 2015 the OpenSSL project reported a total of 54 Common Vulnerabilities and Exposures (CVE). As of May 2016, many more vulnerabilities have been documented in OpenSSL [31]. While many of the past vulnerabilities in the OpenSSL library have been things such as buffer overflows, integer overflows, and other types of memory corruption attacks, many others were implementation errors that are not easily mitigated through an implementation-level randomization approach. Consider these examples:

- CVE-2003-0147 (OpenSSL Advisory) 14th March 2003: RSA blinding was not enabled by default, potentially allowing attackers to obtain the server’s private key through a timing attack.
- CVE-2008-0166 (Debian Advisory) 9th January 2008: OpenSSL on Debian systems generated predictable random numbers due to a change made to remove warnings generated by Purify and valgrind [32].
- CVE-2008-5077 (OpenSSL Advisory) 7th January 2009: Incorrect checking of a return result caused bad signatures to be treated as being correct.
- CVE-2014-0160 (OpenSSL Advisory) 7th April 2014: A critical buffer over-read bug was discovered and named Heartbleed [33].

The above vulnerabilities are all attributable to idiosyncratic implementation mistakes. Different implementations would be unlikely to have made exactly the same mistakes.

D. OpenSSL Forks

Soon after the Heartbleed bug in OpenSSL [33] was disclosed, OpenSSL was forked by two groups looking to improve the code quality of the project: LibreSSL, managed by OpenBSD developers [10], and BoringSSL, managed by Google developers [14].

LibreSSL has been designed as a streamlined drop-in replacement for OpenSSL, implementing the same basic API and much of the same functionality, although many uncommonly used and insecure algorithms have been removed. In contrast, BoringSSL, while also streamlined, has not been intended as a drop-in replacement; instead, the BoringSSL developers have been willing to change APIs and functionality to improve security, performance, and usability. The BoringSSL developers at Google are willing to do such changes because they also control the codebases that use BoringSSL. Nevertheless, the basic APIs provided by BoringSSL are still largely identical to those provided by OpenSSL.

Often when forks are created the original project loses resources; however, around the same time as BoringSSL and LibreSSL were founded, the OpenSSL project also received an influx of developer attention and funding. Thus, OpenSSL has proven to be a kind of “natural experiment” in the creation of security-critical code diversity with multiple, mostly compatible implementations now in existence.

III. Library-Level Software Recombination

In order to create software diversity that will improve software security, here we propose to focus our efforts on creating diverse implementations of security-critical libraries rather than diversifying entire applications. Our strategy follows that of ObjRecombGA [11] in that we wish to recombine the code from different implementations. In order to get finer-grained recombination (and hence more variants) our aim is to enable recombination at the function level rather than the object file level.

We propose to focus on library-level recombination for multiple reasons.

- Library-level vulnerabilities can affect many applications; thus, efforts to improve the security of such libraries will tend to have disproportionate benefit.
- Important libraries often have multiple implementations, and these implementations are often closely related. (For example, libraries are often forked in order to satisfy the requirements of large applications.) Even when these implementations do not provide identical APIs, they are still often substitutable with modest effort.
- Fine-grained recombination will likely require some amount of manual effort. By focusing on libraries we can leverage this effort to improve the security of multiple applications.

FrankenSSL thus is a first effort to do library-level software recombination between OpenSSL library and recently created forks of OpenSSL. As explained in the last section, OpenSSL is widely used, security-critical, has a long history of vulnerabilities, and there exist high quality forks of OpenSSL.

IV. Design

While the basic principle behind FrankenSSL’s design can be translated to any other library with multiple forks or implementations, for clarity here we describe our design in terms of three libraries, OpenSSL, BoringSSL and LibreSSL.

A library can be modeled as an abstract cluster of functions that all interlink. This abstraction is not that far from the reality of their representations in a filesystem as sets of object files with symbolic references to each other.

Applications that wish to use SSL/TLS normally interact with their library of choice through APIs. The functions that represent those libraries also interconnect with each other and use each other through internal APIs, since modern software engineering prioritizes modular over monolithic architecture for functionality.

Normally a target application would link (through its APIs) to a set of functions within the function cluster of the library, and those functions would link to other functions or each other. This pattern creates a straightforward hierarchy of abstraction, from the application (that asks for data to be encrypted or
decrypted) down to terminal functions in the graph (that manipulates and manages data and memory).

The three implementations of SSL can be considered to be three clusters of functions, with some overlap. Since they are related to each other (with BoringSSL and LibreSSL both being forks of OpenSSL), some functionality is shared across multiple libraries. Each effective library can be modeled mostly as a directed graph, starting from the application and splaying out into the shared function cluster. See Figure 1.

From this formalization, it’s easy to see that alternate potential implementations of SSL can simply be modeled as unique graphs. FrankenSSL’s goal is to procedurally generate random, viable API/linker graphs upon this shared library of dependencies.

Exploits in any of the implementations are ultimately located in a specific function, or in the interplay between a small set of functions. As a consequence, randomized implementations drastically cut down on the effective portability of any given exploit, since an attacker can’t guarantee that any given system has the relevant function or function constellation. Even if a given implementation does have the target functions in the target arrangement, alterations to the rest of the graph may lead to the runtime being structured differently in a manner reminiscent of Address Space Layout Randomization [1].

V. Implementation Progress

While we eventually would like to recombine libraries at arbitrary function boundaries, initially we have focused on recombining the libraries at the level of calls that are commonly made by applications into the library. Applications created using OpenSSL (and thus BoringSSL and LibreSSL) generally follow the same workflow: initialize the library, initiate a TLS session, configure the TLS session, read from a session, write to a session, deinitialize the session, and then proceed to deinitialize the library. We wish to distribute these calls across the different libraries: we want to initialize with OpenSSL while reading from a session using LibreSSL’s functionality, for example, as well as distributing internal calls across implementations.

As a first step towards this goal, we have focused on recombining the initialization and deinitialization code of each of the libraries. Specifically, we focused on recombining calls to the following functions:

- SSL_load_error_strings()
- SSL_library_init()
- TLS_client_method()
- SSL_CTX_new()
- SSL_CTX_free(), and
- SSLeay_version().

In the rest of this section we describe the project scaffolding, build system, and linking issues. In the next section we present the results of these efforts.

A. Scaffolding

Installing OpenSSL, BoringSSL, and LibreSSL side-by-side on the same system can be a daunting task for novices. Each of the libraries—due to the nature of having the same origin—export similar symbols, build artifacts with the same name, and install to the same directories.

Installation of any one implementation on a system-wide basis would mean that another implementation cannot be used due to the resulting collisions. This makes having an installation of more than one implementation based on OpenSSL difficult, to say the least. These collisions make it impossible to link all three into the same application without any edits to any of the implementations.

Our solution is to link applications against a hand-crafted dynamically linked stub library. We then specify different combinations of implementations at runtime using LD_PRELOAD. This method gets rid of any high-level namespace collisions, however, it only works if all three libraries can be linked in to the application at once. To allow this to happen we had to change how the libraries were built.

B. Build Infrastructure

One of the changes that both BoringSSL and LibreSSL made from OpenSSL was a change in build infrastructure. OpenSSL makes use of a custom build infrastructure based upon standard Makefiles. BoringSSL and LibreSSL, however, were changed to use CMake.
Initially we tried converting the build system for OpenSSL to CMake. This proved to be challenging. OpenSSL requires many files to be generated before compilation and a number of compiler flags to be passed in at configuration time. Many of these flags are not documented, and as such are difficult to account for when migrating the build system.

Rather than migrate all of the code to a single build system, we instead modified each as needed in order to allow code from all three libraries to be incorporated into a single binary. To accomplish this we had to address the symbol resolution problem.

C. Symbol Resolution

Versioning the shared objects allows maximal modularity as we can combine exported symbols from each linked library at runtime. While OpenSSL versions symbols by default, BoringSSL and LibreSSL do not; thus, BoringSSL and LibreSSL export the same symbols for various functions, preventing them from both being linked in to the same binary.

Before we could implement a wrapper around each library we needed to ensure that the correct function would have been called from each library. That is, we needed to make sure that the program, at runtime, knew which version of a function to call.

We investigated multiple solutions to get around symbol collisions, namely: the use of objcopy to rename important symbols, writing a compiler pass, and using symbol versioning.

- **objcopy**: Using this method would require us to modify not only the compiled shared object files, but also to create our own header files with prototypes to each of the functions. This would, in essence, give a result equivalent to writing our own compiler pass.

- **Compiler pass**: This method included creating a plugin for GCC or writing an LLVM pass to rename symbols of each library at compilation time such that there would be no colliding symbols. However, writing a plugin for GCC or a pass for LLVM would require anyone who would want to build the software to have their compiler configured correctly. While not unreasonable, it is also not ideal as it would introduce complexities that come with working at the compiler level.

- **Symbol versioning**: After reading the section on Symbol Relocations, and Export Control of Drepper’s “How To Write Shared Libraries” [34], we decided that his method was perfect for our needs. It required an additional two compiler flags and the creation of a symbol map which was trivial to write. By exporting the version of a symbol, each library we linked to would have saved a reference to the library it was referring to at link-time, which meant that at runtime the expected functions should be called.

Ultimately, we moved forward with symbol versioning as it is, we believe, the most transparent way to achieve non-colliding, globally exported symbols and only requires minimal changes to each codebase.

D. Runtime Linking

Each method that is wrapped from OpenSSL, BoringSSL, and LibreSSL is packaged into its own shared object file. While this generates three times as many files as wrapped methods, it safely separates each method from each library.

We link each method from the desired library to a test program which uses each of the methods. Linking is done at runtime by specifying each wrapper through the use of the `LD_PRELOAD` environment variable. If the program runs without any errors, it is considered to have a successful combination of methods.

VI. PRELIMINARY RESULTS

We have tested all combinations for initialization and deinitialization using the three libraries and none of them are viable for practical use. Programs have either thrown a fatal error or have leaked memory—not ideal in either case. The memory errors arise because of the difference in the SSL_CTX structure defined in each library. This causes complications as the SSL_CTX structure is used to store state throughout each session.

We created a simple application to test whether or not a combination was viable. Any combination ran by the program had to go through the initialization and deinitialization processes without errors to be considered a viable combination. Errors during runtime were checked for using valgrind and GDB.

<table>
<thead>
<tr>
<th>init</th>
<th>deinit</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>OpenSSL</td>
<td>OpenSSL</td>
<td>Works</td>
</tr>
<tr>
<td>OpenSSL</td>
<td>BoringSSL</td>
<td>SIGABRT</td>
</tr>
<tr>
<td>OpenSSL</td>
<td>LibreSSL</td>
<td>SIGSEGV</td>
</tr>
<tr>
<td>BoringSSL</td>
<td>BoringSSL</td>
<td>Works</td>
</tr>
<tr>
<td>BoringSSL</td>
<td>LibreSSL</td>
<td>Leak</td>
</tr>
<tr>
<td>BoringSSL</td>
<td>OpenSSL</td>
<td>Leak</td>
</tr>
<tr>
<td>LibreSSL</td>
<td>LibreSSL</td>
<td>Works</td>
</tr>
<tr>
<td>LibreSSL</td>
<td>BoringSSL</td>
<td>SIGABRT</td>
</tr>
<tr>
<td>LibreSSL</td>
<td>OpenSSL</td>
<td>SIGSEGV</td>
</tr>
</tbody>
</table>

Read and write operations in the sessions were not supported by any combination that was not error-free; thus, we omitted these results from the above table.

There was no case in which any of the libraries worked well together, even with some rudimentary “shim” code. The differences between each library’s SSL context, and what is required to correctly allocate it vary too greatly to work without significant amounts of interface code.

VII. DISCUSSION

We have attempted to recombine parts of OpenSSL, LibreSSL, and BoringSSL to create functional variants. As highlighted in the previous section, the namespace and symbol collision issue that arises from the recombination of the three libraries have successfully been resolved. We were also successful in developing a proof-of-concept application that
can swap out implementations of a function at runtime. We were however unable to resolve the memory error issue which resulted from inaccurate memory allocation by the variants produced. As such, we were unsuccessful in the production of fully functional variants.

While automated recombination methods such as those used by ObjRecombGA [11] might allow for the creation of more functional variants, changes to core data structures such as the SSL context SSL_CTX will require the creation of glue code of some kind to account for these differences, either by maintaining parallel versions of the data structures (one for each version of the library used) or by dynamically changing the data structure as it is accessed. While this adaptation could eventually be automated, a first step would be to develop such glue code manually. We have made some progress with the development of such code; finishing this work is a key goal for future work on this project.

One of the surprising results of this work is the degree to which the forks of OpenSSL have diverged. Both BoringSSL and LibreSSL have reorganized the code, changed build systems, and changed core data structures. These changes are likely due to the large technical debt that OpenSSL had incurred over the years which contributed to the large number of vulnerabilities and motivated the creation of the forks in the first place. Indeed, we ourselves ran into issues regarding the build environment of OpenSSL, with us finding that multiple files generated before compilation in OpenSSL are not documented. We can thus understand the motivation for the work that has been done on cleaning up the code and build system in BoringSSL and LibreSSL. Going forward, then, it may be wise to focus more on recombining the forks of OpenSSL rather than using the original library.

The extensive modifications also give additional motivation for the development of FrankenSSL. At a minimum, such extensive code changes have likely introduced vulnerabilities; further, as these codebases diverge, it becomes more and more difficult for vulnerabilities found in any one of them to also be found and removed from the others. Library-level software recombination is a strategy that can mitigate the risks of these vulnerabilities.

VIII. CONCLUSION

This paper presents a new methodology for defending systems from intrusion through diversity, library-level software recombination. It also presents progress on the development of FrankenSSL, a randomly-generated recombination of code present in OpenSSL, LibreSSL, and BoringSSL.

While we are able to solve the basic problem of doing the recombination using symbol versioning and per-function object file wrappers, changes to core data structures prevents complete interoperability of initialization and deinitialization routines between libraries. Glue code to manage these incompatibilities needs to be developed in order to allow for finer-grained recombination between these libraries. The development of such glue code, automation of the recombination process, and an evaluation of vulnerability mitigation are key goals for future work.

ACKNOWLEDGMENTS

The authors would like to acknowledge the support of Canada’s Natural Sciences and Engineering Research Council (NSERC) through their Discovery grants program.

REFERENCES


A Sophistication Index for Evaluating Security Breaches

Nic DePaula
Department of Informatics
University at Albany, SUNY
Albany, NY, USA

Sanjay Goel
Information Technology Management
University at Albany, SUNY
Albany, NY, USA

Abstract—The focus of this research is to develop a sophistication index for evaluating security breaches due to cyber-attacks. Although reports about cyber-attacks elucidate the sophistication involved in a given security or data breach, it is difficult to compare the sophistication of breaches across multiple attacks. Once we have an attack sophistication index, incidents can be compared and consequences of sophistication level, such as damage impact, may be assessed. To develop this index, we first explore the evolution of malware and security breaches due to cyber-attacks. We then develop a dataset of 32 security breach incidents that were well publicized in the media over the last 10 years. Based on our literature review and expert evaluation we develop a set of basic features that may be used to classify the sophistication level of an attack. We also ask security experts to rate the incidents based on their perceived notion of sophistication. Our results indicate that our sophistication index is correlated with the level of perceived sophistication. We also observe that the level of sophistication of the well publicized attacks have not consistently increased over time. Simpler attacks continue to propagate and cause damage. Moreover, a correlation of the sophistication level and the damage caused (number of records exposed) is not significant, suggesting that a more sophisticated attack is not necessarily more damaging to the institution at question.

Keywords—cyber-attack sophistication; security breach sophistication; attack complexity; data breaches

I. INTRODUCTION

It is generally acknowledged that cyber-attacks and security breaches have become more sophisticated over the years [1], [2]. Professional reports suggest characteristics of these increasingly sophisticated attacks, indicating, for example, how they possess certain types of backdoors or originate from certain nation states or networks of criminals [3]. Sophistication is also a noted feature in the general media. In 2009 a Wired article noted a description of the RBS World Pay cyber-attack as "perhaps the most sophisticated and organized computer fraud attack ever" [4]. In 2015, the New York Time reported on "likely the most sophisticated [cyber] attack" up to then, one in which an estimated $300 million were stolen from banks across the world [5]. In the more academic literature work has been done exploring the technical evolution of malware, the development of taxonomies and tools for dealing with these new kinds of sophisticated security threats [6], [7].

However, limited work has been carried out examining defining characteristics of cyber-attack sophistication, and more specifically we have found no metric for sophistication of security breaches per se. There have been studies identifying and suggesting characteristics of sophisticated attacks, and metrics related to the concept of complexity. For example, one way of looking at sophistication could be from the complexity of the code used in the attack, such as with Kolmogorov Complexity measures that are based on the computational resources required for a program to carry out a task [8]. Another measure of complexity or sophistication would be to model each attack as an attack tree [9] and then based on the parameters of the tree estimate the complexity of the attack.

Security breaches due to cyber-attacks occur in a variety of ways for a variety of purposes and this makes it difficult to obtain a single scale to measure it. Not only are there multiple pieces of technology and organizational resources that go into a cyber-attack, the concept of sophistication is inherently relative—relative to time and technical progress as well as relative to the purpose of the attack. That is to say, an attack that is technically simple may be considered sophisticated given the ingenuity of the attacker or the precision with which its particular mission is carried out. Moreover, what is sophisticated in a given year could be considered relatively ordinary in a short period of time. These characteristics of the concept of sophistication make it difficult to define an objective measure of sophistication that is useful through a meaningful period of time and that also does not require difficult to acquire information, such as attack purpose and resources of the attackers.

In this preliminary study we investigate the following questions. How has the sophistication of security breaches due to cyber-attacks evolved over time? What specific features of these security breaches may be used to measure sophistication? In section 2 we present a literature review on the evolution of cyber-attacks and propose a general set of features that can be used to define an attack as sophisticated, our sophistication index features. In section 3 we explain our methodology which involves the creation of a dataset of the most well publicized cyber-attacks and data breaches over the past 10 years. We use data breaches since they are more reliably reported, given regulations to do so. We developed a survey that includes explanations about these incidents which we use to obtain expert input, both to identify feature terms and a score of perceived sophistication. In section 4 we present our preliminary findings, which involve a set of tests about the measure we are proposing. Lastly we provide a set of discussion points and insights in how we may interpret our findings, some of our limitations and suggestions for future research.
II. LITERATURE REVIEW

Security breaches due to cyber-attacks incorporate technical components in the use of specific malware and technical skills as well as psychological components in exploiting user vulnerabilities. Moreover, the availability of resources such as markets and the support of nation-states play a role. In how malware are developed, hacking skills are acquired and knowledge about the target is obtained. All of these components reflect the complex nature of cyber-attacks in general. In this section we carry out a literature review that identifies how these components have evolved and what set of features are present in the incidents that are considered more sophisticated. Since this is an iterative study, the ultimate framework of analysis we propose has been assisted by the results of our data collection itself.

A. Malware and Attack Evolution

The identification of computer system flaws that could be exploited for penetration were identified in the early developments of computer systems [10], [11]. Although initially developed as an exercise in computer science, during the 1990s various malware types were being developed for the disruption of systems [12]. At this time, malware witnessed a variety of modifications and improvements, consisting of the addition of features, such as multipartite and polymorphic characteristics [13], [14]. The notion of “increasingly sophisticated attacks” would come in the early 2000s [15]. Casey suggested that “sophisticated security breaches” were characterized by the use of “customized tool sets”, “time-activated backdoors” and communication with command and control servers [15]. These sophisticated attacks deleted logs, altered date-time stamps and installed utilities to remove evidence of intrusion.

Some of the examples of sophisticated malware used in cyber-attack security breaches included the Zeus Trojan first discovered in 2007 infecting systems at the U.S. Department of Transportation [16]. This malware gathered information from Protected Storage areas of a machine or by intercepting websites, enabling keylogging and form grabbing [17]. As a trojan it used a back-door to contact and command and control server, allowing remote access to the infected system. In 2008 the Conficker, or Downadup, worm was discovered, demonstrating the integration of a number of components that made it “the most prolific worm in recent times” [18]. The most famous attack to date is perhaps Stuxnet, discovered in 2010, which infected the Nantaz Iranian nuclear reactor. Among other features, the attack used valid digital signatures, multiple zero-day vulnerabilities and manipulated nuclear programmable controllers [19]. From the Internet Worm written by Robert Morris in 1988 [20], to the tools used by hacktivists [21] and to the Stuxnet worm released in 2010, we can suggest that the sophistication levels are vastly different.

Besides malware, actors and their motivation have also evolved. As the actors have changed from aficionados to criminals, hacktivists and nation states, the stakes have risen and resources to support development and the launch of attacks has also increased. Furnell and Ward, for example, showed a disproportional increase in the number of non-destructive malware in the early 2000s [8]. Instead of disrupting their hosts, these malware implemented backdoors in order to extract data to be sold in the black market. A complex black market has emerged to support cyber criminal activity with various web platforms, a hierarchy of knowledgeable and less knowledgeable participants, and a range of products including credit card numbers, passwords, user names, contact information, malware, hacking tools and other potentially useful data [22].

The black market for hacking tools has given incentive to malware writers to create increasingly sophisticated tools and detect vulnerabilities in the Internet infrastructure. Moreover, this in turn has led to certain items, such as rootkit malware or a zero-day vulnerability to not be considered sophisticated by itself, as it may be simply purchased. Nevertheless, the combination of the different features hypothetically does increase sophistication, as it signifies ingenuity and complexity in successfully managing the different components of the attack. Entities most able to use these tools in combination with tools never previously observed are nation states, given the resources they command and their political-economic purposes. Indeed, units of governments are likely the most sophisticated perpetrators of cyber-attacks for political and economic reasons [23]–[25].

B. Measures of Sophistication

Most definitions of sophistication have been provided in a purely descriptive manner. We have found no quantitative analysis of sophistication per se in our research. However, some attempts have been made to provide some basic or general categories of sophistication features. [26] suggested that the evolution of malware could be categorized by increases in speed, increase in stealth capability and changes in purpose. [27] suggested that sophisticated attacks consist of “methodical reconnaissance” to learn more about the victim and target system as well as “multiple agents” which “persist in the target’s network”. Although there is not much consensus on what advanced persistent threats (APTs) entail, the term has been considered as synonymous with sophisticated attacks. [28] explains that APT’s use multiple techniques in combination, backdoors, social engineering and persist in the network of their victims.

Attacks metrics have been developed for estimating vulnerability of systems and organizations, such as those of CERT, SANS, and NIST. The Common Vulnerability Scoring System, for example, consist of 3 metric groups regarding characteristics of the system vulnerabilities [29]. One of these measures is on “access complexity”, the complexity required from the attack to overcome the vulnerabilities. Access complexity involves the following elements: access to elevated privileges, spoofing, the use of social engineering, and the ability of the attacker to identify vulnerable configurations. The CVSS also contains measures related to the impact and damage potential if certain vulnerabilities are exploited. These measures, however, are generally focused on assessing vulnerabilities and not attack characteristics themselves.

Based on this literature, we propose a preliminary set of features that characterize sophisticated attacks as summarized in table 1. Each general feature type includes a set of particular items or criteria. We propose that a sophistication index may be developed using these 5 feature types. These feature types
were chosen based on the review of the literature and iteratively based on expert input from our survey. The index thus may be said to reflect partially reflect perceived sophistication. The index is calculated based on the identification of items in a particular attack. For example, if an attack does not have any of the items listed for any of the feature types, the score is 0 (zero). If an attack contains one item for each feature type, the score is 5 (five). Essentially, each feature type receives one additional score.

We propose that a scoring system such as described in Table 1 can be used to evaluate sophistication across publicized security breaches due to cyber-attacks. Moreover, the general types of features and terms identified allow for ease of classification regardless of the specific tools used. The feature types identified and the scoring system created are based on both the literature review and expert input on perceived sophistication. Although purpose-related characteristics have been highlighted in the literature we thought that using purpose as a type of sophistication characteristic would be difficult to measure. Moreover, at this stage we have good results with the current sophistication index, the derivation of which are explained further in the following sections.

### III. Methodology

This study is divided in three stages. In the first stage we carried out the literature review and identified characteristics of sophisticated attacks as presented in the information security literature. We then acquired information on a number of data breaches and asked information security experts to identify characteristics of the attacks they perceived as more or less sophisticated. We used these responses to refine and improve our model. Lastly, we asked security experts to rate the same set of cyber-attack incidents in order to compare their own score of perceived sophistication with our sophistication index.

**A. Data and Survey Development**

In our data and survey development process, we first obtained secondary data of data breach incidents from a combination of both the Data Privacy Clearinghouse dataset and the Datalossdb.org project. A combination of these two datasets included almost 10,000 records for hacking, web, and virus incidents from January of 2000 to February of 2015. These datasets have been regarded as a good source for the population of data breaches in the U.S. [30]. Most of these breaches, however, have no publicly available information about their characteristics. We thus decided to focus on the incidents with the most records breached and for which public information was available. After researching over 450 incidents, we obtained 25 incidents which have detailed descriptions available on public sources on the web. We also added to the dataset an additional 7 notable incidents which were not available in the data breach databases but have been extensively noted in the media and are technically advanced—such as Stuxnet, Duqu and the RSA attacks—for a total of 32 data points. We consider this dataset a representative of the most well-publicized security breaches in the past 10 years.

After identifying the incidents, we prepared scenario descriptions with a compilation of publicly available information from different sources (e.g. newspapers, technical reports, legal documents, academic literature) for each of the 32 scenarios. We created a questionnaire to present the 32 distinct scenarios to professionals working directly in information security. For each scenario description we asked the respondents: (1) to identify the features that make the incident more or less sophisticated, and (2) how sophisticated they deemed the incident to be on a scale from 1 to 7. We removed the count of records stolen and names of individuals or organizations involved in the breach to avoid response bias. We randomized the order of the scenarios and provided the questionnaire online. A total of 9 individual experts completed our survey. These individuals worked as faculty in academia researching information security issues and as information security specialists working for the public and private sector. Each individual had at least 5 years of experience working with information security. Ultimately, each scenario was evaluated by 3 distinct experts.

<table>
<thead>
<tr>
<th>TABLE I. Features Types of Sophistication Index</th>
</tr>
</thead>
<tbody>
<tr>
<td>Feature Type</td>
</tr>
<tr>
<td>---------------</td>
</tr>
<tr>
<td>Social Engineering</td>
</tr>
<tr>
<td>Remote Administration</td>
</tr>
<tr>
<td>Stealth</td>
</tr>
<tr>
<td>Zero-Day Exploit</td>
</tr>
<tr>
<td>APT</td>
</tr>
<tr>
<td>E.g.</td>
</tr>
<tr>
<td>E.g.</td>
</tr>
</tbody>
</table>

**B. Analytical Tools**

In order to answer some of our research questions, we compute the change in sophistication level over the years based on 3 distinct evaluations, the average evaluation and our sophistication index score. We also carry out a series of correlation tests to validate our measures. First we obtained a measure of reliability of the respondents via a Pearson correlation statistic. This Pearson correlation provides a good measure of inter-rater reliability as others (e.g. Cohen kappa)
do not account for ordinal and relative differences across respondents [31]. We also tested the correlation among our perceived sophistication measures and our sophistication index, as well as the correlation between sophistication level and the amount of records breached. For this latter analysis we removed the incidents which did not have a measure of records breached (e.g. Stuxnet).

IV. ANALYSIS AND RESULTS

Table 3 below lists the scenarios obtained and analyzed. The lowest score is for 2008-Comcast (2 for perceived sophistication and 0 for sophistication index) and the highest is for 2010-Stuxnet (7 for perceived sophistication, and 5 for sophistication index) in table 2 we present some of the characteristics of sophistication highlighted in the expert responses. The features for the more sophisticated attacks are represented by some of the following terms: Remote Administration Tool, zero-day vulnerability, and digital signature. Some of the features for the less sophisticated attacks are represented by the following terms: poor system design, no technical knowledge, SQL injection, brute force, and standard tools.

Figure 1 below shows the incidents plotted over time. The figure presents the 3 distinct evaluation scores for each incident in our dataset, along with the average score (dashed line) and our sophistication index score (lower dotted line). The inter-coder reliability for these 3 measures based on Pearson correlation is 0.63 ($p < 0.01$), which may be considered good [31]. The correlation for our sophistication index and the average perceived sophistication is 0.72 ($p < 0.01$) which may also be considered good [26]. A few incidents differed more than others in the evaluation across respondents. After reviewing their explanation we could observe that whereas some individuals focused on the failures of the organization, others highlighted the different set of components required for the attack to succeed. Also, some respondents focused on the larger set of infrastructure required to carry out the attack, whereas others focused on the attack method itself.

Table II. Characteristics of More and Less Sophistication Breaches

<table>
<thead>
<tr>
<th>More Sophisticated</th>
<th>Less Sophisticated</th>
</tr>
</thead>
<tbody>
<tr>
<td>Zero-day exploit</td>
<td>Insider access</td>
</tr>
<tr>
<td>Remote Administration Tool</td>
<td>Poor system design</td>
</tr>
<tr>
<td>Insight into target</td>
<td>No technical knowledge</td>
</tr>
<tr>
<td>Avoiding detection</td>
<td>Standard tools</td>
</tr>
<tr>
<td>Encrypted data</td>
<td>SQL injection</td>
</tr>
<tr>
<td>Digital signature</td>
<td>Brute force</td>
</tr>
</tbody>
</table>

Figure 1 also suggests that security breaches have not consistently increased in sophistication even though we are focused on the most well publicized attacks—thus potentially also most prominent and important attacks. Since sophistication and technological progress increases overtime, we assume that our current measure of previous attacks would be under-evaluated. By definition, we should have observed earlier attacks as being less sophisticated, which is not what we observe in figure 1.

![Fig. 1. Changes in sophistication level over time](image)

We also computed two other correlation tests. First we correlated average sophistication with year, resulting in a relatively low and positive coefficient ($r = .28$) but not significant ($p = .11$), indicating that sophistication and time are not significantly correlated. For 25 of the security breaches we had the number of records breached, which we may consider to be a type of damage impact. We computed a correlation coefficient for these data points based on the average sophistication level and the log of records breached, resulting in a negative correlation coefficient ($r = -0.29$) but insignificant ($p = .15$). The negative result would suggest that more sophisticated attacks would have less damage impact based on the amount of data breached. The low coefficient, however, and the insignificant probability may more simply
suggest that more sophisticated attacks are not necessarily the ones that will cause greater damage of records breached.

V. DISCUSSION AND LIMITATIONS

Proposing a basic set of feature types and feature terms that, when combined, signify the sophistication level of an attack is probably not entirely controversial. However, it may seem overly simplistic to some. We believe there is a trade-off between simplicity and accurateness. A more encompassing measure may be more accurate, but it is also more difficult to use in practice. Moreover, the observation of changes in level of sophistication over the past 10 years may not be intuitive. In order to understand this, we should note that sophistication is intrinsically related to technical progress. It is surprising therefore to observe that the well-publicized attacks over the years have not generally increased in sophistication or that most well publicized attacks still have low measures of sophistication. This could mean that our data set was perhaps not representative enough of the sophisticated and non-sophisticated attacks or that we are missing some features of attack sophistication in the metric. It may also mean that, potentially since Stuxnet, most attacks have leveled off in their degree of sophistication. Attacks that are well publicized, which is often the case by the damage they create, are not necessarily sophisticated. This was also demonstrated in our results of the lack of correlation between damage and sophistication.

One of the limitations of the study is that the data is based on publicly available information. Although we have used investigative reports from law enforcement and intelligence agencies when available, they were not available for all attack incidents. Therefore our information may have holes. Each scenario description contained basic information about the attack including method of intrusion and particular tools used. Nevertheless, certain attacks that have been evaluated as more or less sophisticated could be regarded otherwise given more information about the particular incident. The second limitation is the number of scenarios and the number and quality of experts evaluating the scenarios. The more the scenarios and the stronger the level of expertise in evaluators, the better would be our results.

One of the other major challenges with this work is the notion of relativity that is intrinsic to the concept of sophistication. Sophistication is a quality that changes over time and specific to the application in which it is employed. Therefore an attack that is precise in its manipulation of a system, and uses only a few resources and tools to carry out its goal, may be considered sophisticated if it achieves its aim. Conversely, an attack that uses multiple tools but are not successful may not be considered as such. Moreover, the conditions of the target itself play a role, as indicated by expert responses. In a data breach, poor system design of a target qualifies the incident as less sophisticated. However, in our measure we do not include characteristics of the target itself. Another issue is related to the estimation of damage impact. In our calculation of damage impact, data may not be always an accurate measure of damage. Certain data are more valuable than others, and some attacks may cause great damage without stealing any data (e.g. Stuxnet).

Despite these limitations, we believe this research begins to lay the foundation for further investigation of sophistication. As previously indicated we encountered no such attempt in creating a metric of sophistication per se. Although in this study information about each incident may not be complete, publicly available information is what most individuals would be presented with about a particular security breach. Therefore, potential consequences of a data breach based on the notion of sophistication, such as the financial impact on firms or the political impact on society would be based on the perception of the attack sophistication and not necessarily the objective level of sophistication or complexity. However, the use of number of records exposed to evaluate damage impact does provides an objective measure of damage impact and is suited for data breach incidents, which is the major focus and intended application of this framework.

VI. CONCLUSION AND FUTURE WORK

In this work we were interested in identifying the characteristics of sophisticated cyber-attacks to develop a sophistication index. Through a review of the literature we identified a variety of features that have been proposed to explain the evolution of malware and cyber-attacks. Based on this review and expert input we proposed an initial generalization of feature types and feature terms that may be used to classify and quantify sophistication of security breaches due to cyber-attacks, what we called the sophistication index. We obtained expert scores on their own evaluation of perceived sophistication and correlated them with our sophistication index in order to validate it. There is a good level of reliability on how experts evaluated the cyber-attacks among themselves and a strong correlation between our sophistication index and the average perceived sophistication. We also found that the incidents in our data sets have not consistently increased in sophistication during the past 10 years, which may indicate a leveling off in the general increase of cyber-attack sophistication or that the level of sophistication is not a predictor of attack success or attack damage.

Based on our findings we suggest that future work may examine how to extract feature terms from the publically available media, or others sources, and automatically classify the sophistication index of a cyber-attack via machine learning and natural language processing techniques. In this study we also present the concept of perceived sophistication. Although we used perceived sophistication to validate a general index measure we propose that future work may further examine issues present in perceived sophistication itself. For example, future work may explore how the concept of sophistication is used by the media in order to manipulate the perception of the importance or the culpability of an attack. Although reports mention that an attack is sophisticated, an explanation of the actual characteristics may show otherwise. The potential for using a text base analysis of incidents via automatic means would also assist in this examination of media reports and framing. Ultimately, we expect future research to test our sophistication index in a larger dataset and provide suggestions on how to improve our metric. Improvements to the metric may be based on objective characteristics of an attack or on related characteristics such as identifying attack motivation and attack source.
REFERENCES

Information Sharing to Manage Cyber Incidents:
Deluged with Data Yet Starving for Timely Critical Data

Sanjay Goel  
School of Business  
University at Albany, SUNY  
Albany, New York

Charles Barry  
National Defense University  
Washington, DC

Cyber security incidents that degrade user security and access to the Internet are often international in nature, simply because of the distributed borderless nature of Internet traffic flows. Effective incident response management involves detection of the breach, blocking the perpetrators from damage or theft, and identifying those responsible for prosecution. Defense against cyber threats requires the ability not only to detect breaches, but also to contain them quickly and remedy the vulnerabilities that were exploited. The robust exchange of threat and vulnerability information across organizations can improve collective national security. Development of situational awareness capability during incidents requires the collection of data from log files, both on the victim organization’s and the attackers’ servers, through Internet Service Providers that may not even be in the same country. Sharing of relevant data among responders is important both for broad participation and timeliness. No less important is close cooperation among public and private-sector actors; those essential to successful response, but also those who may become the future victims of cyber incidents, both systemic and malicious. Success means ameliorating the effects of causal factors as rapidly as possible, and restoring critical systems to full operation. Information sharing is a monumental task encumbered by administrative morass, and, often by our inability to pinpoint the precise data needed.

This paper explores the current methods of international information sharing across both private and public-sector organizations. It examines what information is critical to be shared initially, and then on a continuing basis during incident response. The paper provides a critical assessment of the hurdles still to be overcome before broad and timely sharing of information can become a reality. Finally, it offers options for improving the identification of what characterizes essential data within the overwhelming data volume of the Internet, and methods for sharing such data among responders. The salient conclusions are that: cooperation in cyber incident response, while well recognized, is in its infancy in terms of identification of critical data to be shared; effective information sharing is the rare exception rather than a well-established practice; and the primary hurdles remaining are human factors not technological inadequacy. The long-term strategy for information sharing requires streamlining the administrative process, standardizing formats for data exchange, and creating specific data collection requirements for organizations as well as Internet operators. The paper concludes with recommendations for future of information sharing among organizations.

The full version will be made available online through the ASIA website at: http://www.albany.edu/iasymposium
Network Defense Exercises with Simulated Attacks

Delbert Hart
Computer Science Department
SUNY Plattsburgh
Plattsburgh, NY 12901
Email: hartdr@plattsburgh.edu

Abstract—This paper describes a new network defense learning environment, simzero. The novel approach this system takes is to create simulated vulnerabilities for students to practice with. The use of simulated vulnerabilities eliminates the possibility of malware accidentally spreading to other machines and networks. This system provides students with the opportunity to gain network defense experience in a realistic environment, without the risk associated with using actual malware.

Keywords—Security and Forensics Education, Hands-on Security Laboratories and Table Top Exercises

I. INTRODUCTION

Hands-on network defense exercises are very valuable to students. Creating realistic environments for these exercises though has significant challenges and risks. The system described in this paper addresses the following challenges:

- **Risk:** Using actual exploits creates a potential risk to the campus's and students' systems. Traditional environments must take special precautions to keep the educational activity separate from other systems and networks. A common approach to mitigate this risk is to use virtualization, e.g., [1], [2].

- **Distraction:** Network defense and vulnerability analysis and exploitation are separate skills. In competitive network defense exercises (e.g., [3] [4]), most of the student's time is spent on creating and deploying exploits. It would be useful to be able to offer attack and defense scenarios that mostly focused on network intrusion defense skills.

- **Overhead:** There is significant overhead in setting up, maintaining, monitoring, and evaluating full network attack and defense scenarios [5]. The overhead arises from the complexity of designing and deploying new scenarios, and ensuring that safety is maintained. This overhead limits the number and range of scenarios that can be offered. Tools exist to aid in these tasks, but there is further room for improvement.

The work described in this paper contributes to the field of computer security education, by providing a new approach for creating network defense exercises. The key insight is that safety, configurability, and pedagogical flexibility can be improved by using simulated vulnerabilities rather than real vulnerabilities. The next section reviews related work in the field. The third section describes how the simulated attacks are modeled. The simzero environment’s implementation is given in section 4, and a case study is presented in section 5.

II. RELATED WORK

A. Network Attack and Defense Exercises

Simulated environments play an important role in network defense education. One starting point to learn about network defense is to review/experiment with packet capture files illustrating different types of traffic. These pcap files are widely available (e.g., [6]) and can be a useful resource to practice with different tools. The main drawback of pcap files is their static nature.

Another jumping off point, which is more interactive, is working with virtual machines, pre-installed with known vulnerabilities. (One collection of them can be found at [7].) While virtual machines allow students to experiment, they do not provide sufficient context to create a realistic network defense scenario.

There has been significant effort in creating realistic, educational security labs, e.g., [1] [2] [8] [9]. The three principle challenges are to create environments that are safe, realistic, and configurable. Virtualization is a key technology used in these lab management tools.

The NEMESIS framework [5] is a representative example of this type of work. NEMESIS is a scenario generation tool used to simplify the deployment of a virtual network of machines for an exercise environment on top of a set of physical machines. A central Management Agent reads the XML configuration file and contacts Deployment Agents running on the available physical machines. The configuration file describes the virtual network topology, the set of virtual machines to be deployed, and the configuration of those machines. The Deployment Agents are responsible for starting and stopping the virtual machines and applying the configuration options to the running machine instances.

Tools like NEMESIS help create/manager the network topology and machines within the virtual network. simzero provides a complementary approach to these lab management tools. Instead of managing the vulnerabilities by managing the machines and services installed on the machines, simzero virtualizes the vulnerabilities directly. simzero can be used in conjunction with these tools to increase the safety and flexibility of the overall system.

B. Security Contests

Security contests are a popular venue for students to compete with each other and demonstrate the skills that they...
have learned. The most popular types of contests are: 1) task-based and 2) network attack and defense. An example of a task based contest is NYU Cyber Security Awareness Week Capture the Flag contest [10], where a number of challenges are available for students to complete. The challenges vary in difficulty, with more difficult challenges being worth more points.

Network attack and defense competitions attempt to provide a realistic environment, within which students play the role of both attacker and defender. A well known example of this type is DefCon’s annual Capture the Flag [3] contest. In this contest each team is responsible for defending their machine against attacks, while ensuring that the services on the machine remain functioning normally. The environment is created with vulnerabilities in the services, and the teams work on discovering those vulnerabilities and exploiting them to score points against the other teams. One challenge in putting on a competition of this sort is getting the vulnerability difficult correct. The difficulty of exploiting the vulnerabilities has to be just right to have a successful contest. The National Collegiate Cyber Defense Competition [4] approach to getting the vulnerability difficulty correct is to delegate it to the teams. Each team embeds a vulnerability in a service. Then the teams work to discover and exploit each other’s vulnerabilities.

Often these contests are long, lasting all day or multiple days. This is necessitated by the work needed to identify vulnerabilities and design exploits for them. One goal of the simzero system is to enable network defense contests that are shorter and more focused on the defense aspect.

C. Other Simulations

Another use of network simulation is to automatically evaluate network defense tools. The main idea is to automate both attacks and defense to quickly evaluate a range of scenarios. One example of this type of system is [11], where an automated attack tool was created to evaluate the effectiveness of different intrusion detection systems. Attacks are generated based on recorded data and stochastic attack models. A different approach to attack generation was taken in [12], which created an ontology of network attacks, which was then used to automatically generate attack sequences. The attacks are generated based on what the attacker knows thus far about the victim, the attacker’s intent, and the defender’s response. These types of tools are useful for improving security tools, but not flexible enough for educational purposes.

D. Attack Taxonomies

Simzero’s key innovation is using simulated vulnerabilities rather than real ones for network defense exercises. The literature on computer attack taxonomies was reviewed as an initial step in designing how vulnerabilities should be simulated. The first general purpose attack taxonomy was created by Howard and Longstaff [13] and had 10 dimensions. This taxonomy viewed a computer security incident as an “attack” performed by “attackers” who had an “objective”. The attack itself consisted of a computer security “event”, performed by a “tool”, exploiting a “vulnerability”, with an “unauthorized result”. The computer security event was a specific type of “action” against a “target”.

Over the years, there have been many different taxonomies of computer attacks. A survey of over 30 taxonomies is provided in [14]. This survey identified four of the most frequently used dimensions as:

- **source** - Where did the attack come from?
- **target** - What subsystem or application was exploited?
- **vulnerability** - What was the nature of the vulnerability exploited?
- **impact** - How did the exploit affect the system’s security?

The proliferation of attack taxonomies has several causes. One reason is that as computing expands into new areas, specialized attack taxonomies are needed (e.g., a wireless attack taxonomy [15], distributed denial of service taxonomy [16], etc).

The other two major factors behind the plethora of taxonomies is lack of agreement about which dimensions are relevant and what the valid values within a dimension are. An illustration of these factors can be seen by considering two recent, relatively similar taxonomies: the ADMIT taxonomy [17] and the AVOIDIT taxonomy [18]. While they share many of the same dimensions, they differ in others. The ADMIT taxonomy classifies the person behind the attack, whereas, the AVOIDIT taxonomy chooses to include a dimension describing the types of defenses that could be used against the attack. Even when taxonomies agree on which dimensions should be used, they often differ on the values to be used within that dimension, as seen by considering ADMIT’s and AVOIDIT’s “attack vector” dimension.

Although existing attack taxonomies provide some guidance in designing simulated attacks, there are significant differences also. Existing attack taxonomies are oriented to understanding and communicating about actual exploits. The goal of the simzero system is to support network defense exercises. Focusing on network intrusion detection, only the packets being transferred between the attacker and victim are visible. This renders a number of dimensions in most taxonomies moot. In particular, the attackers, their motivation, the tools they are using, and the specific vulnerabilities being exploited are unknowable/irrelevant. Unauthorized actions are also seen differently from a purely network intrusion detection perspective. Changes that remain within the host, such as destroying or modifying data, are practically invisible to the network. Others types of misuse, such as information disclosure or denial of service, may be detectable by the network.

E. Attack Languages

Once simulated vulnerabilities have been designed it is necessary to decide how the exploits will be encoded. Three related families of languages are event languages, exploit languages, and detection languages.

One way to encode an attack is to use an event language to describe its sequence of events. The most commonly used event language is the pcap format [19]. pcap files fully describe network packets as they arrive at a machine. A related format is IPFIX [20], which describes packet flows. Event languages are
too low level to offer enough flexibility in creating simulated vulnerabilities and exploits.

Exploit languages are domain specific languages used to implement attacks. The Nessus Attack Scripting Language [21] and Metasploit’s meterpreter [22] both offer convenient ways to script an attack. Exploit languages are needed in the simzero system because no real vulnerabilities are being exploited.

Detection languages are used to attempt to identify attacks against a system. Students write rules for systems such as Snort [23] or Bro [24] to implement network defense. Ideally the rules that students write to detect the simulated attacks would be almost identical to the rules written to detect actual attacks.

III. ATTACK MODELING

A. Definitions

An attack consists of one or messages that causes a misuse action to occur on the victim’s machine. A misuse action is a change made directly or indirectly by the attacker that violates the system’s security policies. An independent attack, is one whose success is not dependent on any other attacks against the machine. A coordinated attack is a set of attacks, each with their own misuse actions, where the success of some of the attacks is dependent upon the success of previous attacks.

B. Attack Simulation

Consider how a network attack causes a misuse action. An attack message is delivered to the target service. Some vulnerability within the target service reacts to the message in a way that causes some sort of misuse to occur. The attack surface is the union of all vulnerabilities exposed by the host’s services.

Formally, there is a set of attack messages \( A_i \) (against a vulnerability \( v_i \)) which is a subset of all possible messages \( \mathcal{M} \). Whether a given message is an attack message is decided by the indicator function \( I_{A_i} \).

\[
I_{A_i} : \mathcal{M} \rightarrow \{0, 1\}
\]

A host \( h \) consists of a set of target services exposing a set of vulnerabilities, \( v_1...v_n \). Thus the set of attack messages, \( A \), against a host is defined as:

\[
A = \bigcup_{i=1}^{n} A_i
\]

In practice, \( I_A \) is implemented collectively (and inadvertently) by the target services running on the host. Network defenders attempt to approximate \( I_A \) with the indicator function \( I_D \) which is defined via the intrusion detection system. False negatives are described by \( \{A \cap D\} \) and false positives by \( \mathcal{N} \cap \{D \cap A\} \), where \( \mathcal{N} \) is the set of non attack messages received.

Successful attacks occur when an attacker is aware of members of the set \( \{A \cap D\} \). A zero-day exploit occurs when an attacker discover new members of the set \( \{A \cap D\} \).

In a typical educational environment the set \( A \) is explicitly managed by installing known vulnerabilities in the running services. Students then work to properly define \( D \) by analyzing the traffic traversing the network. The main challenge with this approach is that the set of attack messages is a derived construct. The set \( A \) can not be modified directly, instead the environment has to undergo significant modification.

The approach taken by the simzero system is to redefine what an attack message is. Instead of using \( I_A \), the simzero system explicitly defines a new set of simulated vulnerabilities. A simulated vulnerability, \( sv_i \) has an associated set of messages \( S_i \) that are considered attack messages. Together, all of the simulated attack messages form the set \( S \). To modify the set of simulated attack messages it is only necessary to change a single configuration file. Changes to the set of simulated exploits can be made easily by the system at any time.

C. Attack Properties

For realism purposes, a set \( S_i \) of simulated attack messages should be indistinguishable from the set of attack messages \( A_i \) associated with an actual vulnerability. While messages in \( A_i \) are determined by the vulnerability, messages in \( S_i \) can be chosen by the system. In addition to realism, the simulated attack messages should be able to represent different classes of attacks and be configurable in how difficult the are to detect.

simzero considers eight different aspects of an attack in specifying simulated vulnerabilities, four properties of the vulnerability and four properties of the attack messages. Notice that some commonly used properties (attacker, vulnerability type, motivation, etc.) are not relevant to the system’s ability to simulate attacks.

1) Vulnerability Properties: simzero does not attempt to simulate specific types of vulnerabilities. From the network defense perspective, the specific types of exploits that occur look very similar. For instance, both buffer overflow attacks and SQL-injection attacks consist of data that does not meet the vulnerable server’s expectations. Without using information from deep packet inspection, these attacks would appear virtually identical at the network level. For simzero, the vulnerabilities properties that are pertinent are:

- **resource**: A resource is a data store that can be the target of misuse. The specific resource being attacked may be implicitly determined by the vulnerability being exploited, or it may be passed in as a parameter.
Some taxonomies distinguish between data, processes, subsystems, user accounts, etc. In this paper all resources are viewed as data stores. An attack that infects an application is a modification of the executable on the hard disk. An attack that infects a running process is an attack against the operating systems’ data structures.

- **action**: Part of each simulated vulnerability is a high level script that performs the misuse action. One of the goals of the simzero system is to focus on network defense education, hence the payload actions are constrained based on the vulnerability. This also allows the instructor to adjust the difficulty level of misuse detection.

- **expressiveness**: is the degree of control the attacker has over the misuse actions performed. The Ping O’ Death [25] only allowed the attacker to perform a denial of service attack. The ShellShock exploit [26] allowed the attacker to run arbitrary code.

While the misuse action will be determined by the simulated vulnerability, the number and range of parameters that will be accepted can be adjusted to modify the difficulty level.

- **environment**: The effectiveness of some attacks depends on factors outside the control of the attacker. For instance, the HeartBleed vulnerability [27] allowed an attacker to retrieve data from the web server’s memory, but without other actions, the disclosed data may not have been useful.

Configuring the environmental dependencies of a simulated vulnerability allows the instructor to adjust the difficulty level of the exploit, or to use it as a mechanism to require a coordinated attack.

2) **Message Properties**:

- **[target] service**: Services often use their own application specific protocols. This allows the service to be inferred by inspecting the packets bound for it. Services typically default to predetermined port numbers to make it easier to advertise the service’s availability.

Many taxonomies use the target dimension to identify the subsystem/process that is being exploited. The term “service” or “target service” is used in this paper because of the generality of the word target. A service in simzero is a process that receives packets from the network.

- **size**: The size of attack messages is configured to keep the simulated attacks similar to actual attacks and to adjust the level of difficulty for network defenders.

- **staging**: Multistage simulated attacks are designed to mimic actual multistage attacks. The difference between a multistage independent attack and a coordinated attack is in the number of misuse actions taken. A multistage independent attack performs one instance of misuse. A coordinated attack has a misuse action associated with each component attack.

- **normalcy**: The degree of similarity between the attack traffic and normal traffic sent to the service. A number of different metrics can be used to capture this. One simple metric is to compute the entropy of the message using the set of historical messages to the service.

The normalcy of attack messages is important, because messages that are too abnormal will be easily flagged by anomaly detection systems. Adjusting this property directly affects how difficult it is for network defenders to detect the attack.

### IV. Simzero

Simzero’s purpose is to support students practicing network defense in a realistic environment. A session of simzero starts the necessary daemons on the machines participating in the session. A session will have at least one participant who will be practicing network defense. A contest is a session with multiple participants who also take on the role of using the simulated attacks against other participants to score points. An exercise is a session where the attacks are automated and the participants only defend. The machines associated with a session are referred to as session hosts.

Our design goals for the simzero system were safety, realism, and configurability.

- **Safety** - is provided by using simulated vulnerabilities rather than actual software vulnerabilities. This ensures that there is no chance of the attacks infecting systems outside of the simzero environment. Participants do use actual reconnaissance attacks and tools. To protect outside networks, sessions are run on a (physically or virtually) isolated network.

- **Realism** - is provided by having the characteristics of the simulated attacks resemble actual exploits. Participants also make use of current tools for network defense and reconnaissance.

- **Configurability** - is achieved through a configuration file that describes the number of vulnerabilities to be generated and their properties. The configuration file also specifies how knowledge of the vulnerabilities is distributed amongst the participants. During the contest the simulated vulnerabilities installed on each machine can be updated without impacting the operation of the services. Different scenarios can be implemented without modifying the set of virtual machines or network topology.

#### A. Architecture

To simulate attacks the system must be able to intercept the simulated attack messages and then act upon them. The best place to intercept the messages is between the operating system and the application. This is achieved in the Linux operating system by making use of the netfilter architecture [28]. The message flow is show in Figure 2. Network traffic, including simulated attacks, arrive at the machine. The traffic passes through the protocol stack, including iptables/netfilter. Netfilter is an extensible packet filtering mechanism included with Linux systems. Netfilter provides a hook for the simzero system to intercept network packets before they are delivered to the application. This simzero filter scans for simulated attacks. Normal traffic is allowed to pass through the filter with
minimal delay. When a simulated attack is found, then code to simulate the effects of the attack is invoked. By positioning the simzero filter between iptables and the applications it is effectively invisible to network defense tools and the applications.

The simzero filter is a daemon installed on each participant’s machine to recognize simulated attacks made against that machine. Upon startup it registers to receive packets from a netfilter queue. The filter keeps a list of patterns corresponding to the simulated vulnerabilities that have been installed. Each vulnerability has an associated misuse action that should be invoked when the simulated attack arrives. Trivial misuse actions are handled within the filter, more complex ones are forwarded to a specialized exploit server that processes the request. The filter also listens for control messages used to update the list of simulated vulnerabilities.

After the filter is done processing a packet it can drop the packet, or allow it to continue to the service it was destined for. In this way it is possible to simulate an attack against a service without the service being modified. Consider a simulated vulnerability in a web server. A simulated attack message arrives embedded in an http request. The filter recognizes the attack and triggers the corresponding misuse action. The original packet is then allowed to continue to the web server.

Exploit servers are programs or daemons installed on the participants’ machines to implement the simulated misuse actions. The exploit servers are separate from the simzero filter to minimize the amount of work done in the filter, which in turn reduces the impact the filter has on traffic. Each exploit server implements a different class of attacks.

The session manager sets up the simulated attack environment. It reads the session configuration file, and then contacts the simzero filters to “install” the vulnerabilities.

V. EXFILTRATION NETWORK DEFENSE CONTEST

The simzero system consists of a set of Perl scripts that intercept network packets, provide scaffolding for a network defense contest, and simulate vulnerabilities in the services on the system. Since simzero’s packet interception utilizes the netfilter architecture, the participating systems must be running the Linux operating system.

The network defense contest described in this section was used with the intrusion detection course that motivated the development of this system. In the contest 5-8 teams of 2-3 students attempted to smuggle data out of each other’s machines using simulated vulnerabilities. Teams received points for successful attacks and successful defenses. Three contests were held, each lasting between 60-75 minutes.

A. Setup

The contest network was physically isolated from the campus network. Each team had a virtual machine configured with a version of Linux and the simzero software pre-installed. To simplify setup and monitoring of the contest machines, they were configured with ssh public-key access. The session manager machine had the private key, and thus root access on each of the contest machines. First the session manager connected to each of the contest machines to start simulatedzero’s daemons and install the data files that were to be protected.

Figure 3 illustrates the next steps the session manager took to set up the simulated vulnerabilities. First, the session manager created the vulnerabilities. Then the simulated vulnerabilities were installed on each machine by the session manager by contacting their simzero filters.

B. Vulnerabilities

Immediately before the start of the first round, each team was given recipes that described how to create simulated exploits. During the contest each team had a limited pool
of exploits to use. The teams did not know which exploits were installed on which machines. Some of the exploits were installed on their own machine, in which case they could use the recipe knowledge to design a signature based defense against that exploit. The distribution of exploits was configured to have some overlap between the exploits known by the different teams. For any pair of teams, roughly 1 out of 10 exploits would be known by both.

The configuration file had 10 services that the participant machines must support. The generated vulnerabilities were randomly assigned to the different services. All of vulnerabilities were installed before the start of the contest.

C. Contest

The contests had a number of rounds. Each round has two phases, a preparation phase where the teams set up their defenses and attack scripts, and an active phase during which the attacks occurred. During the first round’s active phase the teams also performed reconnaissance attacks to learn the identities of the machines on the network. In addition to the machines the teams were defending and using as a base for attacks, were a set of decoy machines. The decoy machines made requests against the participant’s machines to verify that all of the services were available. Decoy machines also had vulnerabilities installed and could be targeted by the teams to score points.

After a round ended, each team received a small number of additional exploits that could be used, and some additional targets for exfiltration. Exploits from previous rounds were still valid, but were more likely to be defended against. Teams had an incentive to be careful with their use of an exploit.

At the end of the contest teams scored points for the target files exfiltrated. Team also received points for recognizing attacks, even if they were too late to stop it, and points for preventing attacks.

VI. Summary

Gaining experience with network defense tools in a realistic environment is challenging due to the overhead of creating such environments in a way that does not threaten production networks. This paper described an approach for creating network defense training exercises with minimal risk to existing networks. A system, simzero, that implements this approach is also described.

simzero provides a flexible and safe environment for network attack and defense exercises. The use of simulated vulnerabilities provides a new means of reducing the risk associated with supporting these learning opportunities. The simulated vulnerabilities are tunable to provide different levels of difficulty. Another advantage of this approach is the ability provide exercises that focus almost exclusively on network defense skills. Participants do not have to be thoroughly versed with the exploits being used or how to write them, they just focus on how to identify threats as they traverse the network.

REFERENCES


Reducing Cyber Risk through Diplomacy

Michele G. Markoff
Deputy Coordinator for Cyber Issues at the U.S. Department of State

We all have a part to play in making our networks more secure, but the global nature of cyber threats means that there is a unique role for diplomacy in reducing risk. Michele Markoff, Deputy Coordinator for Cyber Issues at the U.S. Department of State, will describe how the United States is working internationally to promote greater stability in cyberspace. Her talk will touch on efforts to build consensus among governments on principles of responsible state behavior in cyberspace, to develop practical confidence building measures to reduce the risk of conflict, and improve global cybersecurity.

BIOGRAPHY

Michele Markoff is Deputy Coordinator for Cyber Issues in the Office of the Coordinator for Cyber Affairs. Since 1998 Michele has been the senior State Department subject matter expert overseeing the development and implementation of foreign policy initiatives on cyberspace issues. She helps to coordinate United States policy on the spectrum of cyber-related policy issues across the Department, develops diplomatic strategies to encourage states to join the United States in taking steps to protect their critical networks and to cooperate internationally to enhance and preserve global cyber stability. She implements those strategies through negotiations in a wide variety of venues. Her initiative led to the successful completion of the first ever bilateral agreement on confidence-building in cyberspace between the United State and the Russian Federation, announced in June, 2013.

Michele also has been the United States Government Expert on three Groups of UN Government Experts (2005, 2010, and 2013) devoted to cyber issues. The last two led to landmark consensus reports regarding norms for state activity in cyberspace. Ms. Markoff was trained as an expert in Russian and Chinese military affairs and decision-making and spent the first half of her career in a variety of strategic nuclear arms control-related posts, among them as State Department Advisor and then Executive Secretary to the START I Talks; later as Senior Policy Advisor and Director of the U.S. Arms Control and Disarmament Agency’s Policy Planning Group.

Ms. Markoff has a B.A. in International Relations from Reed College, an M.A. in International Relations and an M.Phil. in Political Science from Yale University, and a M.Sc. in National Security Strategy from the National War College of the United States. She also attended the Chinese University of Hong Kong.
Lessons for a More Defensible Cyberspace

Jason Healey
Senior Research Scholar at Columbia University and Senior Fellow of the Atlantic Council

This talk will highlight recent results from the New York Cyber Task Force (NYCTF) that is exploring how to make cyberspace more defensible. Since at least the late 1970s, cyber attackers have had the advantage over defenders, for any number of reasons. Unless we can reverse (or at least slow) this underlying dynamic, defenders will be doomed to, at best, a never-ending cacophony of ever-worse incidents. The NYCTF has explored what a more defensible cyberspace would look like, technologies that have to date made cyberspace more defensible and least cost, and begun to discuss what policy and operational measures (like information sharing) have made a similar difference. We have identified a number key technologies that have worked at such scale that they have given defenses a huge boost at just a modest cost, as well as future technologies that might give similar payoff at scale.

BIOGRAPHY

Jason Healey is a Senior Research Scholar at Columbia University’s School for International and Public Affairs specializing in cyber conflict, competition and cooperation. Prior to this, he was the founding director of the Cyber Statecraft Initiative of the Atlantic Council where he remains a Senior Fellow. He has authored dozens of published articles and is the editor of the first history of conflict in cyberspace, A Fierce Domain: Cyber Conflict, 1986 to 2012. During his time in the White House, he was a director for cyber policy, coordinating efforts to secure US cyberspace and critical infrastructure.

At Goldman Sachs, he created their first cyber incident response team and later oversaw the bank’s crisis management and business continuity in Asia. He started his career as a US Air Force intelligence officer with jobs at the Pentagon and National Security Agency and is president of the Cyber Conflict Studies Association.
Toward Cognitive Immunization of Potential Criminals against Cyberterrorism

Merrill Warkentin
Department of Management & Information Systems
Mississippi State University
MS STATE, MS 39762 USA
m.warkentin@msstate.edu

Joshua M. Regan
Henry C. Lee College of Criminal Justice
University of New Haven
West Haven, CT 06516 USA
jrega1@unh.newhaven.edu

Amin G. Kosseim
Community Affairs Division
New York City Police Department
New York, NY 10128 USA
agkosseim@gmail.com

Abstract—Most research into preventing cyberterrorism, whether by foreign or domestic individuals, has focused on technical measures to detect or prevent the attacks. Such efforts presume that individuals are already radicalized and that events are inevitable. Our investigations look further back; we seek to understand and influence events which occur long before the cyberterrorism attacks are formulated and perpetrated, when individuals experience psychological processes – both cognitive and affective – that lead them to the intentions to engage in cyberterrorism. Our focus, inspired by the “Left of Bang” paradigm, looks at three principle root causes – (1) techniques of neutralization, (2) expressive and instrumental crimes, and (3) perceptions of injustice and disgruntlement. We propose further research into these important factors that temporally precede the focal phenomenon.

Keywords—cyberterrorism, deterrence, criminology, techniques of neutralization, expressive crimes, instrumental crimes, disgruntlement

I. INTRODUCTION

Cyberterrorism as the combination of cyberspace and terrorism is defined as the use of intentional violence against IT systems that support the health of human communities and the information stored in such systems [1]. This type of terrorism is geared toward coercing the targeted people or government to behave in a certain way (influential cyberterrorism) or is designed to inflict damage or revenge (destructive cyberterrorism). Furthermore, it is typically more comprehensive and ruinous than cybercrime in that it harms the health of human communities or threatens such a harm. Reports show that cyberterrorism is currently the fastest growing threat to individuals in the United States has now exceeded illegal drug trafficking [2]. New York Senate digital infrastructure is being attacked on a daily basis (Ibid), indicating that it is crucial to identify and implement protection from these vicious attacks. In this regard, terrorist institutions are actively pursuing the recruitment of young people from Western countries, including and especially those with high-level computer skills. The focus of this research is to understand the factors that make the young generation susceptible and likely to be open to terrorists’ recruitments and to identify the ways to neutralize this susceptibility. In other words, investigation of the methods to cognitively immunize young generations from being lured into cyberterrorism activities is central to this work. We draw on Willison & Warkentin [3] concept of “left of bang” in order to understand the mechanisms of countering the emergence of malicious cyberterrorism intentions.

II. ANTECEDENTS OF CRIME & CYBERTERRORISM

Information Security literature has extensively investigated the phenomenon of security violations. Based on the abundance of new and emerging security threats, Willison & Warkentin (2013) called for new perspectives and theoretical lenses that not only include the criminal act and its immediate antecedents of intention to commit the abuses and deterrence crimes, but also the factors that temporally precede these issues. They assert the need to consider the cognition processes of the “potential offenders” and how these are impacted by the organizational and societal contexts prior to deterrence. The interplay between cognitive processes and these contexts can substantially influence the effectiveness of deterrence safeguards.

Drawing on theories in criminology, the Willison and Warkentin [4] paradigm extends the Straub & Welke [5] Security Action Cycle, and proposes three primary recommendations for future research into the fundamental antecedents of deviant cybersecurity behaviors, including insider computer abuse (security policy violation, whether malicious or benign in intent), reckless computer security hygiene behaviors by individuals, and malicious hacking activities including cyberterrorism against organizational targets (companies, governments, etc.) or against society at large. These three research focus areas are (1) the utilization of neutralization (rationalization) techniques, (2) assessment of both expressive and instrumental criminal motivations, and (3) the role of disgruntlement as a result of perceptions of injustice (either organizational or societal). Each of these has been
applied to the study of organizational insiders (employees) who commit acts of insider computer abuse; we apply them to gain a more holistic view of cyberterrorism by individuals, whether they have been recruited or were “self-radicalized.” This manuscript represents the initial investigation into the phenomenon of cyberterrorism, especially home-grown domestic cyberterrorism, and strategies for intervening “left of bang” to deter and prevent such cyberterrorism acts.

Traditional criminologists infer that deviant behavior can be explained using two facets: human nature, i.e. genetics and brain development [6] and environmental conditions, i.e. residential location and its local supporting institutions [7][8]. We believe that ecological dynamics – including religious and educational influences, peer pressures, insufficient economic opportunities, and institutions – offer greater explanatory power when studying cyberterrorism. Many theories related to crime and deviant behavior rely on rational choice theory, which suggests that individuals are guided by a rational cognitive assessment of the relative risks and rewards for committing an act that is understood to be a crime, policy violation, or behavior that is counter to social norms. The assumption is that we are motivated to avoid negative consequences of our actions, such as punishment from sanctions, unless the potential gains are great enough. Several studies, however, have shown that rationality may not adequately explain real-world decisions. Decision makers have repeatedly been shown to violate the tenets of expected utility in making risk decisions based on framing effects [9][10][11][12]. Tversky and Kahneman [12] show that risk decisions are situational. Further, in the cybersecurity context, individuals are influenced by their social context [13].

III. TECHNIQUES OF NEUTRALIZATION

The first research focus recommended by Willison and Warkentin [4] is investigation into the role of the techniques of neutralization. Deviant behavior, including violation of organizational information security policies, is characterized as actions that members of a social group judge to be a violation of their shared rules, values, or accepted conduct. When contemplating such behaviors, most individuals are normally dissuaded by feelings of guilt and shame. However, Sykes and Matza [14] showed how offenders who might otherwise feel guilt and shame are able to neutralize these feelings by justifying or rationalizing their behaviors before committing the deviant act. These “techniques of neutralization” are processes that serve to reduce or eliminate the influence of internal norms and social censure, thereby deflecting the disapproval they would otherwise experience from others in the social environment and protecting the violator from feelings of self-blame, which enables him to engage in the deviant act. Sykes and Matza [14] identified five such techniques, which include denial of responsibility, denial of injury, denial of the victim, condemnation of the condemners, and the appeal to higher loyalties. Later researchers added many more techniques. Siponen and Vance [15] empirically established the role of techniques of neutralization in enabling computer-related violations, and Barlow, et al. [16] empirically showed that the use of such techniques (in the cybersecurity context) can be lessened or eliminated through the effective application of proper training and message framing.

IV. INSTRUMENTAL AND EXPRESSIVE CRIMES

In criminology literature, the difference between instrumental and expressive crime is extensively explicated. Instrumental crimes are conducted in order to gain explicit future goals such as acquiring financial gains. On the other hand, expressive crimes are characterized by unplanned acts of anger and frustration [17]. This distinction usually parallels the differences between premeditated and spontaneous crimes. Accordingly, criminologists consider instrumental acts as qualitatively different from expressive ones. This distinction is also an important one in recognizing the typologies of vehicle theft, vandalism, terrorism, violence in workplace and so forth [18][19]. It is reasonable to argue that the prevalence and nature of instrumental and expressive crime has significant implications for policies. According to deterrence theory, the threat of legal sanctions is the most useful leverage for instrumental crimes by individuals with low commitment to a criminal lifestyle [20][21]. In contrast, expressive crimes are regarded as “undeterrable” actions because the perpetrators do not thoughtfully evaluate options and choices (rational cognitive process), but are influence more by emotional factors. Furthermore, expressive computer crimes are substantially more undeterrable in cases where companies are reluctant to bring in law enforcement agencies.

Though instrumental and expressive crimes can be contrasted in terms of motivation and process, the dominant theory in behavioral cybersecurity area – Deterrence Theory – does not differentiate between these two types. If a crime is expressive, then analyzing a computer abuse through the lens of deterrence theory may not offer adequate insight, and research results may reflect this, leading to a flawed assumption. Therefore, utilizing security countermeasures without recognizing this distinction could result in ineffective IT safeguards.

V. ROLE OF INJUSTICE AND DISGRUNTLEMENT

The final topic into which Willison and Warkentin [4] call for research is the role of disgruntlement. As a “far left” of bang cause of criminal and deviant behaviors, including acts of cyberterrorism, disgruntled individuals – whether they are
employees or citizens – represent a significant cause of concern to those who wish to maintain peace and order. When an individual feels that he or she has been treated unfairly, that person will often attach an emotional response to the perceived source of this unfairness. In the organizational context, the research foundation is the study of “perceived organizational justice” (and injustice), and has been found in numerous studies to cause various negative outcomes, including sabotage against an employer. In the context of society at large, research has indicated that certain individuals, especially within key minority groups such as immigrant communities who may experience reduced economic or educational opportunities, will “blame the system” (and its perceived key representatives, such as law enforcement) and develop a deep-seated sense of disgruntlement. Such individuals often form intentions to lash out against the source of the perceived injustice, whether an employer or society at large. This can be out of retribution or out of a sense of “righting a wrong.”

Cyberterrorists may be fully integrated with a group of terrorists (especially when operating in a foreign country) or may be so-called “lone wolf” terrorists (often when they are self-radicalized and living within the target country domestically). In the former case, there may be cultural influences, social pressures, and reinforcing processes that lead to (and “reward”) the commission of acts of cyberterrorism. Families, friends, schools, and terrorist organizations may all support and encourage such individual in their home country. However, in the case of a terrorist operating in another country, the inability to undergo the acculturation and assimilation process successfully that would enable them to become fully productive members of their “host” society often leaves them vulnerable and susceptible to the influences of radicalizing forces [22]. Such individuals often fail to enjoy the economic benefits of opportunities in their host countries; they may be unemployed, for example. This can lead to perceptions of injustice and disgruntlement, which has been tied to the formation of intention to retaliate against the unjust system, including acts of cyber-attack [4]. By evaluating this social process, one can see that official efforts to reach disaffected communities, to engage young individuals vulnerable to radicalization, and to build trust between these communities and law enforcement can offer the opportunity to ameliorate and avert the process that can lead to radically-inspired acts of homegrown terrorism, including cyberterrorism [23]. Counter-radicalization efforts should be built upon restoring community trust, building bridges, and developing educational programs aimed at gaining the cooperation of both the immigrant communities and a given population as a whole [24]. Building resilience against violent extremism begins at the local level. Research has established the efficacy of counter-radicalization strategies which are best achieved through the engagement and empowerment of individuals and groups [23].

VI. THE ROLE OF PRE-KINETIC EVENTS

Drawing on the military strategy context, Willison & Warkentin [3] introduced the concept of “left of bang” to the security literature. The idea on moving “to the left” is inspired in part by fourth-generation military strategy, which has ascertained the need to consider “left of bang” in order to stop the emergence of malicious intention long before an act of aggression happens. In this regard, security is mainly a thinking game where intellect borne of training and experience should be to be the most prized commodity”. The trained individuals can design security systems to minimize risk in the initial stages and thus avoid violations. In military terminology, this is often called pre-kinetic or left-of-bang, meaning that certain actions should be taken long before an incident happens. This could be the most crucial, yet the most underrated dimension of information security management [25] – it is incumbent that we act proactively before kinetic events ensue. Comparably, military strategists advocate winning the hearts and minds of the population, analogous to the way that organizations or governments may try to psychologically affect employees long before any cybercrime occurs. In the context of counter-cyberterrorism, efforts to influence potential terrorists long before they form their dangerous intentions can yield significant payoff in the long run.

VII. CONCLUSION

The challenges of addressing the increasing threats from cyberterrorism are great. Governments are expending increasing resources to counter such threats, focusing the greatest efforts on technical measures to prevent or detect cyberterrorism events, though the seeds are often sowed far in advance of the perpetration of such acts. We propose greater attention to the processes “left of bang” to understand root causes of the actions of cyberterrorists so that we can establish efficacious methods to break the chain early in the sequence and avert the formation of cyberterrorist intentions.

REFERENCES


Ranking Terrorist Nodes of 26/11 Mumbai Attack using Analytical Hierarchy Process with Social Network Analysis

Pankaj Choudhary  
Department of Computer Engineering  
Defence Institute of Advanced Technology  
Pune, India  
pankaj_pcse13@diat.ac.in

Upasna Singh  
Department of Computer Engineering  
Defence Institute of Advanced Technology  
Pune, India  
upasnasingh@diat.ac.in

Abstract—26/11 Mumbai attack is one of the major terrorist attack in India, executed on November 26, 2008. For analyzing such terrorist attacks, Social Network Analysis (SNA) is the most effective technique, recognized worldwide. Various centrality measures of SNA have capability to find the key players/leaders and obtain a rank ordering of nodes in such networks. But, most of the times, each of these individual measures result in identifying different key players and rank ordering. Analytical Hierarchy Process (AHP) can be combined with existing SNA measures in order to find the overall ranking of nodes based upon decision maker’s subjective or objective judgements. In this paper, 26/11 Mumbai terrorist network is analyzed using AHP with existing SNA centrality measures as decision criteria, to discover the overall ranking of all 13 terrorists involved in the attack and to find the key players/leaders among them. Further, sensitivity analysis is discussed to deal with changes in subjective judgements. The experimental results show that when AHP is combined with SNA centrality measures, results in more promising ranking of nodes in terrorist networks than only centrality is considered.

Keywords—Analytical Hierarchy Process (AHP); Social Network Analysis (SNA); centrality measures; 26/11 terrorist attack; finding key players; ranking terrorist nodes

I. INTRODUCTION

In recent years, India has been witnessed frequent terrorist attacks. On November 26, 2008, one of the major terrorist attack, known as 26/11 terror attack, was executed in Mumbai (India) by members of Lashkar-e-Taiba (LeT), a Pakistan based terrorist group. Ten terrorists equipped with advanced weaponry and three handlers in Pakistan, were involved in the attack and five major locations in Mumbai were targeted, causing nearly 260 casualties from ten different countries [2], [3]. The attack was well planned and handlers from Pakistan were in communication with these terrorists, during the attack.

Various high-ended methodologies have been evolved over time for analyzing terrorist groups and networks. Social Network Analysis (SNA) is one of the most effective and predictive analytical tool for studying various terrorist activities. Just after tragic 9/11 terrorist attack, research based on SNA in counter-terrorism became very popular. In counter-terrorism domain, particularly, centrality measures of SNA [5], [6], [7], have been frequently used for identifying the key players/leaders and ranking of terrorists based on various terrorist networks/activities.

The Analytical Hierarchy Process (AHP), first proposed by Satty [15], [16], is a multiple attribute decision making technique for investigating and organizing complex decisions. Based on some criteria (attributes), alternatives and subjective or objective measures, AHP helps in finding relative importance or ranking of alternatives. In many cases, AHP is used as an effective tool for finding key players/leaders and ranking of nodes in various social network studies with respect to several criteria (attributes).

In this paper, 26/11 terrorist network is analyzed using AHP in addition with existing SNA Centrality measures, in order to discover the ranking of all 13 terrorists involved in the attack and identifying the key players/leaders among them. Data used for 26/11 terrorist network is based on the work done by [1] and Government of India’s report on Mumbai attack [2].

The rest of the paper is structured as follows: Section 2 of the paper briefly about the SNA and various centrality measures for identifying key players/leaders and related research done in the Counter-Terrorism domain. Section 3 discusses about AHP process, steps for finding ranking of alternatives and related studies using AHP with SNA in various decision problems, including its application in Counter-Terrorism. Section 4 presents research work done by us of analyzing 26/11 attack network and ranking the terrorist nodes using AHP as a tool, in addition with SNA Centrality measures. Finally, work is concluded and some suggestions for future research is provided in Section 5.

II. SNA AND AHP: RANKING NODES IN SOCIAL NETWORK

A. Social Network Analysis (SNA)

Social Network Analysis (SNA) is a graph and network theory based approach for studying social relationships in terms of nodes (individuals, organizations, events etc.) and ties between them (relationship i.e. friendship, kinship, conversation, money transaction, co-workers etc.). Various SNA measures has been evolved for representing relations among individuals, identifying key players/leaders and subgroups and communities, finding topology, examining strong or weak ties and strength of network.
Many studies are the evidence of effective use of SNA in Counter-Terrorism applications. For example, Krebs’s work on 9/11 Al-Qaeda terrorist network and using SNA Centrality measures for finding key players [8], study of SNA and multi-agent models for destabilizing covert terrorist networks by Carley [9], [10], study of Noordin Top’s terrorist network and disruption of terrorist networks using SNA centrality measures by Everton [11], analysis of 26/11 terror network using SNA by Azad [1], SNA based studies on major terrorist groups operating in India by [12], [13], our previous survey article on Social Network Analysis for Counter-Terrorism [14] etc.

Terrorist organizations/groups can be represented using SNA, as individual terrorist members as nodes and relationship among them as edges. This graph/network is often represented as an adjacency matrix. SNA offers several measures for finding key players or central node within the network and ranking of nodes in the network, often grouped as Centrality measures. Numerous measures of Centrality like Degree, Betweenness, Closeness and Eigenvector has been proposed over time for the same [5], [6].

Degree centrality [5] of particular node is the number of direct links to other nodes. Higher value of Degree Centrality often considers the node as an active individual in the network. In terrorist network study, this helps in identifying the leader or the hub of the network.

Betweenness centrality [5] infers the nodes, which act as a broker between groups or communities in the network. In any network, it is calculated as the number of times the node lie between the shortest paths between any pair. In terrorist networks, nodes with higher betweenness values help in identifying potential brokers between two groups or communities, which reflects powerful and influencing nodes, containing maximum information.

Closeness centrality [5] is a mean length of all shortest paths from an individual node to all other nodes in the network. In terrorist networks, nodes with high closeness values are much closer to other nodes and can help in accessing the information in quicker manner.

Eigenvector centrality [6] a variation of Degree Centrality, is a measure of how much an individual is connected to other highly connected members of the network. It infers the relative importance of a node, in terms of influence, to other connected nodes in the network. In terrorist networks, nodes with high eigenvector centrality help in identifying most central nodes in the network, globally and tracking individuals, well connected to other well connected actors.

B. Analytical Hierarchy Process (AHP)

Analytical Hierarchy Process (AHP), proposed by Saaty [15], [16], [17], in 1977, is a multiple attribute decision making technique to organize and analyze complex decision problems, based on various criteria (attributes), alternatives and subjective judgements on them. AHP also facilitates the objective values of criteria and alternatives for decision making. AHP is applied in many decision problems, for example, choosing the best alternative from a given set, ranking a set of alternatives, prioritizing alternatives, conflict resolution, quality management, corporate planning, government policy development etc.

In many recent studies, AHP is found out to be an effective technique in addition with SNA, especially for identifying key players/leaders and ranking of nodes in various social networks. Fox and Everton used AHP as a tool to find the influence and ranking of nodes in the social network [18] and to identify key nodes in Noordin terrorist network in this study [19]. Fox [20] in another study, used AHP combined with TOPSIS for ranking terrorist nodes in network. Liebowitz [21] studied the integration of AHP with SNA for knowledge mapping in organization.

The steps of AHP for finding the key players and ranking of nodes in particular social network can be outlined as follows,

Step 1: Build the decision hierarchy for the decision problem by decomposing the problem into a hierarchy comprising the decision goal, alternatives for achieving it, and the criteria and sub-criteria for assessing these alternatives.

Step 2: Establish priorities among the criteria, sub-criteria and alternatives, either by using,

- Saaty’s nine-point scale (in Table I) to make series of judgments based on pairwise comparisons of the elements, or
- Objective measures and values available for criteria, sub-criteria and alternatives.

**TABLE I. Saaty’s Nine Point Scale for Pairwise Comparisons**

<table>
<thead>
<tr>
<th>Intensity of Importance</th>
<th>Importance</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Equal</td>
</tr>
<tr>
<td>3</td>
<td>Moderate</td>
</tr>
<tr>
<td>5</td>
<td>Strong</td>
</tr>
<tr>
<td>7</td>
<td>Very Strong</td>
</tr>
<tr>
<td>9</td>
<td>Extreme</td>
</tr>
<tr>
<td>2,4,6,8</td>
<td>Intermediate Values</td>
</tr>
</tbody>
</table>

It result in pairwise comparison matrix for criteria, sub-criteria and alternatives. For ensuring the consistency of judgements, this matrix must be stable according to consistency ratio (CR) defined by Saaty [15], [17]. The value of CR should be less than or equal to 0.1 for pairwise comparison matrix.

Step 3: Using eigenvector method to estimate the weights of each criteria and sub-criteria, which result in comparative ranking of criteria and sub-criteria.

Step 4: For m criteria and n alternatives, aggregate criteria weights (the m × 1 matrix) with values of alternatives with respect to each criteria (the n × m matrix), using matrix multiplication, in order to determine the final ranking of each alternatives (the n × 1 matrix).
III. ANALYZING 26/11 NETWORK USING SNA AND AHP

26/11 Mumbai attack is one of the major terrorist attack in India. As confirmed by various sources [2], [3], [4], ten terrorists were involved in attacking five prime locations of the city, while three handlers from Pakistan were controlling these attackers. The details of these attackers and their handlers are mentioned in Table II and Table III.

<table>
<thead>
<tr>
<th>TABLE II. LeT Handlers who Operated from Pakistan (P)</th>
</tr>
</thead>
<tbody>
<tr>
<td>S. No.</td>
</tr>
<tr>
<td>1.</td>
</tr>
<tr>
<td>2.</td>
</tr>
<tr>
<td>3.</td>
</tr>
</tbody>
</table>

Based on the work done by Azad [1] and government of India’s report, revealing intercepted phone calls [2], data for 26/11 network is gathered in the form of adjacency matrix and social network graph.

<table>
<thead>
<tr>
<th>TABLE III. Attackers who Operated in Mumbai</th>
</tr>
</thead>
<tbody>
<tr>
<td>S. No.</td>
</tr>
<tr>
<td>1.</td>
</tr>
<tr>
<td>2.</td>
</tr>
<tr>
<td>3.</td>
</tr>
<tr>
<td>4.</td>
</tr>
<tr>
<td>5.</td>
</tr>
<tr>
<td>6.</td>
</tr>
<tr>
<td>7.</td>
</tr>
<tr>
<td>8.</td>
</tr>
<tr>
<td>9.</td>
</tr>
<tr>
<td>10.</td>
</tr>
</tbody>
</table>

Fig. 1 shows the adjacency matrix and Fig. 2 shows the social network graph for 26/11 network, produced using UCINET [22] and NodeXL [23].

A. Applying SNA on 26/11 network

Now based on various SNA Centrality measures, for 26/11 network, value of six important centrality measures (Degree, Eigenvector, In-Degree, Out-Degree, Closeness and Betweenness) are calculated and verified with the help of UCINET [22], ORA [24] and Gephi [25]. All these centrality values are normalized between 0 and 1 for further use in AHP.

These normalized values and their distributions are shown in Table IV and Fig. 3 respectively.

<p>| TABLE IV. SNA Centrality Measure Values for 26/11 Network (Normalized) |
|-----------------|-----------------|-----------------|-----------------|-----------------|-----------------|</p>
<table>
<thead>
<tr>
<th>Terrorist Name</th>
<th>Degree</th>
<th>Eigenvector</th>
<th>In-Degree</th>
<th>Out-Degree</th>
<th>Closeness</th>
<th>Betweenness</th>
</tr>
</thead>
<tbody>
<tr>
<td>Abu Kaahfa</td>
<td>0.080645</td>
<td>0.081041</td>
<td>0.096774</td>
<td>0.064516</td>
<td>0.083074</td>
<td>0.078947</td>
</tr>
<tr>
<td>Wassi</td>
<td>0.193548</td>
<td>0.178492</td>
<td>0.193548</td>
<td>0.193548</td>
<td>0.051921</td>
<td>0.568421</td>
</tr>
<tr>
<td>Zarar</td>
<td>0.080645</td>
<td>0.081041</td>
<td>0.096774</td>
<td>0.064516</td>
<td>0.083074</td>
<td>0.078947</td>
</tr>
<tr>
<td>Hafiz Arshad</td>
<td>0.129032</td>
<td>0.163631</td>
<td>0.129032</td>
<td>0.129032</td>
<td>0.062305</td>
<td>0.131579</td>
</tr>
<tr>
<td>Javed</td>
<td>0.080645</td>
<td>0.122399</td>
<td>0.096774</td>
<td>0.064516</td>
<td>0.093458</td>
<td>0.005263</td>
</tr>
<tr>
<td>Abu Shoaiab</td>
<td>0.080645</td>
<td>0.09498</td>
<td>0.064516</td>
<td>0.096774</td>
<td>0.088266</td>
<td>0.005263</td>
</tr>
<tr>
<td>Abu Umer</td>
<td>0.129032</td>
<td>0.163631</td>
<td>0.129032</td>
<td>0.129032</td>
<td>0.082605</td>
<td>0.131579</td>
</tr>
<tr>
<td>Abdul Rehman</td>
<td>0.048387</td>
<td>0.00198</td>
<td>0.032258</td>
<td>0.064516</td>
<td>0.107996</td>
<td>0.005263</td>
</tr>
<tr>
<td>Fahadullah</td>
<td>0.048387</td>
<td>0.00198</td>
<td>0.032258</td>
<td>0.064516</td>
<td>0.107996</td>
<td>0.005263</td>
</tr>
<tr>
<td>Baba Imran</td>
<td>0.032258</td>
<td>0.053433</td>
<td>0.032258</td>
<td>0.032258</td>
<td>0.088266</td>
<td>0.005263</td>
</tr>
<tr>
<td>Nasir</td>
<td>0.032258</td>
<td>0.053433</td>
<td>0.032258</td>
<td>0.032258</td>
<td>0.088266</td>
<td>0.005263</td>
</tr>
<tr>
<td>Ismail Khan</td>
<td>0.032258</td>
<td>0.00198</td>
<td>0.032258</td>
<td>0.032258</td>
<td>0.041537</td>
<td>0.041537</td>
</tr>
<tr>
<td>Ajmal Amir Kasab</td>
<td>0.032258</td>
<td>0.00198</td>
<td>0.032258</td>
<td>0.032258</td>
<td>0.041537</td>
<td>0.041537</td>
</tr>
</tbody>
</table>

Fig. 1. Adjacency Matrix of 26/11 Network [1]
Individual values of each of these centrality measures, can help in identifying the relative importance of each node compared to others. But all the measures are unable to identify the same node as the key player/leader, also results in different rankings of nodes. For example, in 26/11 network, Wassi is identified as a key player, as its higher values of Betweenness, Degree, Eigenvector, In-degree and Out-degree centrality, but on the other hand, Abdul Rehman and Fahadullah are the key players according to the higher value of Closeness centrality.

For obtaining the overall/final ranking of nodes, priorities can be assigned to these centrality measures using the decision maker’s subjective judgements as an input. For this purpose, AHP is used as a multiple attribute decision making technique.

B. Applying AHP on 26/11 network

Now, for finding the overall ranking of terrorist nodes for 26/11 network, decision hierarchy is framed as shown in Fig. 4.

Above mentioned six centrality measures are considered as our decision criteria and all thirteen terrorist nodes are considered as alternatives for achieving our decision goal. Based on the work done by Fox and Everton [18] and using Saaty’s nine point scale [15] pairwise comparison matrix for our six decision criteria is generated, as shown in Table V. These pairwise comparisons are based on the preferences of each centrality measure over other, usually decision makers consider for ranking nodes in various social networks.

Our pairwise comparison matrix is consistent, as the value of consistency ratio, CR is 0.021 (i.e. less than 0.1)

For 6 criteria and 13 alternatives, using eigenvector method criteria weights (in form of 6 X 1 matrix) are calculated as shown in Table VI.

In order to obtain the final ranking of 26/11 terrorist nodes, criteria weights (6 X 1 matrix, in Table VI) is aggregated with calculated centrality values of each terrorist nodes (13 X 6 matrix, in Table IV), using matrix multiplication.

The final AHP score values (13 X 1 matrix) and final ranking of all terrorist nodes of 26/11 network is shown in Table VII.
The final ranking of terrorist nodes based on calculated AHP score is visualized in social network graph (node size ordered by AHP score) as shown in the Fig. 5, using NodeXL [23] and distribution of final ranking are shown in Fig. 6.

![Social Network Visualization of Final Rankings using AHP for 26/11 Network](image)

**Fig. 5.** Social Network Visualization of Final Rankings using AHP for 26/11 Network

![Distribution of Final Rankings for 26/11 Network](image)

**Fig. 6.** Distribution of Final Rankings for 26/11 Network

Based on this study, it is found that Wassi, with highest AHP score is the most important individual, as he was handling the entire operation. Other key players in top 5 ranking, are Hafiz Arshad, Abu Umer, Abu Kaahfa and Zarar. The result represents the overall ranking of terrorist nodes based on the subjective judgments over considered six centrality measures as our decision criteria.

These rankings are subjected to change with the deviation in pairwise comparison of decision criteria. Sensitivity analysis using controlled trial and error method can be used to assess the change in rankings with variation in subjective judgments of the criteria and finding the cut-off point, affecting the rankings.

**IV. CONCLUSION**

SNA is one of the most powerful and effective analytical tool for studying various complex terrorist networks and organization. SNA Centrality Measures have been frequently used for identifying the key players/leaders and ranking of terrorists based on various terrorist networks/activities. Individual values of each of these centrality measures leads in identifying different nodes as the key players/leaders, which results in different rankings of nodes. So, for obtaining the overall cumulative ranking, based on the combined effect of each of these centrality measures, AHP can be used in addition to SNA. AHP is found out to be an effective technique, for identifying key players/leaders and ranking of nodes in various social networks, based on several criteria/attributes and subjective or objective comparisons over them.

In our work, we analyzed 26/11 Mumbai terrorist network using Analytical Hierarchy Process in addition to existing SNA Centrality measures, for finding the overall ranking of all 13 terrorists involved in the attack and identifying the key players/leaders among them. Ranking we got is based on the subjective judgments over considered six centrality measures as our decision criteria. Sensitivity analysis with trial and error method can be used to deal with changes in ranking with changes in subjective judgements for our decision criteria.

Though the data used in this study is small (13 nodes, 17 edges), further, the approach can be applied for large datasets. In future, some other multiple attribute decision making techniques (other than AHP) can be considered with SNA for finding key players and ranking nodes. We believe that, AHP when applied with SNA can be very effective in counter-terrorism applications, especially in, finding key players/leaders and ranking of terrorists in terrorist networks.

**ACKNOWLEDGMENT**

We would like to thank to all the members of Digital Forensics Lab, Dept. of Computer Engineering, DIAT, Pune for their comments and suggestions and successful completion of this work.

**REFERENCES**


Linguistic Features of Phone Scams: 
A Qualitative Survey

Judith L. Tabron
Information Technology / The Institute for Forensic Linguistics, Threat Assessment, and Strategic Analysis at Hofstra University
Hempstead, NY 11549 USA
judith.tabron@hofstra.edu

Abstract—The interactive human components of serious cybersecurity breaches continue to be a problem for every organization. Helping people identify social engineering attempts over the phone will be cheaper and more effective than yet another technological implementation. At minimum it will add an important and necessary layer to defense in depth.

Forensic linguistics is the study of language as evidence for the law. It is a relatively new field and has not previously been applied to cybersecurity. Linguistic analysis uncovers several features of language interaction in a limited data set (recorded IRS phone calls) that begin to answer how forensic linguistics could assist in cybersecurity defense.

This paper will briefly introduce and explain polar tag questions, topic control, question deferral, and irregular narrative constructions in IRS scam phone calls, and offer some starting points for identifying such linguistic properties during the course of a phone call to help improve defense at the human level.

Keywords—social engineering; forensic linguistics; cybersecurity; securing the human; phone scam

I. FORENSIC LINGUISTICS: LANGUAGE AS EVIDENCE

To secure the human, perhaps our most vulnerable link in the security chain, I suggest that we move the goalposts. Trying to train people not to be fooled by social engineering attacks that are designed to fool them may have a limited effect. Forensic linguistics is the study of evidence provided by language use we cannot disguise. This is a preliminary investigation for a larger study we hope to carry out at Hofstra University applying forensic linguistics methodology to social engineering to identify features of the interaction that the target can, if trained, identify and stop.

Forensic linguistics uses the science of linguistics to examine language as evidence in civil or criminal cases. It is perhaps most frequently associated with author identification, such as was employed in the famous Unabomer case by Jim Fitzgerald, then with the FBI. Author identification has been shown to be most reliable when determining between a small universe of given possible authors, to answer whether a suicide note was written by a daughter or her mother, or whether text messages from a particular phone were written by a murder victim or her alleged killer, as described in Tim Grant’s “Txt 4n6” [1]. It has also been used to evaluate the severity of threat letters, or to build a profile of the writer. It is fundamental to forensic linguistics that our built-in language “programming” is not something we can supersede. Evidence of our own language tendencies arise even when we are trying to hide them, such as in a kidnap case worked on by the founder of the field, Roger Shuy, where a writer attempting to disguise himself misspelled several common words but correctly spelled “daughter”, a word many native speakers of English still have trouble spelling, and used a regionalism only found in the Akron, Ohio area.

All the sub-fields of linguistics can be brought into play for forensic work: phonology, morphology, semantics and pragmatics describe how language works from the smallest unit of sound up through words and phrases to sentences. How a speaker forms a “speech act”, an act committed solely through language, has been analyzed for decades by linguists, and is particularly important in the areas where a speech act commits a crime, such as a bribe or a threat [2]. And sociolinguistics help us understand how speech works in interaction across social units such as divisions of gender, ethnicity, communities of practice (like all you security professionals here today), and other groups. Robert Leonard, head of the Institute, who has worked with law enforcement and linguists all over the world, has written an article summarizing “Forensic Linguistics” much better than I could [3].

Basic research in the area of forensic linguistics can improve outcomes of justice, such as my own assistance with a project by Prof. Eric Freedman of the Law School of Hofstra University to make clearer instructions to juries who must consider awarding the death penalty; the work of the Innocence Project in identifying confessions that were not written by the suspect who allegedly wrote them; or to help us understand the nature of language evidence in civil or criminal cases, such as Tammy Gales’ work analyzing a corpus of threat letters in cooperation with the FBI to better understand what types of language actually appear in those threat letters rather than analysts going by their personal experience of threat letters [4,5,6]. Research like this strongly helps inform law enforcement.

II. APPLICABILITY TO SOCIAL ENGINEERING

To date, forensic linguistics has not been brought to bear on cybercrime. Like all of you I worry about identifying anomalous network activity, making sure not to pass credentials in the clear, and encrypting data at rest and in
transit. But the more I learned about cybercrime the more I realized that it often included a language element between humans that was not being analyzed. The human is still the most vulnerable point of any system, as recent events in the news have shown. Social engineering has long been studied by psychologists, with a great deal of attention paid to the act of deception, but not to the linguistic features of phone hacking or even phishing emails themselves, which are language acts.

We all instinctively know and recognize genres of conversation that we engage in all the time: phone calls home to Mom, making a date with a friend, taking a help desk call from a customer. Linguists have studied such interactions for decades, especially those researchers who have followed in the footsteps of J.R. Firth and Michael Halliday who developed systemic functional linguistics [7]. In a very small way this paper is a first gesture toward applying that methodology and background to social engineering in order to better understand how it works, perhaps ideally to help train people to defend against it.

This qualitative study begins to gather information on the features of phone scams that could be pursued in a quantitative study with a larger data set and control set.

III. DATA: RECORDINGS OF IRS SCAM CALLS

I begin with an analysis of publicly available IRS phone scam recordings. While not ideal data, as such recordings are made by people who know that they are scams and such people may not participate in the conversations just as an unsuspecting target would, even a small set of such recordings allow us to begin to develop a theory of their operation that could inform a larger study with better data. This study is a qualitative study, following the grounded theory approach commonly used in social sciences since the 1970s [8,9,10] in which one searches for what is atypical as well as what is typical in order to fill out a theoretical understanding of the data and to develop areas of inquiry I hope to pursue with a larger data set.

The IRS phone scam posted on YouTube at https://www.youtube.com/watch?v=Jb9iilimPWyw, “IRS PHONE SCAM CAUGHT LIVE”, is used throughout this paper to provide examples. Additional samples were gathered for the genre analysis as well as to expand upon the other analyses for this paper. The samples were all collected as videos from YouTube.com, the audio extracted and transcribed.

These IRS scams have risen dramatically in the recent year or two in the United States; the Treasury Inspector General for Tax Administration has reported receiving at least 650,000 reports of them this year, between 9,000 and 12,000 per week [11,12], and that incurs tremendous overhead costs for investigation, even before one considers the thousands of victims who have succumbed to the scam and who have lost millions of dollars. As the calls tend to take similar forms [13], they may form a recognizable genre of their own and as such be subject to the type of analysis linguists are familiar with from Swales’ Genre Analysis [14] and subsequent works building upon it.

In “IRS PHONE SCAM CAUGHT LIVE”, the recording is of a phone call in which there are three participants. The person recording the call is the target of the scam; would-be scammers have robo-called her phone and left an alarming message and instructions to call them back. Knowing that this is a scam, the would-be target records the entire call from the beginning. That the target is not fooled is a feature of the samples I was able to collect, because people do not record scam phone calls to share with others unless they know before the call occurs or close to its beginning that it is a scam. This dual-deception nature of the phone call samples on YouTube - that the scammers are trying to fool the call recipient into thinking that they are in trouble, and that the call recipient is trying to fool the scammers into thinking that they can be duped - is a peculiarity of this genre that complicates understanding the operation of the calls as actual attempts at crime.

The scam is simulating a business phone call, a genre that has been studied by systemic functional linguists such as Fairclough [15] and Ventola [16]. All language interactions are social in nature, including service interactions such as phone calls from bureaucratic offices such as the IRS. Though in these calls, both sides (or the various callers, as the illustrative sample included 3 conversants) are aware that the service interaction here is being simulated, the sides still participate in it as if it were the interaction it purports to be.

Data from the sample have been put in table form, turns are numbered and the time of the turn onset is noted. Latching - starting one turn directly at the end of another with no pause - is indicated with //, and single / indicates a rough phonetic inscription such as in Wha/du/ I do. :: indicates an extended sound, () indicates a pause briefer than one second, and (#) indicates roughly how many seconds a pause extends.
Later on speaker C, the purported supervisor of speaker A, comes on the line; speaker A never appears again in the conversation and the rest of the call is between speaker B, the target, and speaker C, looking something like this:
### IV. FINDINGS

#### A. Polar tag questions

The use of a polar tag question is when someone ends a conversational turn (not a sentence, but a turn) with a yes/no question. As it turns out, research has been done on polar tag questions: answers are preferred over no answer, agreement is preferred over disagreement, and answers that accept the terms of the question are preferred over answers that resist.\\(^1\)

“Preferred” means we feel a natural pressure through the normal process of language to respond that way. Phone scammers use noticeably more polar tag questions than most conversation. This may also be a feature of any coercive conversation (the TSA doing a patdown, your mother’s instructions on how to do something, etc.) We’ll look at how this occurs in the sample.

When the target, speaker B, asks questions, she is asking primarily informational questions.

\(^1\) From Stivers [17], “Prior work has documented that certain responses are preferred over others in terms of their structural “fit.” For instance, answers are preferred over non-answer responses (Clayman, 2002; Stivers and Robinson, 2006); confirmations are preferred over disconfirmations (Heritage, 1984); answers which accept the terms of the question, and accept reduced agency over them, are preferred over those that resist either of these dimensions (Raymond, 2003; Heritage, forthcoming; Stivers, forthcoming; Stivers and Hayashi, 2010). All of these more fitted responses are typically delivered as “preferred” responses in that they are typically delivered more quickly than the alternative, without mitigation and without accounts (Heritage, 1984; Pomerantz, 1984; Stivers et al., 2009)."
As pointed out in Athanasiadou [18], “the use of the different modes of questioning indicate the relationship that exists between the questioner and the respondent (intimacy, social distance, authority).” (107) Speaker B is clearly the supplicant, begging for information.

Speaker C, on the other hand, primarily seeks agreement through his questions:

TABLE III. TYPES OF QUESTIONS OVERALL FROM SPEAKER B

<table>
<thead>
<tr>
<th>Type of Question</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tag questions</td>
<td>2</td>
</tr>
<tr>
<td>Informational questions</td>
<td>21</td>
</tr>
<tr>
<td>Falling rather than rising intonation,</td>
<td>1</td>
</tr>
<tr>
<td>yet phrased as a question</td>
<td></td>
</tr>
<tr>
<td>Rhetorical questions</td>
<td>0</td>
</tr>
</tbody>
</table>

As stated in Athanasiadou [18], “Any type of question may be rhetorical, but rhetorical questions are not questions at all but assertions, and are usually phrased as an open-ended question. They minimize the emphasis on the problem.” Rhetorical questions provide information, they don’t ask for it. Rhetorical questions “minimize the emphasis on the information channel and stress the social relationships involved”, and as such rhetorical questions are actually the opposite of informational questions. I would argue that speaker C’s tag questions are also intended primarily as rhetorical, creating an obligation in speaker B to respond positively if at all, but, as we saw in turn 61, speaker C’s tag questions are also used as much to keep the floor as to elicit response. Athanasiadou adds that in the other two types of questions, interrogation and exam questions, the stress again is on the “dominance of the speaker” in requiring an answer. Speaker C’s use of questions puts Speaker B in a subordinate position while constantly requiring response and agreement.

B. Topic control

Controlling the topic and the floor - what is said and who gets to talk - is a classic power play identified by conversational discourse analysis.

The next topic, question deferral, is related to topic control and you saw it in the sample above between Speaker B and Speaker C. Speaker B wishes to move ahead just one step in the narrative - to know how to address the payment she must send - but Speaker C refuses to give that information and defers to let her move ahead in the series of future steps that she must follow. Critical discourse analysis highlights Speaker C’s power relation to Speaker B: speaker B’s questions will not be answered - all answers are deferred to some future moment which has not happened yet, and the deferment is controlled by Speaker C. In this way Speaker C will not permit Speaker B to take control of the topic or the floor. One can see through this series of turns that Speaker B attempts not only to ask questions, and thereby control the topic, but even to speak. She and Speaker C are wrestling for control of the floor throughout this section, interrupting each other throughout.

For a closer analysis look at turn 60. Speaker C issues instructions. Speaker B interrupts with positive feedback (“Oh thank you”). Speaker C still does not yield the floor, overlapping her “Oh thank you” with a question of his own, “All right?” Rather than directly respond to the question that interrupted her affirmative response looking for an affirmative response, speaker B attempts a topic redirect, starting with a second affirmative response “Thank you so much” but then effectively latching on to her own turn so as to retain the floor when she adds “And now, if, once I go to the Winn Dixie why don’t you just give me, um, your address so I can send the tax voucher because”. Speaker C interrupts her in turn 63 to take control of the floor and the topic again: “I’ll let you know that ma’am, I’ll, I’ll give you that information step by step.” Her attempt to interject “um” is overridden by his speech in turn 63.

Speaker A, the first speaker, had a topic that was, in a sense, built in. She answered the phone when the target called her and informed the target that she had a legal problem because of her taxes. If you look at the first sample above, you notice that Speaker A actually leaves rather long silences, up to two seconds or even more. She is leaving topic gaps waiting for Speaker B, the target, to introduce a new topic, like perhaps paying the bill.

Speaker C never leaves a gap. His part of the conversation is filled with much more latching, and absolute topic control.

C. Question deferral

Perhaps a more unique signature of phone hacking/scamming attempts is the deferral of answering questions. In this sample we can see one of the speakers (speaker C) refuses to answer any questions and repeatedly defers answering them until some future time. Or he may just ignore the question. Deferring answering questions is not a feature of everyday conversation and may be one of the key ways one can tell that one is being manipulated. As discussed
above, a question creates requirements in a conversation, at the very least some sort of expectation of response. Speaker C does not answer questions. Research indicates that it is difficult to quantitatively differentiate between different types of institutional conversations, but one can do so based on qualitative assessment such as noting distinctive turn-taking systems - even institutional language rarely has distinctive turn-taking [19]. It is rare that questions are deferred and it occurs only in situations like a formal meeting, where there are rules regarding when questions may be submitted or ignored.

<table>
<thead>
<tr>
<th>Who do I send them to?</th>
<th>deflected to future</th>
</tr>
</thead>
<tbody>
<tr>
<td>Where do I mail this?</td>
<td>deflected to future</td>
</tr>
<tr>
<td>Are you ready?</td>
<td>interrupted</td>
</tr>
<tr>
<td>Who do I mail this to?</td>
<td>deflected with “anger”</td>
</tr>
<tr>
<td>Which Winn Dixie do you want</td>
<td>deflected to future</td>
</tr>
<tr>
<td>What’s your name?</td>
<td>ignored</td>
</tr>
<tr>
<td>Are you gonna hold?</td>
<td>Answered (“Yes”)</td>
</tr>
</tbody>
</table>

At this moment it’s worth returning to Fairclough’s formative text on social discourse where he points out (following the work of Elliot Mishler on medical interviewing) that in fact questions have THREE parts: the question, the answer, and the (often implied) acceptance of the answer. Not only does Speaker C deny Speaker B answers, by simply deflecting her questions to some future moment or ignoring them, he does not allow her the opportunity to reject or accept his answers. In this way he disallows the implied third part of a question and cuts it short.

Does this happen to you in everyday conversation? Would you notice if it did? Mishler’s research pointed out the lack of social connection between doctors who did not appropriately provide acceptance of an answer from a patient, as well as, of course, the alienating effect of a doctor not answering a question from a patient. It is illustrative of the coercive nature of the phone call that one can easily see here that it is no longer even attempting to simulate a service call such as one might find in any bureaucratic situation, but instead is simply a conversational trap with no way out for the target.

D. Violations of narrative structure

I will end by quickly mentioning that narrative analysis identifies in any story multiple phases that follow in a predictable order. William Labov [20] in 1967² defined a well-formed narrative sequence as follows:

1. Abstract: What is the story about?
2. Orientation: Who, when, where, how?
3. Complicating action: Delays resolution, adds suspense...
4. Evaluation: Who's the hero? Who's the bad guy?
5. Result / Resolution: What happened in the end?
6. Coda "And that's why I can never go back there again."

Witness statements and confessions can be analyzed as following or breaking this sequence, which often reveals interesting places to focus investigation.

In the samples discussed here, the flow switches from an orientation section set in the past (“this is what has happened”) straight into an emergency that is happening right now (“do this now to solve it”). This atypical shift of tense creates a sense of urgency in the target.

The shift can occur together with a shift in speaker, and several of the calls include at least two scamming speakers. The first speaker sets up an odd narrative, if you think about it: “You are in trouble with the IRS. Goodbye.” These speakers leave topic space open, though, for the target to introduce a new topic such as paying the bill.

Once the topic of paying is introduced the target is usually shifted to another scammer who has an entirely different narrative. The focus is entirely on “These are the things you can do to get out of the situation.” If the situation were in the past, as the first caller indicated, it would be over. The scammers’ language even reflects that this is a present-tense situation: “This is a very serious situation that is going on in your life right now” says one. The second caller uses present tense to create a situation that isn’t over, it’s going on right now, and can be resolved right now but only if the target follows all the instructions.

There are no instructions in a narrative sequence (except perhaps to intensify focus: “listen to this part”). A narrative is a demand for an extremely long conversational turn where the speaker narrates and the listener listens. These calls are all set up as beginning with a narrative. “These are the things that have occurred in the past. You did not pay your taxes correctly. A warrant has been issued for your arrest. You did not receive notification.” And then... nothing. The narrative structure simply stops. Only a statement about what will happen in the near future: “Police are coming to arrest you.” Very short timeframes are given, like within the next ten to thirty minutes. Loved ones may be involved: no, you can’t call your wife, she is going to be arrested at the same time as you. It’s no longer the past, it’s not even the future, it’s the now. That is a malformed narrative.

Bureaucratic interactions need not involve narrative, of course: but when they do they are complete narratives. Imagine a conversation where you are trying to return a purchase but

² Many scholars since have followed and refined his work, perhaps most notably C. Kohler Riessman [21].
don't have a receipt. You may employ even a very abbreviated narrative: (with an abstract) “I’d like to return (and an orientation) this shirt (complicating action) but I don’t have the receipt. (Further complicating action) I know I had it in my wallet yesterday but today I can’t find it. (Evaluation) I can’t keep track of anything, I know it’s my fault. (Resolution) But I thought I’d bring it in anyway. (Coda) I hope you can help me.” You might try the same interaction without the narrative: just thrusting the shirt forward and saying “No receipt”. But you’re not likely to just offer the orientation, are you? “I’d like to return this.” Or maybe you are, and you’re waiting for her next line: “Certainly, do you have the receipt?” because you expect to carry out the interaction as a dialogue. You may offer the narrative as part of the conversation. You will not offer just the first part of the narrative and then say “Well, nothing more you can do” and hand over the conversation to someone else.

If our staff were oriented to this type of question, they might realize that it’s reasonable to respond “Wait, what?” to a narrative that is malformed. They could also be trained, as police have been trained, to ask questions that fill in the missing information. “Okay, tell me again how it goes from notifying me of a tax mistake to issuing a warrant for my arrest. How do the IRS and the police work together on this?” Or they could simply realize that the parts don’t add up to a whole, and ask the person to tell the whole story again from the beginning. They can challenge the present tense: “Either this is done and there is nothing I can do about it, or it’s happening now. Which is it?” and be very skeptical about answers they receive. You can easily imagine that the coercive nature of the scam requires whatever emergency situation is being simulated to be the result of actions that are over and unchangeable, but to have an effect that is immediate and present. Having targets stop and think about whether the story makes sense may be enough to simply derail the conversation, as the effect of phone scams often require that the target not feel that they have time to stop and think. (At a CISO conference in April we heard a phone scam that simply started “This is Powerful Person] this is what I need you to do for me.”)

V. CONCLUSION

Humans are unreliable detectors of deception but they represent too large an attack surface for us to simply give up on securing the human. I suggest that new forms of security training could orient members of our various organizational communities to the language features of phone scams and help them detect when an attack is occurring. Forensic linguistics, widely used in courts and law enforcement for the last thirty years for many purposes, can identify language features that cannot be simply discarded by an attacker because language requires certain operational mechanisms in order to work the way that we expect. We cannot simply arbitrarily change the rules of language to suit our purposes. Polar tag questions, question deferral, topic control, and malformed narrative sequences seem to be features of coercive phone scams worth further statistical investigation; but they are also worth the focus of training efforts to help members of our organization resist interactive phone hacking and reinforce cybersecurity.

REFERENCES

Facebook Forensics for Windows 10

Pankaj Choudhary, Upasna Singh, Nitesh K Bharadwaj, Bhupendra Singh
Department of Computer Engineering, Defence Institute of Advanced Technology (DU), Pune
{pankaj_pcse13, upasnasingh, nitesh_pcse14, bhupendra_pcse14}@diat.ac.in

Abstract—The launch of Windows 10 has attracted various forensic communities worldwide, as many new features are incorporated in it. The Universal App Platform (UAP) of new version of Windows enables the same app to run on different Windows devices. Facebook App on Windows platform is the most popular social networking app, which also attracts various criminal activities. In the past, Facebook artefacts have acknowledged as evidence in various criminal trials and may be admissible in court of law. Windows Facebook App has changed the way of forensics, as it temporarily generates SQLite databases for storing the private data of user, such as users friend list, chat messages, events, notifications, news feed and their public location information. The records maintained by these databases have the potential to provide the forensic investigator a rich source of evidences about a user’s social networking activity. In this paper, forensic artefacts of Windows 10 Facebook App are extracted from SQLite databases and their forensics importance are discussed.

Index Terms—Windows Forensics, Windows 10, Facebook Forensics, Facebook App, Universal App Platform (UAP), SQLite Database.

I. INTRODUCTION

Microsoft launched Windows 10, on 29th July, 2015, In 190 countries. Within the 24 hours of the launch, Windows 10 was installed more than 14 million devices [?]. According to Windows Central [?], more than 67 million devices are running Windows 10 now. Microsoft introduced many new feature in its new launch, including Windows 10 Apps, Edge Browser, Cortana, Notification Center, OneDrive, Universal App Platform (UAP), Quick Access, Continuum, Windows Hello etc [?].

As a digital forensics viewpoint, these features will have capability to reveal many new forensic artefacts than ever. These potential forensic artefacts can be found in event logs, internet browsers, LNK files, recycle bin, thumbnails, pagefile.sys, hyberfil.sys, OneDrive, prefetch files and Windows apps like Facebook, Twitter, Skype etc [?].

The Universal App Platform in Windows 10, enables one application to run on every Windows devices like, desktop, laptop, mobile, XBox, IoT device etc. One of the most popular social networking application is Facebook. In 2013, the official Facebook App was first appeared for Windows 8.1 and is also available for Windows 10. Almost all features are included in this app as in Facebook website.

Before the Facebook App released, only web browser traces were used as the source of Facebook related artefacts [?][?]. Facebook App forensics of Windows 8 mobile has been research in forensics community [?][?]. However, Facebook App forensics on PC running Windows 10 has changed the way of forensics, as it temporarily generates SQLite databases for storing the private data of user, such as user’s friend list, chat messages, events, notifications, news feed and their public location information.

The paper focuses on, where these databases are located on the disk, what potential forensic artefacts can be extracted from these databases and how they can contribute in an investigation. Section 2 focuses on Windows 10 Facebook App features and location of SQLite databases. Section 3 describes about the potential artefacts extracted from these databases and their importance from forensics viewpoint. The next section, concludes the paper with research challenges for further research.

II. WINDOWS 10 FACEBOOK APP

Windows Facebook App is accessible from Windows App Store for Windows 10. After installing, when a user first logs in the app with Facebook credentials, the application creates various SQLite database files for storing users private data. In Windows 10 these databases can be found on this location: C:\Users\UserName\AppData\Local\Packages\Facebook.Facebook_8xx8rvfyw5m7\LocalState\FacebookUserID\DB [?]. In the given file path, following seven SQLite database files are generated as shown in Figure 1: Analytics.sqlite, FriendRequests.sqlite, Friends.sqlite, Messages.sqlite, Notifications.sqlite, StickerPacks.sqlite and Stories.sqlite.

III. EXTRACTING ARTEFACTS FROM FACEBOOK APP DATABASES

As discussed in the previous section, each SQLite database contains a wealth of information regarding the user’s activities on the Facebook App. Our focus is to explore each database with its tables and fields and to highlight the type of information within it. This information may be very important in the case of user’s social networking activities are of forensics investigator’s interest. For examining these SQLite databases, DB Browser for SQLite [?] is used. SQLite database of Windows 10 Facebook app is shown in Table I. In the following subsection, each database with its tables and type of information in it, is discussed in detail.

1) Analytics.sqlite: Analytics.sqlite database comprises various statistics of user activity on the app, which is used by Facebook as an application feedback. It includes information such as user’s chatting activity with time of last chat. Statistics of user activity in Analytics.sqlite Database is shown in Figure 2.
This database contains following tables:

(i) analytics_log: It stores following information:
- *id*: unique identifier for users app related activity.
- *time*: time of user activity or client_event (in unix/epoch time format)
- *log_type*: type of user/client action or event
- *name*: name of user action such as page navigation, selecting bookmark
- *extra*: detailed description of users app activity such as selecting news feed, page, notification, users activity time on the app etc.

(ii) sqlite_sequence_table: Stores the value of last recorded ID of analytics_log table, for keeping track of last session

2) FriendRequests.sqlite: This database contains the detail of all pending friend requests sent to user with timestamps of these friend requests. Pending friend requests in FriendRequests.sqlite database are shown in Figure 3. It contains friend_requests with following fields:
- *uid_from*: Facebook user ID from which friend request is sent
- *time*: time when friend request is sent (in regular time format)
- *unread*: whether the friend request is seen or not (value 0 or 1)
- *name*: first and last name of Facebook user, who sent the friend request.
- *affiliations_name*: affiliation details about the user

3) Friends.sqlite: This database keeps track of user’s Facebook friends and related details. In most of the cases it stores information of all the friends of user. Freind-list information in Friends.sqlite database is shown in Figure 4. It contains friends table with following fields:
- *uid*: Facebook user ID of user’s Facebook friends
- *name*: Full name of user’s Facebook friends

---

**Fig. 1.** Location of databases of Windows 10 Facebook App

**TABLE I**

<table>
<thead>
<tr>
<th>SQLite Database</th>
<th>Tables</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Analytics.sqlite</td>
<td>analytics_log</td>
<td>Contains users analytical information, used by Facebook to get app feedback</td>
</tr>
<tr>
<td>FriendRequests.sqlite</td>
<td>friend_requests</td>
<td>Contains all the pending friend requests sent to user with timestamps</td>
</tr>
<tr>
<td>Friends.sqlite</td>
<td>friends</td>
<td>Information about users Facebook friends such as, name, e-mail, phone number, communication rank, birthday etc.</td>
</tr>
<tr>
<td>Messages.sqlite</td>
<td>folder counts</td>
<td>Contains information about cached chat messages on the device such as chat text, sender and recipient detail, timestamps, attachments, geolocation coordinates etc.</td>
</tr>
<tr>
<td></td>
<td>messages</td>
<td></td>
</tr>
<tr>
<td></td>
<td>thread_to_user</td>
<td></td>
</tr>
<tr>
<td></td>
<td>thread</td>
<td></td>
</tr>
<tr>
<td></td>
<td>users</td>
<td></td>
</tr>
<tr>
<td>Notifications.sqlite</td>
<td>notifications</td>
<td>Contains information of users Facebook notification such as likes, comments, events, birthdays etc.</td>
</tr>
<tr>
<td>Sticker Packs.sqlite</td>
<td>recent_stickers</td>
<td>Contains information of stickers used in the chat messages by the user with timestamps</td>
</tr>
<tr>
<td></td>
<td>sticker_packs</td>
<td></td>
</tr>
<tr>
<td></td>
<td>stickers</td>
<td></td>
</tr>
<tr>
<td>Stories.sqlite</td>
<td>actors</td>
<td>Stores information about users news feed and stories available on user timeline with content of stories and timestamps</td>
</tr>
<tr>
<td></td>
<td>attachments</td>
<td></td>
</tr>
<tr>
<td></td>
<td>cursors</td>
<td></td>
</tr>
<tr>
<td></td>
<td>feed_media</td>
<td></td>
</tr>
<tr>
<td></td>
<td>feed_sections</td>
<td></td>
</tr>
<tr>
<td></td>
<td>feedback</td>
<td></td>
</tr>
<tr>
<td></td>
<td>places</td>
<td></td>
</tr>
<tr>
<td></td>
<td>profiles</td>
<td></td>
</tr>
</tbody>
</table>
Fig. 2. Statistics of user activity in Analytics.sqlite Database

<table>
<thead>
<tr>
<th>id</th>
<th>time</th>
<th>log_type</th>
<th>name</th>
<th>module</th>
<th>extra</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>200</td>
<td>1440149617822</td>
<td>client_event</td>
<td>app_layout_state</td>
<td>{'state':&quot;ContentDiVideBarAndSidebar&quot;}</td>
</tr>
<tr>
<td>2</td>
<td>201</td>
<td>1440149617822</td>
<td>client_event</td>
<td>chat_turned_on</td>
<td>messages_events</td>
</tr>
<tr>
<td>3</td>
<td>202</td>
<td>1440149618166</td>
<td>client_event</td>
<td>navigation</td>
<td>page_navigation_events</td>
</tr>
<tr>
<td>4</td>
<td>203</td>
<td>1440149625479</td>
<td>client_event</td>
<td>jewel_opened</td>
<td>notification_events</td>
</tr>
<tr>
<td>5</td>
<td>204</td>
<td>144014968745</td>
<td>client_event</td>
<td>friend_request_</td>
<td>notification_events</td>
</tr>
</tbody>
</table>

Fig. 3. Pending Friend Requests in FriendRequests.sqlite Database

<table>
<thead>
<tr>
<th>uid_from</th>
<th>time</th>
<th>unread</th>
<th>name</th>
<th>affiliations_name</th>
</tr>
</thead>
<tbody>
<tr>
<td>10009929323023</td>
<td>2015-07-29 12:46:47</td>
<td>0</td>
<td>Aakash Thakur</td>
<td>[]</td>
</tr>
<tr>
<td>2009955421995</td>
<td>2015-07-12 16:09:30</td>
<td>0</td>
<td>Bharti Pande</td>
<td>[]</td>
</tr>
<tr>
<td>30092729049933</td>
<td>2015-03-01 23:30:10</td>
<td>0</td>
<td>Penkaj Choudhry</td>
<td>[]</td>
</tr>
<tr>
<td>40007914259044</td>
<td>2014-12-26 11:04:26</td>
<td>0</td>
<td>Sinu Choudhry</td>
<td>[]</td>
</tr>
<tr>
<td>5000706025610</td>
<td>2014-10-23 05:25:00</td>
<td>0</td>
<td>Ajay Narwal</td>
<td>[]</td>
</tr>
</tbody>
</table>

Fig. 4. Freind-list Information in Friends.sqlite Database

- **contact_email**: Email addresses of user’s Facebook friends
- **phones**: Detail of phone number of user’s Facebook friends with country code and type of phone such as cell phones
- **is_pushable**: Information about whether or not the user can receive push notifications (value 0 or 1)
- **has_messenger**: whether or not the user is using Facebook messenger app (value 0 or 1)
- **communication_rank**: Ranking or rating of Facebook users based on the recent interactions. It represents, how often the user communicates with particular Facebook friend.

- **birthday_date**: birthday date of user’s friend (if shared publicly)

4) **Messages.sqlite**: This database contains information about cached chat messages on the device such as, chat text, sender and recipient detail, timestamps, attachments, geolocation coordinates etc. It stores all the messages which are currently loaded in the Facebook App. The database contains following table:

(i) **folder_count**: This table contains following fields:
- **folder**: type of folder for storing message such as Inbox
- **unread_count**: total number of unread messages in the chat inbox
- **unseen_count**: total number of unseen messages in the
Chat messages information in folder_count table of Messages.sqlite database is shown in Figure 5.

(ii) messages: This table contains following informations:

- id: unique identifier for each message instance in inbox. These IDs follow different structures as Facebook updates its message ID structure time to time. For example, m_mid.<timestamp>;<hex hash value> is the new message ID structure and m_id. <numeric value> is old message ID structure.
- thread_id: identifier for the message thread. It can be used to identify the message thread or conversation of user with friends and groups. It contains a timestamp (in unix/epoch time) of message thread generation, in other words, it is the time of users first chat message with particular friend.
- body: It contains the original chat message between Facebook users or group in text.
- sender: It stores the details of sender of the particular chat message. It contains Facebook user ID, name and email of the sender.
- tags: It contains information related to message storage location, messaging app used, status of message sent or received etc.
- timestamp: It stores the server timestamp (in unix/epoch time) value of the message
- action_id: Identifier, composed of a timestamp (in unix/epoch time) value of last messaging related action.
- coordinates: It stores the geolocation coordinates of the message sender/receiver, in case of GPS sensor is used.
- attachments: It contains details of attachments, if attached with chat message. It includes attached file name, unique ID if attached image, thumbnail and image URL of attached image etc.
- shares: It contains detail of, if any link or URL is shared in the chat message.
- type: Type of chat message such as personal or group message.
- server_timestamp: Timestamp (in unix/epoch time) of particular message_id on the message server.

Chat messages information in messages table of Messages.sqlite database is shown in Figure 6.

(iii) thread_to_user: This table stores the detail of individual chat message thread of user. It stores following information:

- t_id: Message thread ID of, unique for particular chat message thread.
- u_id: Facebook user ID of all Facebook users involved in particular message thread.
- is_participant: Whether or not user participated in the particular message thread.
- read_recipient_timestamp: Timestamp (in unix/epoch time) to acknowledge, when recipient of the message reads it.

Individual chat message thread of user in thread_to_user table of Messages.sqlite database is shown in Figure 7.

(iv) thread: This table keeps track of each Facebook message thread such as timestamps, sender and receiver details, number of messages in the thread etc. It stores following information:

- thread_id: Unique identifier for all message threads
- timestamp: Timestamp (in unix/epoch time) of last message instance within the particular message thread.
- action_id: Unique identifier consisting of a timestamp (in unix/epoch time) value of last action related to message thread.
- name: Name of message thread, if any
- folder: folder within the app, where particular thread is stored, e.g. inbox
- snippet: Text of last chat message in the particular message thread.
- snippet_sender_id: Facebook user ID of user for last sent message with respect to particular thread.
- senders: Facebook user ID of all the participants in the personal or group conversation for particular message thread.
- num_messages: Total number of messages in the particular message thread.
- single_recipient: Facebook user ID of the recipient of the message.
- can_reply: whether or not the user can reply in the particular message thread (value 0 or 1)
- is_subscribed: whether or not the user is subscribed to get notifications for particular message thread (value 0 or 1)
- draft: Whether message is stored in the draft, in case of not sent.

Message thread information in thread table of Messages.sqlite database is shown in Figure 8.

(v) users: This table stores the detail of all participants involved in Facebook chat with user. It contains following information

- id: Facebook user ID of all participants involved in Facebook chat with user.
- type: Type of user such as Facebook user, page or group.
- email: Facebook user email ID
- name: full name of all Facebook users involved in chat with user.
- last_active_timestamp: Last activity timestamp (in unix/epoch time) of Facebook users involved in conversation with user.

Chat participants in users table of Messages.sqlite database are shown in Figure 9.
5) **Notifications.sqlite**: This database contains information of user’s last received Facebook notification such as likes, comments, events, birthdays, page and group posts etc. It contains **notifications** table storing following information about user’s notifications:

- **notification_id**: Unique identifier, consisting of a timestamp (in unix/epoch time) value of particular user notification of Facebook App.
- **object_id**: unique identifier for objects involved in the notification.
- **object_type**: type of objects in the notification such as photos, page, group, birthday reminder, event, stream etc.
- **sender_id**: Facebook user ID of the sender of current notification.
- **title_text**: Text content of the notification.
- **href**: reference of the page, user, group or event for which notification is generated.
- **unread**: Whether or not the notification is read (value 0 or 1)
- **updated_time**: last updated time of the notification in the app.
- **created_time**: time when notification is first created.

Last received notifications in Notifications.sqlite database are shown in Figure 10.

6) **StickerPacks.sqlite**: This database contains information of stickers used in the chat messages by the user with various timestamp values. These timestamp values are very essential, when stickers are used in chat messages. It contains following tables:

(i) **recent_stickers**: It stores following information:

- **uid**: unique identifier of individual sticker.
- **added_datetime**: Time when sticker is added with particular chat message.

(ii) **sticker_packs**: This table contains information about particular sticker pack such as unique id, name of sticker pack, artist, date and time of adding sticker pack etc.

(iii) **stickers**: This table contains the detail of cached stickers in the chat messaging window.
7) **Stories.sqlite**: This database stores information about users' news feed and stories available on user timeline. It is very important as it stores the content and timestamp of the users' news feed and the current activity of user can be captured from this. It stores following tables:

(i) **actors**: This table keeps track of Facebook users, pages and groups involved in the particular news feed or story. Facebook users involved in news feed in the `actors` table of Stories.sqlite database are shown in Figure 11.

(ii) **attachments**: It keeps track of all the attachments used in particular news feed or story such as story id, author of the story, details of like, comments and shares for particular story, post creation time, text of post, url of shared images, videos and files, target with respect to story.

Attachments used in news feed in the `attachments` table of Stories.sqlite database are shown in Figure 12.

(iii) **feed_media**: This table contains the information images, videos and other media items shared in the news feed. It may help in finding media details such as url, created and modified time of media etc.

(iv) **feedback**: It keeps track of various information for particular posts such as: likes, comments, shares etc. It contains following fields:

- **id**: unique identifier with respect to particular Facebook post.
- **can_viewer_comment**: Whether or not the Facebook
user can comment on the particular post (value 0 or 1).

- **can_view_like**: Whether or not the Facebook user can like on the particular post (value 0 or 1).
- **does_view_like**: Whether user liked the particular post (value 0 or 1).
- **legacy_api_post_id**: Facebook id to identify and visit particular post.
- **comments**: Total number of comments for particular Facebook post.
- **likes**: Total number of likes for particular Facebook post.
- **reshares**: How many times a particular Facebook post is reshared.

Feedback information in **feedback** table of Stories.sqlite database is shown in Figure 13.

(vi) **places**: This table keeps track of location and places shared in the news feed. It may be helpful for tracking person based on shared places. It stores following information:

- **id**: Facebook id of shared place, location page in the post
- **facebook_sub_type**: type of place such as page, free form place, location etc.
- **address**: address of the shared location (if available)
- **category_names**: category of location or place such as shopping mall, city, restaurant, historic or tourist place, home etc.
- **contextual_name**: name of shared place such as Mumbai, waterfall, school, peoples mall etc.
- **description**: description of the place
- **location**: coordinates and time zone details of shared place.
- **url**: Facebook url of the shared place.

Shared places and locations in news feed in **places** table of Stories.sqlite database are shown in Figure 14.

(vii) **profiles**: It contains the detail of user profiles, pages and groups involved in the news feed. It may help in identifying users, pages, groups, events for particular post. It stores following information:

- **id**: Facebook ID of user, page, group or events.
- **facebook_sub_type**: whether user, page, group or event.
- **name**: name of Facebook user, page, group or event.
- **url**: url of Facebook user, page, group or event.
- **friend_request_status**: Whether or not the user is friend with particular Facebook user.
- **is_owned**: Whether user liked or subscribed to particular page or group.
- **mutual_friends**: number of mutual friends between user and other users.

User profiles, pages and groups in **profiles** table of Stories.sqlite database are shown in Figure 15.
Fig. 13. Feedback Information in feedback Table of Stories.sqlite Database

Fig. 14. Shared Places in News Feed in places Table of Stories.sqlite Database

Fig. 15. User Profiles, Pages and Groups in profiles Table of Stories.sqlite Database

- **id**: Facebook ID of particular news feed or story.
- **hidden**: Whether or not post is hidden (value 0 or 1)
- **can_viewer_delete**: Whether or not user has permission to delete the particular post.
- **can_viewer_edit**: Whether or not user has permission to edit the particular post.
- **creation_time**: Timestamp (in unix/epoch time) of creation of the post
- **url**: Facebook url of the post
- **to**: To which page or group the post is shared.
- **message**: Original text of the post.
- **explicit_place_id**: Facebook Id of the exact place such as airport, station or Mall shared in the post.
- **implicite_place_id**: Facebook id of the place where explicit place is located.
- **privacy_scope**: privacy scope of the post such as public, private or custom.
- **edit_history**: Number of times the particular post has been edited.

All stories in stories table of Stories.sqlite database are shown in Figure 16.

IV. CONCLUSION AND FUTURE RESEARCH

Windows 10 was released with lot of new features such as Edge Browser, Cortana, Notification Center, OneDrive, Universal App Platform (UAP), Windows 10 Apps (Facebook, Twitter, mail), Quick Access, Continuum, new Start menu etc. These features discover treasure new forensic artefacts, useful in forensic investigation of devices running Windows 10. As a source of digital evidence against various digital crimes,
Facebook artefacts are being used by various law enforcement agencies and administrations. Windows version of Facebook App was first appeared for Windows 8.1 and is also available for Windows 10, involving same features as in Facebook website. The Universal App Platform in Windows 10, enables this app to run on all Windows devices like desktop, laptop, mobile, XBox, IoT device etc. Windows Facebook App has transformed the direction of forensics, as it’s SQLite database structure for storing the private data of user. However, as of now, limited research has been done in this domain.

In this paper, we discussed about the location of seven databases generated by Facebook App in Windows 10, potential social networking artefacts extracted from these databases, and their importance in forensics investigation. As a forensics point of view, information extracted from these databases, such as users friend list, chat messages, events, notifications, news feed and their public location information, are important and may assist in an investigation. The paper does not cover the recovery of deleted SQLite database, however, the carving of deleted data from SQLite database is a current forensics challenge. Further research may be focused on this issue.

REFERENCES


Covert Channel in HTTP User-Agents

Susan Heilman
B. Thomas Golisano College of Computing and Information Sciences
Rochester Institute of Technology
Rochester, New York 14623
sih4146@rit.edu

Jonathan Williams
B. Thomas Golisano College of Computing and Information Sciences
Rochester Institute of Technology
Rochester, New York 14623
jpw2408@rit.edu

Daryl Johnson
B. Thomas Golisano College of Computing and Information Sciences
Rochester Institute of Technology
Rochester, New York 14623
dgjics@rit.edu

Abstract—A subliminal covert channel establishes a nearly undetectable communication session within a pre-established data stream between two separate entities. This document explains how HTTP can be utilized to facilitate a covert channel over both local and wide area networks. The Hypertext Transfer Protocol (HTTP) accounts for a majority of the Internet’s daily web traffic and is permitted within almost all network topologies. Therefore, HTTP is a prime medium for hiding messages and information communicated between separate parties. This paper illustrates a new approach to covertly encoding messages in the HTTP message through use of the User-Agent and referer strings in the HTTP Request Header.

Index Terms—Covert, Covert Channel, HTTP, User-Agent, HTTP Request

I. INTRODUCTION

Research in the field of covert communications and covert channels has grown steadily since the topic was first introduced in the 1970s. Four decades later the Internet has expanded to include several hundred networking and application-based protocols. This means that data and information can be hidden in almost any level of the Internet’s hierarchical structure. One of the most commonly used Internet protocols is the Hypertext Transfer Protocol, abbreviated HTTP for short. This protocol is responsible for the delivery of web pages and web content to a particular user or device. With 3,585,000,000 users in just North America and over 8.7 billion devices connected to the Internet, thousands of HTTP requests and HTTP responses are generated every minute accounting for nearly half of all traffic on the Internet [1]. This opens up a myriad of possible places to store hidden data as a means of communication. Some individuals have already discovered different methods hiding messages using the HTTP protocol. The covert channel that is proposed below provides a new way of hiding multiple messages in the Hypertext Transfer Protocol by manipulating fields within the HTTP request header. In some cases modifying the HTTP request header can be problematic if certain network security or web server security mechanisms are in place. When properly deployed these mechanisms may possibly cause the HTTP request to not be recorded into a security access log. The proposed covert channel manipulates areas of the HTTPS request header that have been known to not be kept under surveillance or are not policed often. This characteristic of the covert channel, among many others, increases the likelihood that the covert message will successfully reach the intended recipient.

In the following text below you will be introduced to the definition of what a covert channel consists of and a summary of the Hypertext Transfer Protocol. Next, there is a demonstration of how the proposed covert channel hides a message within the HTTP request header with a detailed analysis of the channel’s design and properties followed by data that illustrates a proof of concept of the covert channel on a local network and over the Internet. Finally, there is an analysis of the proposed covert channel to measure the channel’s effectiveness as well as examine its disadvantages.

II. COVERT CHANNELS

A covert channel uses the structure of an existing medium to send and receive small parts of data through unused portions of the medium. A covert channel hides in the nooks and crannies of the formal structure of the operation and is virtually undetectable. In Lampson’s famous paper [2] he coins the idea of covert channels by describing them as “those not intended for information transfer at all, such as the service program’s effect on the system load”. For our purposes, we rely on the dismissive behavior towards HTTP User-Agents and referer fields to hide discrete messages in the User-Agent string.

A. Characteristics of Covert Channel

The characteristics of a covert channel help define and gauge the effectiveness of a given covert channel when it is being used. There are three main characteristics that establish the foundation for a covert channel which are stealthiness, followed by bandwidth and last timing. In the subsections below each individual characteristic will be further explained.

1) Stealthiness: If a covert channel is described as stealthy it means that the channel does not provide an obvious view of the conversation to outside parties. In essence, if the two parties of the covert communication session were to have their network traffic monitored no flags or warnings would be generated by the monitoring party. It is as if the communication is not taking place at all.

2) Bandwidth: Bandwidth is a very sensitive factor when developing a reliable and robust covert channel. Bandwidth refers to the ratio of total messages sent in a given unit of time. Typically with overt channels a higher bandwidth rate is
B. Types of Covert Channels

Below are five of the commonly known ways to implement a covert communication channel into a Internet networking or application-based protocol.

1) Steganographic: Steganographic channels involve hidden covert information within a overt information medium. Steganography on the Internet usually entails hiding a covert message or photo within another image file like the .jpg or .png file extensions. [3].

2) Subliminal: Subliminal Covert channels are similar to Steganographic channels in that it hides a message within a normal pre-existing channel. However, when compared the subliminal channel and the overt channel are virtually indistinguishable from each other when compared against each other. [4].

3) Behavioral: A behavior-based covert channel is defined as a communication channel achieved by modulating the internal states of the sender or receiver by purposely selecting certain inputs to the systems. [5], [4]

4) Storage: Storage channels involve directly or indirectly writing to a storage location by one application and the directly or indirectly reading the bits of a storage location by another application. This is different from regular file reading and writing information because a storage channel will write information to the file in a way that communicates a secret message instead of reading the intended information that the file is storing [3].

5) Timing: Timing channels work by one party signaling the other party with a message via the modulation of system resources that cause a change in response time that is noticed by the second party. A clock value or time measurement would convey the value being sent. For example, the clock or time measurement would be altered by the first party so as to communicate with the second party [3].

III. RELATED WORK

Applications exist to generate covert messages using steganography. Snow [6], for example, uses steganography as its means of covertly communicating messages from one party to another by relying on whitespace - the spaces and tabs that naturally appear in writing that are invisible to the human eye when displayed at the end of lines in text viewing applications. This way, the hidden message can be placed in regular ASCII without modifying the original message’s appearance.

On the other hand, [7] hides messages via steganography by mapping known ASCII characters to unused ASCII symbols that are visibly indistinguishable to the human eye.

Moving to the HTTP protocol, a covert channel can exist in HTTP/1.1 by manipulating the Query_String field when a HTTP redirect is executed. Such examples have been demonstrated by [8] through HTTP redirects.

IV. HTTP: HYPERTEXT TRANSFER PROTOCOL

The Hypertext Transfer Protocol was first standardized for use in June of 1999. The revolutionary protocol provided a simple and scalable hypertext and media delivery solution. HTTP is a stateless protocol with an open ended set of methods and headers that indicate the purpose of each HTTP request. [9]. HTTP request chains and response chains are sent between a User-Agent and the origin server. Participants in the HTTP request and response chains can be known to initiate multiple simultaneous communications. This same protocol is used in other application layer information delivery mechanisms such as the FTP and SMTP Internet protocols.

The combination of widespread HTTP 1.1 use and the HTTP 1.1 design qualifies the Hypertext Transfer Protocol to be the best candidate for a robust, reliable covert channel. Because of the HTTP standard communicating properties a large number of requests or replies may be generated to facilitate a minimal request. This covert channel uses this HTTP communication property to its advantage in order to send multiple characters of a message to a HTTP origin server where it will be received and processed into a human readable format.

A. The HTTP Message Format

HTTP messages exist in two different types; They are either requests or responses. In a typical environment, the client is the one sending requests and receiving responses while the server receives requests and sends responses. It is not likely for a client to send HTTP responses, which is why the proposed covert channel avoids this interaction. The format for an HTTP message (without specifying request or response) follows this structure:

```
<start-line>
<message-headers>
<empty-line>
[<message-body>]
[<message-trailers>]
```

To get a better understanding of where the proposed message location fits into the HTTP message, a breakdown of the HTTP request message is needed. Figure 1 visually explains where the User-Agent string fits into the request.

The HTTP request “Host” variable shows the URL that is being requested. The “User-Agent” field is filled by the browser that is being used to send the request. The above User-Agent was filled by a Mozilla Firefox browser. This is where the message is hidden. An additional field to note in order to better understand the covert channel is the optional “referer” field that fits into the Request Header. This field identifies the original address of the webpage from which the current resource was linked. The referer field is typically populated.
when the user clicks on a link on a webpage. The webpage that had the link is referring the user to the new link, so the address of the first page is put in the referer field [10].

For the purpose of this covert channel, the HTTP response is not a concern. The HTTP request solely exists to confirm that the HTTP request sent in the channel was successful in reaching the server.

V. A COVERT CHANNEL IN THE USER-AGENT STRING

The proposed covert channel works by altering the User-Agent string that is present in the HTTP request header. In order for the message to be covertly communicated through this string, the message must undergo changes and mapping before being relayed to the server.

A. Encoding

The ASCII message entered by the user is converted to hexadecimal which is then hidden by modulating the User-Agent string. The chosen User-Agent string is a commonly seen User-Agent of "Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/46.0.2490.71 Safari/537.36." In this string there are several white spaces that are utilized to hide the message. In order to send the “0” hexadecimal value to the receiving server, the first white space that appears in the chosen User-Agent string is doubled. That is to say, instead of there being one white space, “ “, there becomes two, “ “. This is very difficult for the naked eye to notice. To convey a “1”, the second available white space is doubled, and so on. Figure 2 outlines which hexadecimal value corresponds to the extra space or spaces. The highlighted values that contain two hyphens are the areas where the extra spaces can be found.

Common User-Agent validation tools use regular expressions to parse the User-Agent field to detect bots and other malicious requests. If User-Agent validation tools were used to parse our selected User-Agent string it would most likely focus on the "Chrome/46.0.2490.71" substring. To avoid running the risk of throwing an error, we avoided manipulating this specific substring of the User-Agent field.

B. Design

The implementation of the proposed covert channel relies on the following components:

• Server – Acts as message receiver
• Client – Acts as message sender
• Linux "curl" command
• Linux "xxd" command
• Defined table mapped to hexadecimal characters (See Figure 2)

The channel works in the following manner: A server running Apache with port 80 open awaits a message. The client machine or sender runs the client.sh script that takes input from the user. The user input is then converted from an ASCII message to a hexadecimal string, and depending on the hexadecimal character a different User-Agent string is crafted as outlined in Figure 2. Once the User-Agent string is crafted with the necessary added spaces, the script executes a curl command to send an HTTP request to the server. The server response to the curl request is irrelevant to the channel for two reasons. Firstly, the client.sh script is setup to only send messages to a specified IP address and will discard the response. Secondly, even if the receiving server processing the curl request and does not provide a usable resource to the user encoded message characters are still logged and stored in the access.log file.

The “-A” and “-e” flags for curl are used to specify the User-Agent string and the referer field, respectively. The purpose of
Fig. 2: User-Agent to Character Mapping Table. Note, the denotation of yellow in this table is to provide visual aid to a notation of a double dash marking, and does not reflect any meaning to the content provided.

<table>
<thead>
<tr>
<th>Hexadecimal Value</th>
<th>User-Agent String with Hidden Component (denoted by two dashes “-“ and highlighting)</th>
<th>Bit Location in User-Agent String</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/64.0.3290.71 Safari/537.36</td>
<td>1s</td>
</tr>
<tr>
<td>1</td>
<td>Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/64.0.3290.71 Safari/537.36</td>
<td>2s</td>
</tr>
<tr>
<td>2</td>
<td>Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/64.0.3290.71 Safari/537.36</td>
<td>3s</td>
</tr>
<tr>
<td>3</td>
<td>Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/64.0.3290.71 Safari/537.36</td>
<td>4s</td>
</tr>
<tr>
<td>4</td>
<td>Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/64.0.3290.71 Safari/537.36</td>
<td>5s</td>
</tr>
<tr>
<td>5</td>
<td>Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/64.0.3290.71 Safari/537.36</td>
<td>6s</td>
</tr>
<tr>
<td>6</td>
<td>Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/64.0.3290.71 Safari/537.36</td>
<td>7s</td>
</tr>
<tr>
<td>7</td>
<td>Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/64.0.3290.71 Safari/537.36</td>
<td>8s</td>
</tr>
<tr>
<td>8</td>
<td>Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/64.0.3290.71 Safari/537.36</td>
<td>15&amp;11.5s</td>
</tr>
<tr>
<td>9</td>
<td>Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/64.0.3290.71 Safari/537.36</td>
<td>15&amp;21.5s</td>
</tr>
<tr>
<td>A</td>
<td>Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/64.0.3290.71 Safari/537.36</td>
<td>15&amp;31.5s</td>
</tr>
<tr>
<td>B</td>
<td>Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/64.0.3290.71 Safari/537.36</td>
<td>15&amp;41.5s</td>
</tr>
<tr>
<td>C</td>
<td>Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/64.0.3290.71 Safari/537.36</td>
<td>15&amp;51.5s</td>
</tr>
<tr>
<td>D</td>
<td>Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/64.0.3290.71 Safari/537.36</td>
<td>25&amp;11.5s</td>
</tr>
<tr>
<td>E</td>
<td>Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/64.0.3290.71 Safari/537.36</td>
<td>25&amp;21.5s</td>
</tr>
<tr>
<td>F</td>
<td>Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/64.0.3290.71 Safari/537.36</td>
<td>25&amp;31.5s</td>
</tr>
</tbody>
</table>

The referer field for the proposed covert channel is to signify when a message is at the beginning, middle, or end stage. In other words, the referer field of the HTTP request header is a delimiter that is used by the receiver to distinguish between the start and stop of messages.

The client will deliver one HTTP request for each character sent. Once the client has completed sending the server has received and processed numerous requests with seemingly normal parameters that are stored in Apache access log. For Apache, the default path is “/var/log/apache2/access.log”. Once the client has sent the message through HTTP requests with the curl command, the server.sh script need only pull the message from the access log. The script that is run on the server reads in the access.log file and performs a series of checks to search for the double spaces and reassembles the message in hexadecimal. Finally, the message is then converted from hexadecimal to ASCII.

This concludes the sending and receiving of a message over
C. Testing

In order to test the proposed covert channel implementation, a virtual machine running a Linux OS was created. Apache was installed and configured on this server to ensure that the client/sender would have some webpage to which the curl commands could send messages. The sender script was run from a Bash terminal window which prompted the user for a desired message that the script would then convert to hexadecimal prior to sending. Once the sender script finished its execution, the receiver script was run against the apache2 access.log file which converted the delivered message back to ASCII before printing it to Standard Output.

The rate at which data can be sent through the channel was tested by determining how many HTTP requests the server could process and log from the same client. In order to ensure that all requests would be logged on the server side, which also ensures that the entirety of the message is logged, sleep timers were used to guarantee that the requests were sent consecutively and in the correct order.

Further testing was performed by sending the proposed covert channel through a commonly used proxy service. By modifying the curl command to include proxy support, the sender.sh script forwarded the encoded covert traffic to a squid3 proxy server that used the ‘combined’ logging setting. When the message was sent from the client each hexadecimal encoded character was received in order in the '/var/log/squid3/access.log'. To read the message in this case the receiver script was modified to parse the squid access.log rather than the '/var/log/apache2/access.log' file. Considering that the proxy service stores the User-Agent and referer fields by default in its access.log, our covert channel can effectively deliver its message to a HTTP proxy log file as well.

VI. QUALITIES AND PROPERTIES OF THE CHANNEL

A. Categorization

Covert channels exist in various classifications and implementations. The proposed covert channel is best described as a subliminal covert channel. A subliminal channel hides its secret message in a normal channel’s medium. The subliminal channel and regular channel are indistinguishable without prior knowledge of the secret. The proposed channel’s secret is the hexadecimal conversion and its mapping to different User-Agent strings.

B. Stealthiness

The scripts written for the client and server are written in the command language, Bash. The Bash programming language was selected because nearly all Linux distributions are capable, at the very least, of running a bash shell. Furthermore, all commands and string manipulation used within the scripts is done solely in bash without relying on any interpretation other than bash. This is done through regular expressions and bash commands such as ‘curl’, ‘tr’ (trim) and ‘xxd’ (to convert to and from hexadecimal).

C. Data Rate

As mentioned in the Testing section, a long message means many ‘curl’ command executions to send many HTTP requests to the server. The proposed channel uses a one second delay to avoid loss of data transmission between the sender and receiver. With this in mind, the two parties communicating are not severely limited by the length of the message to be sent.

D. Robustness

The proposed covert channel is transmitted through one of the most commonly seen protocols in the wild: HTTP. In addition to its popularity, HTTP is rarely monitored or policed for discrepancies such as a referer field manipulation. It is also routable over wide area networks which allows the sender and receiver to be geographically separated.

E. Potential Issues

The scenario for which the covert channel must be successful is commonly available but relies on certain key aspects.
• There must be an existing and running web service with an open port, namely, Apache since the proposed channel searches in the folder path for Apache’s log file.
• If the server has disabled logging then there is no access log for the message to be sent.
• The rate at which HTTP requests can hit the server is also an area of caution for the covert channel. If the Apache server sees too many requests from on particular client within a given time value, the server may ignore the requests.

F. Benefits
Thought simplistic in nature, the covert channel contains design specifications that enhance its usability and stealthiness. HTTP traffic is common and unlikely to be flagged by a firewall, especially since the covert channel uses legitimate User-Agent strings. By using whitespace to code the message, the communication is obfuscated in the log file and nearly undetectable to the human eye. The script was intentionally written in Bash due to the language’s popularity across Linux distributions (and now in Microsoft’s newest Windows 10 update). The script also contains the minimal number of commands needed to avoid detection in respect to system resources. Overall, the decisions made to use Bash and limited commands was done to avoid dependencies across different operating systems. Finally, the issue of HTTP encryption (SSL/TLS) is moot since the receiver script is run server side.

VII. Future Work
Some future improvements to the proposed covert channel include dynamic User-Agent white space manipulation and a feature where HTTP requests can be tailored to the web page being requested. Rather than having to manually add the extra white space characters into the User-Agent string for each character in the hexadecimal encoding, the code would incorporate a looping feature that would automatically detect spaces in the User-Agent string and add additional spaces where necessary. This may be more complicated to implement, however it will allow the proposed covert channel to be used with other future User-Agent strings if a particular User-Agent becomes flagged as malicious or unauthorized. The other feature that will be implemented into the proposed covert channel involves using the URLs that are returned from the first 'curl' command to legitimized the web traffic that is being sent to the receiving server. After the first 'curl' command returns the web-page’s HTML text it will parse through the returned HTML text line by line searching for any ‘HREF’ links that are on the web-page. Once all of the ‘HREF’ links have been stored into an array the following hexadecimal characters that are transmitted to the receiver will cycle through and use one of the array entries of web-page links to help legitimized the web traffic, thus making the covert channel harder to detect.

VIII. Conclusion
This paper introduced an implementation of a covert channel that utilizes the User-Agent string in the HTTP Request Header. In order to fully detail the mechanism of this covert channel, the paper first discuss the various types of covert channels, the characteristics a covert channel possess, and the the basis of the HTTP Message Format. This covert channel mimics stealthy behavior by relying minimally on system resources and calling up the most base Linux shell and command language. Furthermore, the transmission of the message is carried through a widely and abundantly used protocol: HTTP. The use of HTTP not only allows this channel to hide within network traffic, but it stretches across wide area networks to allow the two parties in the channel to be virtually anywhere in the world. As more and more business is conducted online, the availability of this HTTP User-Agent channel has the potential to become more widespread, accessible, and available to users capable of running the Bash shell.

References
Don’t See Me, Just Edit Me: Towards Secure Cloud-based Video Editing

Odd-Arild Kristensen
State University of New York, Albany
NY, USA 12222
Email: kristensen.odd@gmail.com

Manoranjan Mohanty
New York University Abu Dhabi
Abu Dhabi, UAE 129188
Email: manoranjan.mohanty@nyu.edu

Pradeep K. Atrey
State university of New York, Albany
NY, USA 12222
Email: patrey@albany.edu

Abstract—With the evolution of cloud computing, individuals and organizations are processing video in cloud datacenters. Although this cloud-based video processing has many advantages, security and privacy is a concern. In this paper, we address this concern by proposing a Shamir’s secret sharing-based secure video scaling/cropping scheme that allows datacenters to scale and crop the encrypted video without learning the content. The scaled/cropped video is also in encrypted form. Therefore, the datacenters cannot learn important information contained in video. An authorized user, however, can recover the secret scaled/cropped video from the encrypted scaled/cropped video files from a specific number datacenters.

Index Terms—Cloud Computing, Security, Video Editing

I. INTRODUCTION

Cloud computing is rapidly becoming ubiquitous. Individuals and organizations are increasingly storing and processing data in cloud data centers (CDCs). Computationally expensive operations are ideally suited for cloud computing, and as such, outsourcing expensive video editing operations to CDCs is a cost-effective alternative to local processing [1].

Although moving data to a third-party cloud has many benefits, the security and privacy of data are of serious concerns. Several recent data breaches indicate that users data is not always secure when stored by a third-party service provider. Companies could be the victims of a breach and the attackers could steal or tamper with the data. Additionally, there could be insider attacks at the CDCs.

These security concerns become significant when the video data is of a sensitive nature. With a huge amount of video data emerging from different applications (such as surveillance cameras and smart phones) being outsourced to cloud, it is important that video processing operations performed by cloud services are secure. Our focus in this paper is on securing two video processing operations: scaling and cropping. While traditional cryptographic schemes provide computational security, those that lack homomorphic properties force users to decrypt the video before processing it, thus exposing the data to increased risk. Our aim is to process video in encrypted domain.

The challenge then becomes to construct a method that allows users to encrypt their video before uploading it to cloud services where it can be safely manipulated in encrypted form. Theoretically, we can achieve this goal using a fully homomorphic scheme. For example, if video editing operations $o$ is performed (such as addition, subtraction, multiplication and division), then we can then edit videos in encrypted domain using a homomorphic scheme that holds $E(v_1 \circ v_2) = E(v_1) \circ E(v_2)$, where $v_1$ and $v_2$ represent the two data units in a video (e.g. two frames or two pixels in a frame) and $E$ denotes the encryption function. However, fully homomorphic schemes are not still practical [2].

In this paper, we propose a practical encrypted-domain video scaling/cropping scheme using Shamir’s Secret Sharing (SSS) [3]. We utilize the additive and scalar multiplicative homomorphic properties of SSS [4] to bicubically scale the video in encrypted domain. For each frame in a video, we process byte-by-byte and encrypt each byte, thus creating shares, or shadow copies, of the video. Zooming into a frame is done by first cropping the frame and then scaling it. To the best of our knowledge, this is the first work on performing cropping and scaling of a video in encrypted domain. The overheads in our scheme is still high, and we are working to improve it.

The rest of this paper is outlined as follows: Section II presents prior work on this topic. Section III describes our proposed work along with its assumptions, with Section IV presenting our security analysis and our results. The conclusion and future improvements are discussed in Section V.

II. BACKGROUND AND RELATED WORK

A. SSS scheme

The $(k,n)$ SSS scheme, $2 \leq k \leq n$, was proposed by Adi Shamir in 1979 [3], it divides a secret into $n$ shares such that: i) any $k$ or more shares can reconstruct the secret, and ii) $k-1$ or fewer shares cannot reconstruct the secret.

To share a secret $a_0$ among $n$ participants, a polynomial function $F(x)$ is constructed of degree $k-1$ using $k-1$ random coefficients $a_1, a_2 \ldots a_{k-1}$ in a finite field $GF(p)$ where $p$ is a prime number $> a_i, 0 \leq i \leq k-1$.

$$F(x) = \sum_{i=0}^{n} a_i x^i \mod p$$

Any $k$ out of $n$ shares can reconstruct the secret using Lagrange interpolation; the secret can be obtained at $L(0)$ i.e.
\[ L(0) = a_0, \text{ as:} \]

\[ L(x) = \sum_{j=1}^{k} \prod_{i=1, i \neq j}^{k} \frac{x - x_i}{x_j - x_i} \mod p \quad (2) \]

The SSS scheme has been used to protect different types of data such as text, image, and video [5]. There have been efforts to limit the storage requirements using a variation of the SSS scheme, called Ramp Secret Sharing. While these schemes reduce the space required to store the shares, the scheme loses its prefect security property, thus making it vulnerable to attacks.

There have been a few attempts to process image data in encrypted domain using SSS’s homomorphic property. Examples include: Mohanty et al. [6] for image scaling and cropping, and Lathey and Atrey [7], whose work is an extension of [6]. While in [6], the authors performed scaling and cropping on encrypted images, Lathey and Atrey attempted to perform quality enhancement operations on encrypted images.

B. Encrypted Domain Video Processing

There are several works on the topic of encrypted domain video processing. For instance, Upmanyu et al. [8] proposed secure solutions for applications such as blind authentication, i.e. blindly authenticating a remote-user using his biometric, object tracking, and face detection over cloud. They presented a secure framework for carrying out visual surveillance on random video streams at remote servers. Similarly, Avidan and Butman [9] presented a method to securely perform face detection without leaking vital information about the image. Lu et al. [10] discussed various applications of video processing in encrypted domain, and the challenges and potential techniques, focusing on video search, classification and summarization. Our focus on video scaling and cropping is a novel contribution to this area.

C. Bicubic Scaling

For scaling of video frames, we used the bicubic interpolation method [11], which takes 16 reference points and interpolate them. This scaling algorithm generally works well for upscaling images while still maintaining relatively high levels of detail. Mathematically, it is expressed as:

\[ P(x, y) = \sum_{i=0}^{3} \sum_{j=0}^{3} a_{ij} x^i y^j \quad (3) \]

Here, \( x \) and \( y \) denote the position of the interpolated value, \( a_{ij} \) is the value of the \( ij^{th} \) sub pixel in the targeted image, and \( P(x, y) \) is the interpolated pixel value.

III. PROPOSED WORK

A. Threat Model and Assumptions

Cloud-based video editing can be vulnerable to a slew of security threats. The following entities are involved in the process: the owner of the video, video transmission channels, CDCs, and users of the video. Here CDCs are assumed to be honest but curious in a sense that they honestly perform operations but can try to learn secret information from video. The owner and the user of the video can have full access to the video, and hence, are regarded as trustworthy.

With these vulnerabilities, we need a solution that assumes that the data is at risk as soon as it is not in the possession of the user. To achieve the best security possible, all CDCs should be unrelated and any \( (k \leq n) \) of them must not collude. We assume that both the input and output will be in the mpeg compression format. For convenience and to maintain data integrity, all video editing operations should be nondestructive so as not to change the input file. For simplicity, we assume that all operations will be performed on the entire video and not only for selected frames.

B. Architecture

In order to not to expose the secret video to unnecessary risks, all encryption and decryption should happen on the user’s end. The CDCs should not possess the capability to perform any of these operations. We address this issue with the architecture illustrated in Fig. 1. Here, the CDCs can only crop and scale the shadow copies while the user can create shares and reconstruct the original.

The secret video is inputted into the decoder, which gives us access to all the frames in their presentation order. Decoding is done by the FFmpeg decoder. With each frame, we input the frame, byte-by-byte to the SSS algorithm to create the shares. Once all the frames have been processed, and all shares created, the shadow copies are uploaded to the CDCs and the original secret video is deleted.

With the shadow copies stored in CDCs they can now be processed. This is where shadow copies will be cropped and scaled. Shares are cropped and scaled individually. To reconstruct the secret video, edited or otherwise, \( k \) shares are...
downloaded and reconstructed using $L(0)$ where $x_i$ is the number of that particular share and $y_i$ is the value of the share being used in reconstruction. Reconstruction leads to either the original secret being recovered, or the result of cropping and scaling of the shadow copies, but in plaintext.

C. System Implementation

We utilized FFmpeg’s libav library for decoding and encoding. This also gave us access to the frame data, which we could then encrypt. We originally intended to use libav to re-encode each shadow copy into the same format as the secret video, but this proved difficult to achieve with codecs that use lossy compression techniques. As a result, by re-encoding a share in the same format as the original video, we lose information, thus making reconstruction impossible. We could not reliably re-encode shares using lossy codecs so we opted to develop our own file format to store each share.

Our file format reflects our naive encryption of the video where all I, P and B frames are stored completely, and in presentation order. This is a less than ideal solution and leaves room for future improvements. As a result, we store a lot of redundant information for each frame, and as such, the space required for each share is dramatically higher than that of the original video because we do not employ any compression techniques.

Furthermore, we chose to store the header information in plaintext, while all the frame data is stored as ciphertext. The headers contain important information that it is needed to properly reconstruct the video. Even with headers in plaintext, the content of the video is still secure. An attacker does not gain any insight into the content of the secret video by knowing the headers. Our headers store timestamps, the resolution of the video, and the size of each frame in bytes.

Our system consists of three components: the video source, the $n$ shareholders, and the user, as illustrated by Fig. 1. A user would first create shares of the secret video before distributing all $n$ ($n = 5$ in our case) shadow copies to its shareholders. Cropping and scaling is then done on all $n$ shares. Any $k$ ($k = 3$ in our case) shareholders can bring their shadow copies together to reconstruct the video.

After decoding the input video, we process all P, B and I frames in presentation order. All frames contain $Y$ (luminance), $Cb$ and $Cr$ (chrominance) components with 4:2:0 sub-sampling. Each component is processed individually and independently of each other. The FFmpeg decoder interleaves the components to display the frame. To create all $n$ shares, the system computes $F_b(x)$ for each byte $b$ in the frame and for $n$ different values of $x$.

When all three components are processed, we will have the fully generated shadow copies for the frame. Each shadow frame is now written to its respective shares, where all frames are stored in presentation order. This simplifies the reconstruction phase.

Once the shadow videos have been created, they are uploaded to different CDCs, and the original video is deleted. The CDCs do not possess the means to reconstruct the video, or analyze the contents of the videos.

To achieve zooming in the video, the user submits the dimensions for the Region of Interest (RoI) along with the starting coordinate of the region. The system removes all data outside the RoI for all frames in the shares. When the frame is fully cropped, the system scales the cropped frame to the dimensions of the original frame. This is done by calculating $P(x, y)$ for each sub-pixel in the output frame. Note that all computations are performed under modulo $p$. Now that the RoI has been scaled, the frame is written to a new file that stores the zoomed shadow copy.

To reconstruct the original video, $k$ shareholders input their shares in the system. To find the original secret video, the $i^{th}$ byte of $k$ shadow copies are used in Lagrange interpolation to create the polynomial $L(x)$ where $x = 0$ and byte $b_i = y_i$ and $x_i$ is the number indicating the share’s id. $L(x)$ is now the reconstructed $i^{th}$ subpixel. After reconstructing a frame, it is re-encoded using FFmpeg’s encoding functionality. With all frames reconstructed, the original video has been retrieved.

IV. Analysis

In our testing, our secret video was the Foreman video (size: 436 KB, dimensions: 352 × 288). As shown in Fig. 2, our solution maintains the visual security of the video. An attacker cannot infer the content of the video by viewing the shadow copy. Furthermore, because our system uses the SSS scheme, we provide perfect secrecy. Assuming that an adversary does not have access to $k$ shadow copies, the adversary cannot infer the value of a specific subpixel in a frame. In our system, if an attacker gains access to 2 shares, for any particular value there are $p$ possible values for the secret byte in the finite field $Z_p$.

An additional benefit of using SSS is that it is resistant to data tampering. SSS provides data integrity by making the reconstruction of a secret impossible if the shadow value does not coincide with the original polynomial. Thus, if the adversary modifies a share, the polynomial being reconstructed will be incorrect, resulting in an unintelligible video.

The proposed system, though it provides perfect security, suffers from some limitations in its design. By storing video shares in their entirety the storage requirements of our system are rather strict. The shadow copies of the Foreman video take up 47 MB. The storage requirements of the shadow copies depend on the resolution of the video as well as its length. Thus, long, high resolution videos will require large amounts of free space. In our tests, a 01 : 58 minute video with $720 \times 480$ resolution require only 90 MB of storage in its compressed form, but approximately 2 GB for each of its shadow copies. The exact compressed-to-share storage ratio is difficult to pinpoint as it is a function of the video resolution and length.

Further limitations arise from share generation, cropping, scaling and reconstruction. Because each byte in the video data is considered a secret, any operation must be done on each individual byte. This means that there is a lot of processing...
overhead, especially with share creation, bicubic interpolation and Lagrange interpolation. If only cropping is requested by the user, the time to reconstruct the secret using Lagrange interpolation can be significantly reduced due to the smaller video resolution, as there are fewer secrets in each frame to reconstruct. Table I shows the run time of our system using the Foreman video. Share creation is relatively quick and is largely bound by decoding the input video. Reconstruction can take a lot longer because of Lagrange interpolation, where we have to calculate

\[ \frac{x - x_i}{x_j - x_i} \equiv (x - x_i)(x_j - x_i)^{-1} = 1 \mod p \]

which can be computationally expensive.

V. CONCLUSION AND FUTURE WORK

As previously stated, our implementation of our method on the encrypted videos uses a naïve representation of video data. This leads to increased storage requirements, and as such, a form of compression should be implemented. Video features such as Discrete Cosine Transform coefficients, motion vectors, inter frames and macroblocks would result in significant storage improvements. The benefit would be two-fold: i) reducing storage overhead, and ii) reducing processing overhead.

Implementing macroblocks for encrypted videos results in additional benefits by making the video better suited to be streamed to a user. Thus, the shadow copies can be streamed to a user and cropping and scaling can be done interactively. Also, in the future, we would attempt to identify a more efficient way of storing the encrypted data while still maintaining its security.

REFERENCES


Cybersecurity Benefits of Executive Order 13,694

William Augustine
Department of Informatics
The University at Albany
Albany, New York 12222 USA
Email: WAugustine@albany.edu

Abstract—On April 1, 2015, President Obama issued Executive Order 13,694 entitled “Blocking The Property Of Certain Persons Engaging In Significant Malicious Cyber-Enabled Activities” Is America safer because of this? This paper looks at the environment in which this Executive Order is intended to operate (including what may be meant by certain persons and significant malicious activities) and how it may aid the fight against cybercrime and cyberterrorism.

I. INTRODUCTION

Among the duties of the United States President, as the head of the executive branch of the government, is the responsibility to “take Care that the Laws be faithfully executed” [1]. One means of discharging this duty is through the use of Executive Orders. Executive Orders are directives from the President to the agencies which report to the executive branch [2], such as the Department of the Treasury. The issuance of Executive Orders is not intended to create law, but rather to influence an interpretation and enforcement of the nation’s existing laws. They have been issued by every President since George Washington1. Some have had more historical significance than others; it was through Executive Order that President Roosevelt ordered the internment of U.S. citizens of primarily Japanese descent [3] and in 1861 it was by Executive Order that President Lincoln suspended the writ of habeas corpus2.

In a February 2015 press release, the White House said that “[c]yber threats are among the gravest national security dangers to the United States” [4]. Less than two months later, on April 1, 2015, President Barack Obama issued an Executive Order [5] to add an additional means of protecting American interests against “certain persons engaging in significant malicious cyber-enabled activities”. This was done because, as chief executive, he “bears no greater responsibility than ensuring the safety and security of the American people” [6]. To understand what this Executive Order means, we will need to understand what “significant malicious cyber-enabled activities” are and who the “certain persons” might be. An understanding of the threat posed to society and the expectations for improved security will follow.

II. THE INTERNET THREAT ENVIRONMENT

1) Certain Persons: A cyberattack might be launched by an individual or group to further their own ends, or by an agent of a foreign power as an element of international conflict. The distributed and ubiquitous nature of the Internet means that the cyber-enabled activities of either type of actor could be enacted from and targeted to just about anywhere in the world. There are as many as 15 ways to identify the types of individuals or non-state-sponsored groups who commit these cyberattacks. These include organized crime and terrorist groups as well as others which fall under monikers such as hacker, hacktivist, script kiddie and scammer [7]. These acts could be driven by any combination of political interests, financial gain and emotion, among other motives.

State sponsored actors are becoming an integral part of the cyberwarfare and cyberdefense capabilities being built by many nations. The United States Cyber Command reached full operational capability on October 31, 2010 with a directive to “conduct full spectrum military cyberspace operations in order to enable actions in all domains, ensure US/Allied freedom of action in cyberspace and deny the same to our adversaries” [8]. The U.S. is not alone in the militarization of cyberspace. “China, India, and Russia alongside the U.S., the U.K. and South Korea are among the first group of countries to establish formal command and control (C2) over military assets in the cyber-domain” [9, p523].

It can be difficult to identify those persons who are engaged in cyberattacks or to determine their motives and affiliations. Appropriate requital requires accurate attribution. Even if the personal identity of a cyberattacker was unequivocally determined, what level of retribution is acceptable? Might it differ depending on the level of damage caused? What if they are located outside of one’s territorial borders? Is it different if they were acting under the orders of a foreign government, and could that even be determined? There is reason to believe that state sponsored cyberattacks have already been launched.

Some accuse national governments (particularly Russia, China, and the tandem of U.S. and Israel) of having already employed cyberattacks against other countries. Former Soviet nations experienced computer disruptions just prior to Russian military operations. “Russia has been implicated in several recent, high-profile cyberattacks, including those against Kyrgyzstan, Lithuania, Chechnya, Estonia, and Georgia. In most cases, no definitive evidence tied the Russian government to these attacks” [10, p18]. Those behind the massive online espionage incidents labeled Titan Rain, in 2003, and GhostNet, in 2009, have been traced back to China. “Although researchers investigating the GhostNet attacks have been unable to tie

---

1Except William Henry Harrison, who served less than 31 days in office.
2It was not until the following year, 1862, that Executive Orders were issued by number (http://www.presidency.ucsb.edu/ws/?pid=69792).
them directly to China or any other state, evidence suggests that these attacks were state sponsored at some level” [10, p17]. Reports have accused the U.S. and Israel of working in concert to develop and deploy the Stuxnet worm with the intent of disrupting Iran’s nuclear program [11, p1082]. These are just some of the ways that the internet has enabled cyber activities and exemplifies how difficult it is to appropriately assign blame.

These examples have focused on sources of foreign origin for a reason. For the purposes of Executive Order 13,694, “certain persons” applies to those “located in whole or in substantial part, outside the United States.”

2) Cyber-Enabled Activities: The Internet has radically changed the way that organizations interact with their clients in both the public and private sectors. It has done so by creating a means of communication that is world-wide and immediate. The resulting flow of information and money (which are not necessarily distinct) no longer knows limitations of national borders and thus falls into no single, well described legal jurisdiction. The interactions within these markets and forums occur in cyberspace. Wherever money and power are parleyed, there will be crime, and cyber-enabled activities will give rise to cybercrime. This “where” does not exist physically and therefore no universal agreement exists regarding how to go about policing such a transnational and ethereal dominion. No nation has the mandate providing legitimate authority.

Making this more difficult still is the fact that even among key stakeholders in Internet security (such as legal experts, academics and cybersecurity practitioners) there does not exist a single accepted definition of what constitutes cybercrime [12]. Whose statutes are in effect: those governing where the perpetrator physically was at the time of the crime, wherever the communication flow passed, or where the victim was? If no one can claim control of the ephemeral realm of cyberspace, should it have its own laws and, if so, who would enforce them?

Similar to the vagaries surrounding cybercrime, there is no definitive meaning for the term cyberterrorism. “[N]o single definition of cyberterrorism is agreed upon by all, in the same way that no single, globally accepted definition of classical political terrorism exists” [13, p3]. One does not need to look at this on a global scale to see the problem. The FBI and the U.S. Department of State have different definitions of terrorism. “Even the U.S. government cannot agree on one single definition” [14, p1]. The difficulties in differentiating criminal activity from those that are terror-related in the physical world mirror the issues defining cybercrime and cyberterrorism.

Whatever the official definition, a 2014 Pew Research study showed that a majority of the experts surveyed believed that in the next decade there will be a major cyberattack resulting in “widespread harm to a nation’s security and capacity to defend itself and its people” [15, p6]. This attitude is also reflected by the law enforcement community. Addressing the Senate Judiciary Committees Subcommittee on Crime and Terrorism, FBI Cyber Division Assistant Director Jordan Snow stated that “[a]s the subcommittee is aware, the number and sophistication of cyber attacks has increased dramatically over the past five years and is expected to continue to grow” [16]. “The Federal Bureau of Investigation (FBI) considers high-tech crimes to be the most significant crimes confronting the United States” [17].

When “certain persons”, regardless of whether they are under the direction of a foreign government, engage in “significant malicious cyber-enabled activities”, those activities can be classified within four types of action: use, misuse, offensive use and cyberterrorism [13, p276]. Each of these differs in the way they are employed and the types of damage that might be dealt to susceptible targets.

3) Susceptible Targets: Computer technology can be used as a communication tool to recruit, train, mobilize and coordinate the activities of the participants of a cyber-enabled activity. This constitutes the first action within the Conway typology –use– and targets people to influence them to become involved in the activity in ways ranging from fundraising through direct action. Terrorist groups, such as Hamas and Hezbollah, use websites, online chat groups and email to radicalize and organize new adherents. They post propaganda to incite violence and provide instructions on how to build bombs and launder money [13, p277].

Misuse and offensive use involve computers attacking computers. These actions are often the result of compromises caused by attack software (a.k.a. malware). Malware comes in many forms (e.g. virus, buffer overflow) and delivery methods (e.g. infected email, vulnerable server software). Sometimes the malware leaves control software on the infected computer so it can later be remotely controlled and joined to a botnet4. Misuse of computers involves the disruption of legitimate cyber-enabled activities. This can be through the defacement of web sites or by using denial of service (DoS) attacks. Inadvertent DoS conditions can be the effect of propagating malware. The Blaster worm may have contributed to the severity of the 2003 Northeast blackout that left over 50 million people in Canada and the U.S. without power [18].

Given how reliant corporations have become on computers (some businesses do not even have a physical presence) attacks against networks could have devastating economic effects; consider the effect of a successful attack on a stock exchange. Offensive use of the Internet includes using technology for theft, such as identity theft, account siphoning, or copyright infringement.

The fourth action in the Conway typology is cyberterrorism, which means using Internet technologies to cause physical harm. This is the rarest form of cyberattack, but the one that could wreak the most havoc. There have only been two confirmed fully-cyber attacks that resulted in physical destruction [19]. The first was the Stuxnet worm that caused damage to centrifuges in the Iranian nuclear facility at Natanz. Only recently, a second event at a German steel plant caused

4 A collection of infected computers (called zombies) that a hacker can remotely take control of to act en masse.
massive damage to a blast furnace. Neither incident resulted in personal injury.

Given how much critical infrastructure is controlled by computer it may only be a matter of time before more of these occur and eventually result in bodily harm or loss of life. SCADA\(^4\) systems manage a wide array of functions such as air traffic control, ground traffic signals, and municipal water processing. If a cyberterrorist group could compromise an active nuclear power facility or weapons control system, a truly terrifying event could result. “The more technologically developed a country is, the more vulnerable it becomes to cyberattacks against its infrastructures” [20, p1].

III. THE STATE OF U.S. CYBER-SECURITY POLICY

So far, the majority of targets of “significant malicious cyber-enabled activities” have been private sector entities and individuals. These are dealt with using the same legal framework as employed in the physical world. “The United States does not have a national strategy exclusively focused on combating cybercrime. Rather, there are other, broader strategies that have cybercrime components” [17, p16].

Incidents with an international component necessitate federal government involvement. In the case of the 2014 Sony hack (widely considered a reaction to the movie “The Interview”) the FBI determined that the attack was state-sponsored. In response, the President issued an Executive Order imposing additional economic sanctions against the North Korean government [21]. This scenario demonstrates again the difficulties inherent in attribution for acts in cyberspace. There are computer security experts who dispute that North Korea was behind the computer compromise and release of the Sony emails and instead believe it may have been perpetrated by disgruntled former employees [22].

In a theater of wartime operations the stakes are far greater than in the movie theater. Insurgent forces in Iraq have been found in possession of video feeds from U.S. military predator drones. They were able to hack into the video surveillance communication streams sent from the drones to their operational base [23]. Virus code has been detected on computers that serve as command and control consoles for military drones [24]. Even after multiple attempts to remove the malware it continued to persist. To this point, Air Force network security specialists have considered it benign, but just consider the scenario where hostile forces were able to take remote control of our drones (as is done with botnets) and turn them against U.S. or allied troops.

IV. WHAT THIS EXECUTIVE ORDER PROVIDES

Executive Order 13,694 does not address securing cyberspace against the most potentially damaging effects resulting from “significant malicious cyber-enabled activities”. The lack of focus on prior restraint does not mean that it is without reason. The particulars of this Executive Order are consistent with the way in which cyber-enabled activities have been dealt. “[M]any of these so-called cybercrimes can be easily likened to traditional crimes...criminals are often prosecuted using laws intended to combat crimes in the real world” [17, p 4]. Asset forfeiture is one such tactic that is employed to combat crimes in the real world. There are strong similarities in the structure of organized crime syndicates and terrorist groups. Federal forfeiture statutes exist in order to suppress the ability of criminal organizations from using their ill-gotten gains to improve their defense against prosecution (both criminally and civilly) and provide recompense to injured parties. These are Racketeer Influenced and Corrupt Organizations statutes, a.k.a. RICO [25]. This provides for the ability of the government to seize any assets that are involved, in part or in whole, in the commission of criminal activity. In 2013, RICO was first used in the fight against online crime [26].

Forfeiture statutes when applied against terror organizations were strengthened by the Patriot Act [27]. All assets belonging to someone involved in terrorism (either domestically or internationally) are acceptable targets for seizure and forfeiture regardless of whether those assets were involved in the planning or commission of an act of terror. The Patriot Act also provides an impendence to terror funding by applying RICO provisions to those involved in reverse money laundering\(^5\).

If Patriot Act strengthened RICO statutes already allowed for “blocking the property of certain persons engaging in significant malicious cyber-enabled activities” what was gained in Executive Order 13,694? Under previous interpretation of these laws judicial action was required before a law enforcement agency would be allowed to seize assets. Now, assets can be seized from these “certain persons” as determined “by the Secretary of the Treasury, in consultation with the Attorney General and the Secretary of State”. This shows that, according to the President, searches and seizures when in suppression of terrorist activity are not considered unreasonable in terms of the Fourth Amendment to the Constitution [28].

V. CYBER POLICIES BEYOND THE U.S.

Although the U.S. created the Internet, Americans are not alone in cyberspace and the difficulty in securing one’s interests is shared by all nations. Each nation will have its own laws influencing the interpretation of cybercrime and cyberterror, as will multi-national bodies. Cybersecurity will need to be an international effort because of the ubiquitous nature of the Internet.

In 1999, the United Nations issued its Manual on the Prevention and Control of Computer Related Crime [29]. An attempt to define cybercrime was included and it called for further cooperation among nations, including a harmonization of laws and the creation of a jurisdictional entity for cyberspace. There was never a binding agreement reached as a result of these efforts.

\(^4\)Supervisory Control And Data Acquisition: systems that monitor and control hardware such as those in a manufacturing plant or telecommunications facility.

\(^5\)Money laundering is the process of disguising the true source of proceeds gained from illegal activities. Reverse money laundering is disguising the source of legitimate money intended to fund illegitimate (e.g terror-related) activities.
The oldest multinational agreement addressing this topic is the Council of Europe’s Convention on Cybercrime, a.k.a. the Budapest Convention. Participation in this effort was extended to nations beyond the 45 members of the Council of Europe. This agreement was meant to create an accepted definition of cybercrime and provide for greater cooperation in the investigation and prosecution of cybercrimes involving multiple nations [30]. Each of the signatories is expected to draft into its own laws a standard legal framework criminalizing activities such as child pornography, hacking, and violations of intellectual property rights. It went into force on July 1, 2004 and has been signed by 44 nations including the United States, which ratified it in 2006. Russia and China are conspicuously absent from the list of ratifying nations.

Just as there exist unions of nations for economic purposes (such as NAFTA) and for physical security and mutual defense (such as NATO), cooperative “cyber-blocs” of nations sharing a common agenda and cybersecurity strategy have formed. Five in particular are:

- “The Anglosphere, led by the US and the UK, emphasizes a leading private sector role, an educated workforce, and outreach and diplomacy.
- The EU, led by Germany, focuses on a robust legal and regulatory framework, and on the promotion of the Council of Europe (Budapest) Convention of Cybercrime as a blueprint for international cooperation and enforcement.
- The Baltic States are in tight cooperation with NATO in the development of their national cyber-security strategies.
- The post-Soviet CIS bloc, led by Russia with some degree of Chinese cooperation, focuses on internal threats, abhors extra-territorial judicial action, and promotes a corresponding international framework under the auspices of the UN.”

And we can possibly add in another bloc:

- “The Nordic States, led by Finland, aiming for practical cooperation in building a common cyber security strategy (with common laws) and supporting each other to develop the necessary capacity to build a defensive coalition.” [31, p147-148].

Global Jus in Bello agreements exist defining the acceptable use of force and limitations on types of weapons allowed in the physical world. “The dangers are so great that cyber arms qualify as weapons of mass destruction” [32, p283]. What are needed are similar international agreements on what are and what are not acceptable cyber-enabled activities and under what conditions their uses are approved. “Without a global cyber-warfare non-aggression pact, it will be impossible to distinguish whether a cyberattack is state sponsored or independent terrorism” [15, p31].

VI. EXPECTED EFFECTIVENESS

In November 2014, months before the passage of Executive Order 13,694, the U.S. employed forfeiture statutes in the fight against online cybercrime. This was prosecuted by the U.S. Department of Justice in concert with 16 international entities toward the closure of Silk Road 2.0 and other dark web market sites [33]. Without using Executive Order 13,694 and even without international cooperation, foreign companies that do business in the United States can be susceptible to forfeiture through legal rather than diplomatic channels. The French bank, BNP Paribas S.A., was “ordered to forfeit $8,833,600,000 to the United States and to pay a $140,000,000 fine” for deliberately evading U.S. sanctions against Sudan, Iran and Cuba[34].

A potential way to evaluate the efficacy of a deterrence effect of this Executive Order is to perform a hypothesis test using attack statistics. We will assume that there is no difference in the number of attacks before the order and after.

**Hypothesis 1** There is a difference in the number of attacks before and after the Executive Order.

Using cyber attack statistics for January 2014 through March 2016, inclusive, as reported by Hackmageddon.com, results (shown in Table 1) from a paired two-tailed t-test fail to reject $H_0$. A Kolmogorov-Smirnov comparison found the two samples were both normally distributed. Additionally, calculating for PreEO<PostEO reveals a $p-value = 0.9915$ meaning that it is statistically significant that there were typically fewer attacks in the months prior the Executive Order than those following to it. Further investigation reveals that the percentage related to cybercrime was also higher (with fewer attributed to hacktivism). This is highly circumstantial and should not be interpreted as implying causation; there is no saying that without Executive Order 13,694 the numbers would have been even higher and due to the time periods used an autocorrelation effect could be present.

**TABLE I**

<table>
<thead>
<tr>
<th>Period</th>
<th>N</th>
<th>Mean</th>
<th>SDev</th>
<th>SE Mean</th>
</tr>
</thead>
<tbody>
<tr>
<td>PreEO</td>
<td>15</td>
<td>76.067</td>
<td>9.7649</td>
<td>2.521</td>
</tr>
<tr>
<td>PostEO</td>
<td>13</td>
<td>84.538</td>
<td>7.4455</td>
<td>2.065</td>
</tr>
</tbody>
</table>

Laws exist for preventive, curative and facilitative functions in order to dissuade behaviors, to punish those that decide to transgress the law, and to support social institutions, respectively[35, p234]. The previous paragraph addressed the case of prevention. In regard to the second, ex post facto form, as of the writing of this paper, Executive Order 13,694 has not been cited in any court cases. In this analysis, there is no reason for consideration of the facilitative form.

If one is inclined to believe in the power of economic sanctions then the provisions of this Executive Order should be expected to assist in effectively suppressing “malicious cyber-enabled activities”, although from the cases against Silk Road 2.0 and BNP Paribas S.A. they are not necessary for it. In fact, according to the World Trade Organization (WTO) this order...

---

6 http://www.hackmageddon.com/category/security/cyber-attacks-statistics/
is unnecessary if the President decides to assess sanctions in the event of “significant malicious cyber-enabled activities”.

“WTO members have successfully argued that the WTO Dispute Settlement Body (DSB) has no authority when sanctions are imposed for national security reasons”[36, p1485]. “It is agreed, however, that online activities substantially improve the ability of such terrorist groups to raise funds, lure new faithful, and reach a mass audience” [13, p277]. It might be expected that asset seizure will help materially reduce the ability of “certain persons” to function in these ways. An investigation by the GAO into the effectiveness of the first decade of RICO forfeiture in the war on drugs found that “neither the dollar value nor the type of assets forfeited to the Government by criminal organizations has been impressive compared to the billions generated annually through drug trafficking.”[37, pi-ii].

More questions than answers have been presented here, but one more question in particular needs to be posed: in the past year, has Executive Order 13,694 made the U.S. safer? There appears to be reason to believe that there are no cyber-security benefits to Executive Order 13,694 although the President of the United States declared a state of emergency to enact it.

ACKNOWLEDGMENT

Gratitude is due to Dr. Deborah Andersen of the University at Albany for guidance, instruction and editorial prowess. All writing errors in this version are attributable to the author alone. John Moore, Esq. provided expert legal guidance and any misinterpretations are (again) solely the fault of the author.

REFERENCES

[21] Executive Order 13,687 ofJanuary, 2, 2015, Imposing Additional Sanc-

ions With Respect To North Korea, Code of Federal Regulations, Title 6 (2015), 80 FR 819
AUTHOR BIOGRAPHIES

Pradeep K. Atrey is an Associate Professor at the State University of New York, Albany, NY, USA. He is also an Adjunct Professor at University of Ottawa, Canada. His current research interests are in the area of Security and Privacy with a focus on multimedia surveillance and privacy, multimedia security, secure-domain cloud-based large-scale multimedia analytics, and social media. He has authored/co-authored over 95 research articles at reputed ACM, IEEE, and Springer journals and conferences. Dr. Atrey is on the editorial board of several journals, and he has been associated with over 30 international conferences/workshops in various roles. Dr. Atrey was a recipient of the ACM TOMM Associate Editor of the Year (2015), the IEEE Comm. Soc. MMTC Best R-Letter Editor Award (2015), the Erica and Arnold Rogers Award for Excellence in Research and Scholarship (2014), ETRI Journal Best Editor Award (2012), ETRI Journal Best Reviewer Award (2009) and the three University of Winnipeg Merit Awards for Exceptional Performance (2010, 2012 and 2013). He was also recognized as the ACM Multimedia Rising Star (2015), the ICME Outstanding Organizing Committee Member (as Publicity Chair) (2013) and the ICME Quality Reviewer (2011).

William Augustine works for the University at Albany Department of Computer Science as the UNIX Systems Administrator and is a part-time student, in the second year of the Information Science PhD program. His primary area of specialization is Information Assurance with a secondary specialization in Information in Organizational Environments. His research interests include computer and network security, intrusion detection and prevention, and risk analysis. He earned his BS in Economics, MS in Computer Science, MBA, and Certificate of Graduate Study in information Security all from the University at Albany.

Nitesh K Bharadwaj is a PhD scholar at the Department of Computer Science and Engineering of Defence Institute of Advanced Technology (DIAT), Pune-India. He received his M.Tech. in VLSI Design from Indian Institute of Information Technology and Management, Gwalior, India. He is currently working with Digital Forensics Lab at DIAT Pune and his research interests include Digital Forensics, Hardware Security, Internet of Things and Data Mining. He is a student member of IEEE and ACM.

Pankaj Choudhary is a PhD student at the Department of Computer Engineering, Defence Institute of Advanced Technology (DIAT DU), Pune, India. He received his MTech in Information Security from Indian Institute of Information Technology and Management, Gwalior, India. He is currently working in Digital Forensics Lab at DIAT Pune. His research interests include Social Network Analysis, Digital Forensics, Social Media Forensics and Information Security. He is a student member of IEEE and ACM.

Matthew Davis earned his BS in Electrical Engineering from Binghamton University in 2015. He currently attends Binghamton University and is pursuing his PhD in Electrical Engineering. He is a member of Tau Beta Pi and Eta Kappa Nu. His research interests are in SCADA and industrial control systems security.

Nic DePaula is a PhD student in Information Science at the University at Albany, State University of New York. His research addresses issues in the area of information security, e-government, communication and the media. Mr. DePaula also holds a degree in Geography and Urban Planning from the University at Albany and has worked in multiple IT and research positions for the public and private sector.

Andrey Dolgikh obtained his engineering degree in Computer Engineering from South Urals State University in 2005 in Russia. He received his PhD degree in Electrical Engineering in
Binghamton University in 2014, while working on a number of behavior based intrusion detection research projects funded by the AFOSR. Currently Dr. Dolgikh is a Research Assistant Professor at the SUNY Binghamton. The current focus of his research activity is Safety and Security of Industrial Control Systems. Under ONR funded project “Security approach for Industrial Control Systems” he develops new methods to improve security and safety of the ICS. Dr. Dolgikh designed and supervised the implementation of Binghamton University ICS testbed. He has developed ICS security course. This course covers security topics specific to cyber-physical systems. In this class students will work with vulnerable industrial control systems, time critical systems such as UAVs (plane or copter), and life critical systems such as medical devices. The course actively uses virtualization to increase student involvement in research experimentation. This course is adopted by Binghamton University and offered to electrical and computer engineering students specializing in computer security as well as off-campus. Dr. Dolgikh is an author of a respectable number of publications reflecting his security research.

Sanjay Goel is a Professor and Chair of the Information Technology Management Department in the School of Business, Director of the Center for Forensics Analytics Complexity Energy Transportation and Security, and the Director of Research at the NYS Center for Information Forensics and Assurance at UAlbany. He is also the Director of the Digital Forensics Program at the University. Dr. Goel received his Ph.D. in Mechanical Engineering from RPI. His research interests include information security, security of cyber physical systems, music piracy, cyber warfare and self-organization in complex systems. His research on self-organizing systems includes traffic light coordination, smart grid and social networks. He is lead author of Smart Grid Vision prepared by IEEE Communications Society and the IEEE Standards Association. He won the promising Inventor’s Award in 2005 from the SUNY Research Foundation. In 2006, he was awarded the SUNY Chancellor’s Award for Excellence in Teaching, the UAlbany Excellence in Teaching Award, and the Graduate Student Organization Award for Faculty Mentoring. In 2010 he was awarded the UAlbany Excellence in Research Award. In 2015, he was also awarded the SUNY Chancellors’ Excellence in Academic Service, UAlbany Presidents’ Excellence in University Service, and School of Business Excellence in Research Award. He is the only faculty in the history of the University who has received excellence awards in all three categories i.e. research, teaching, and service. He was named one of the three AT&T Industrial Ecology Faculty Fellows for 2009-2010. He has received grant funding from multiple sources including: National Institute of Justice, U.S. Department of Education, National Science Foundation, Intelligence Advanced Research Project Activity, Region II University Transportation Research Center, New York State Energy Research and Development Agency (NYSERDA), AT&T Foundation and James S. McDonnell Foundation. He recently received a $800,000 grant to establish a Blackstone Launchpad to foster entrepreneurship and innovation among students at UAlbany and a 1,600,000 award to develop strategies for detection of inside threat actors before they commit theft.

Gayathri Gopalakrishnan is a consultant at Deloitte. She graduated with a MS in MIS degree from SUNY Buffalo in 2014.

Manish Gupta is Vice President of Cybersecurity at one of the top 15 banks in US and an adjunct assistant professor at State University of New York at Buffalo. He has authored or co-authored more than 70 research articles that published in journals, books and conference proceedings. His research has won best paper awards and has been published in MISQ, DSS, ACM Transactions, JOEUC amongst others. He has co-edited 7 books in the area of information assurance and cybersecurity. He holds several professional designations including CISSP, CISM, CISA, CRISC, CFE and PMP. He teaches graduate level courses in IT Auditing and IT Risk Management at SUNY Buffalo. He received PhD (Management/MIS) and MBA (Finance and
MIS) from SUNY Buffalo in 2011 and 2003, respectively and a bachelors degree in Mechanical Engineering from IET Lucknow, India in 1998.

Delbert Hart is an Associate Professor in the Computer Science Department at SUNY Plattsburgh. He earned his D.Sc. in Computer Science at Washington University in St. Louis. His research interests are Computer Security, Computer Science Pedagogy, and Distributed Systems.

Susan Heilman recently graduated from Rochester Institute of Technology (RIT) in May 2016 with a Master of Science in Computing Security. During her time at RIT, Susan participated in many attack/defend and capture the flag competitions hosted by the college and elsewhere. She also served a term as committee head for Women in Computing at RIT where she was in charge of coordinating the organization’s annual all female coding hackathon – WiCHacks. Susan will soon be working for the MITRE Corporation as a Cyber Security Engineer.

Emrah Korkmaz received his M.S. degree in Electrical Engineering from the Yildiz Technical University, Istanbul, Turkey. He is currently pursuing PhD degree in the Electrical and Computer Engineering, from the Binghamton University, NY, USA. He is also member of Electronics Engineering at the Turkish Military Academy awarded with a full scholarship in order to obtain PhD degree in the Binghamton University. His research interests include industrial control systems, cyber-physical systems security and their vulnerability assessment against the cyber-attacks, and security of power systems.

Amin G. Kosseim is a 28 year veteran of the New York City Police Department. Deputy Inspector Kosseim holds the distinction of being the highest ranking Arab American in the Department. He earned his Master’s Degree in Public Administration from John Jay College in New York City in 1998 and returned in 2006 as an Adjunct Professor. He is certified in several highly specialized law enforcement training public courses which include: Dignitary Protection, Methods of Security and Target Hardening, and High Intensity Drug Trafficking. Deputy Inspector Kosseim has recently graduated from the prestigious Naval Postgraduate School in Monterey California, earning his second Master’s Degree in Homeland Security. Appointed to Community Affairs Bureau in 2007, Deputy Inspector Kosseim was responsible for the overall supervision and administration of Community Affairs Bureau Special Projects Unit. As the Commanding Officer of Special Projects Unit for Community Affairs Bureau, where he focuses on improving the relationship between the community and the members of the NYPD. Drawing on his years of law enforcement experience and extensive knowledge of community policing; he assumed a lead role in the development and implementation of several programs specifically designed to improve police/community relations. Working collaboratively with youth groups, religious organizations and local civic organizations; designing participation programs that encourage community involvement and responsibility. He has since been appointed to serve as the Executive Officer of the Community Affairs Division which oversees the Crime prevention Section, Community Outreach, and the New Immigrant Outreach Unit. In November of 2014, Deputy Inspector Kosseim was assigned to be the Special Projects Coordinator for the Borough of Brooklyn North. In that capacity, some of his responsibilities are crime analysis, ceasefire, quality of life initiatives, evidence collection teams, 311 program, school safety issues, traffic management safety, narcotics crime control, gang offenses, and community affairs.

Odd-Arild Kristensen is a Software Developer working at Comcast Corporation. He graduated from State University of New York, Albany with a Master's degree in Computer Science.

Nilofar Mansourzadeh is a Ph.D. student in the School of Computer Science at Carleton University in Ottawa, Canada. She received her Master’s degree in Software Engineering in the UAE and Bachelor’s degree in IT Engineering in Iran. She works as a research assistant in the
Carleton Computer Security Lab (CCSL) focusing on software diversity. Her research interests include information security, machine learning and cloud computing. She has previously published articles in IEEE conference proceedings.

**Manoranjan Mohanty** is a postdoctoral researcher in Center for Cyber Security, New York University Abu Dhabi. He received PhD in Computer Science from National University of Singapore. His research interests include encrypted domain processing, digital forensics, multimedia security, applied cryptography, and privacy-preserving biometrics.

**Ashley Moni** is a graduate student in the School of Computer Science at Carleton University in Ottawa, Canada. She received a B.S. degree in General Science from the University of Waterloo in 2013 and has published research before under the name “Tarun Moni.” Her research interests include computer security, type theory, artificial intelligence, and quantum computation.

**Borke Obada-Obieh** is a M.C.S student in the School of Computer Science at Carleton University in Ottawa, Canada. She works as a teaching assistant at the university and is also a research assistant in the Carleton Computer Security Lab (CCSL). Her current research interests include systems and network security.

**Bheesham Persaud** is a B.C.S student in the School of Computer Science at Carleton University in Ottawa, Canada. He has worked as a research assistant for Prof. Anil Somayaji in the Carleton Computer Security Lab (CCSL). His current research interests include computer security and authentication.

**Joshua Regan** conducts research and teaches national security studies at the University of New Haven. He also is a faculty member affiliated with the Connecticut State University System. His research primarily focuses on failed states, terrorist networks, and contemporary maritime piracy. Josh's dissertation presents a model that predicts the intensity of global pirating attacks. Prior to his doctoral studies, Josh earned a Masters of Science degree in National Security from the University of New Haven and a Masters of Arts in Geography from the University of Connecticut.

**Raj Sharman** is an Associate Professor in the Management Science and Systems Department of the State University of New York at Buffalo. His expertise is in the areas of Information Assurance and the development of Biologically Inspired Computer Security Models, Disaster Preparedness and Response Management, Patient Safety and Health Care Systems. He has published widely in National and International journals and is the recipient of several grants from university and external agencies, including the National Science Foundation. He received his PhD in Computer Science and a Master of Science degree in Industrial Engineering from the Louisiana State University. He received his Bachelors degree in Engineering and Masters Degree in Management from the Indian Institute of Technology, Bombay, India.

**Bhupendra Singh** is a PhD scholar at the department of computer science and engineering of Defence Institute of Advanced Technology (DIAT), Pune-India. He received M.Tech. in Computer Science and Engineering from Central University of Rajasthan. His work is currently focused on File System Analysis, File Carving and User Activity Analysis on Windows systems.

**Upasna Singh** completed her Doctorate Degree from Indian Institute of Information Technology, Allahabad in the field of Robotics and Artificial Intelligence. She is currently working in Department of Computer Science & Engineering, Defence Institute of Advanced Technology, Pune, India as an Assistant Professor. She has been working in the area of Data Mining, Digital Forensics, Big Data and Social Network Analysis. She has developed Digital Forensics Laboratory (DFL) at DIAT and heading Data Recovery Cell which resolves real time cases like Data Theft Detection, Data Recovery and Analysis of Hard Drives and other digital
devices. She is actively involved in various upcoming projects of DIAT Collaborative research with DRDO, CDAC Trivandrum, DIARA, IDS Headquarters, and Delhi.

**Victor Skormin** has completed two one-year terms as an Air Force Senior Research Associate appointed by the National Research Council, working on the information security related projects. Since 2001 he has been an organizer of the bi-annual International conference "Mathematical Methods, Models and Architectures for Computer Networks Security" in St. Petersburg, Russia (sponsored by the US Air Force and Navy), and the co-editor of eight volumes of conference proceedings published by Springer under the Lecture Notes in Computer Science series. Former PhD students supervised by Dr. Skormin are employed by CERT, Assured Information Security, George Mason University and SUNY Research Foundation participate in the most advanced research projects in Cyber Security. Dr. Skormin is an author of numerous publications, he is a recipient of an IEEE Best Paper Award and an international Best Book Award for his book on Immunocomputing.

**Anil Somayaji** is an Associate Professor in the School of Computer Science at Carleton University in Ottawa, Canada. He received a B.S. (1994) in Mathematics from the Massachusetts Institute of Technology and a Ph.D. (2002) in Computer Science from the University of New Mexico. He has served as the general chair and the program committee chair of the New Security Paradigms Workshop. He has also served on the program committees of major computer security venues including ACM CCS, USENIX Security, ACSAC, and RAID, among others. His research interests include computer security, operating systems, complex adaptive systems, and artificial life.

**Judith L. Tabron** received the A.B. degree (with honors) in English literature from the Bryn Mawr College, Bryn Mawr PA, USA, in 1990; and the M.A. degree in 1992 and Ph.D. in 1999, in English literatures, from Brandeis University, Waltham MA, USA. She has completed an internship on improving jury instructions in death penalty cases and expects to complete her M.A. in forensic linguistics from Hofstra University, Hempstead NY, in December 2016. She has been Director of Faculty and Student Computing Services at Hofstra University since 2010. Previously she served as Director of Faculty Computing Services beginning in 2004. She served as Director of Academic Computing at Brandeis University 1997-2002, where she introduced Windows and served as NT system administrator as well as WebCT system administrator (Unix). She has taught college classes ranging from introductory composition to global popular culture. She has been providing faculty development since 1993 and led the development of a financially and educationally successful distance learning program at Hofstra beginning in 2005. Dr. Tabron is a member of Infragard and completed advanced incident response training at Black Hat 2015. She has completed Cornell’s Institute for Internet Culture, Policy, and Law (2014) and attended the first and second annual conferences of the Journal of Law & Cyber Warfare (2014 and 2015). She is active in EDUCAUSE and was a fellow of the Frye Leadership Institute in 2008. She is chair of the New York State chapter of the ACE Women’s Network.

**Merrill Warkentin,** PhD is a Professor of MIS and the Drew Allen Endowed Fellow in the College of Business at Mississippi State University. His research, primarily on the impacts of organizational, contextual, situational, and dispositional influences on individual computer user behaviors in the context of information security and privacy, has appeared in MIS Quarterly, Decision Sciences, Journal of the Association for Information Systems, European Journal of Information Systems, Decision Support Systems, Computers & Security, Information Systems Journal, Communications of the ACM, Communications of the AIS, Journal of Information Systems, and others, and is the author or editor of seven books on technology. He has authored or co-authored over 250 published manuscripts, including over 60 peer-reviewed journal articles.
He serves or has recently served as Associate Editor of MIS Quarterly, Information Systems Research, Decision Sciences, European Journal of Information Systems, Information & Management, and other journals. He has Guest Edited several issues of top journals. He is Senior Editor of the AIS Transactions on Replication Research and an Eminent Area Editor (MIS) for Decision Sciences. Dr. Warkentin has served as AIS Departmental Editor for IS Security & Privacy, the Chair of the IFIP Working Group on Information Systems Security Research (WG8.11/11.13), the Vice President for Publications of the Decision Sciences Institute (DSI), and a national Distinguished Lecturer for the Association for Computing Machinery (ACM). His work has been funded by NATO, NSF, NSA, DoD, Homeland Security, IBM, UN, and others. He has chaired several national and international conferences, has been Track Chair at ICIS, AMCIS, ECIS, and DSI, and will be the Program Co-Chair of the AIS Americas Conference on Information Systems (AMCIS) in 2016. Dr. Warkentin has served as a visiting professor or invited speaker at numerous universities in North America, Europe, and Asia.

Jonathan Williams is a graduating from the Rochester Institute of Technology with a Bachelor's degree in Information Security and Forensics. In his spare time Jonathan enjoys scripting and Windows hardening. He also has focused interests' in the network monitoring and incident response computing security fields. His future goals are to become an incident response team supervisor for a prestigious security firm.
<table>
<thead>
<tr>
<th>Author Name</th>
<th>Pages</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pradeep K. Atrey</td>
<td>pp. 74-78</td>
</tr>
<tr>
<td>William Augustine</td>
<td>pp. 79-83</td>
</tr>
<tr>
<td>Charles Barry</td>
<td>pp. 32</td>
</tr>
<tr>
<td>Nitesh K Bharadwaj</td>
<td>pp. 59-67</td>
</tr>
<tr>
<td>Tim Brown</td>
<td>pp. 1</td>
</tr>
<tr>
<td>Pankaj Choudhary</td>
<td>pp. 59-67, 46-51</td>
</tr>
<tr>
<td>Matthew Davis</td>
<td>pp. 13-18</td>
</tr>
<tr>
<td>Nic DePaula</td>
<td>pp. 26-31</td>
</tr>
<tr>
<td>Andrey Dolgikh</td>
<td>pp. 13-18</td>
</tr>
<tr>
<td>Justin S. Giboney</td>
<td>pp. 2-5</td>
</tr>
<tr>
<td>Sanjay Goel</td>
<td>pp. 26-32</td>
</tr>
<tr>
<td>Gayathri Gopalakrishnan</td>
<td>pp. 6-12</td>
</tr>
<tr>
<td>Manish Gupta</td>
<td>pp. 6-12</td>
</tr>
<tr>
<td>Delbert Hart</td>
<td>pp. 33-39</td>
</tr>
<tr>
<td>Jason Healey</td>
<td>pp. 41</td>
</tr>
<tr>
<td>Susan Heilman</td>
<td>pp. 68-73</td>
</tr>
<tr>
<td>Daryl Johnson</td>
<td>pp. 68-73</td>
</tr>
<tr>
<td>Emrah Korkmaz</td>
<td>pp. 13-18</td>
</tr>
<tr>
<td>Amin G. Kosseim</td>
<td>pp. 42-45</td>
</tr>
<tr>
<td>Adem Kotil</td>
<td>pp. 2-5</td>
</tr>
<tr>
<td>Odd-Arild Kristensen</td>
<td>pp. 74-78</td>
</tr>
<tr>
<td>Nilofar Mansourzadeh</td>
<td>pp. 19-25</td>
</tr>
<tr>
<td>Michele Markoff</td>
<td>pp. 40</td>
</tr>
<tr>
<td>Manoranjan Mohanty</td>
<td>pp. 74-78</td>
</tr>
<tr>
<td>Ashley Moni</td>
<td>pp. 19-25</td>
</tr>
<tr>
<td>Borke Obada-Obieh</td>
<td>pp. 19-25</td>
</tr>
<tr>
<td>Bheesham Persaud</td>
<td>pp. 19-25</td>
</tr>
<tr>
<td>Joshua Regan</td>
<td>pp. 42-45</td>
</tr>
<tr>
<td>Raj Sharman</td>
<td>pp. 6-12</td>
</tr>
<tr>
<td>Bhupendra Singh</td>
<td>pp. 39-43</td>
</tr>
<tr>
<td>Upasna Singh</td>
<td>pp. 46-51, 59-67</td>
</tr>
<tr>
<td>Victor Skormin</td>
<td>pp. 13-18</td>
</tr>
<tr>
<td>Anil Somayaji</td>
<td>pp. 19-25</td>
</tr>
<tr>
<td>Judith L. Tabron</td>
<td>pp. 52-58</td>
</tr>
<tr>
<td>Merrill Warkentin</td>
<td>pp. 42-45</td>
</tr>
<tr>
<td>Jonathan Williams</td>
<td>pp. 68-73</td>
</tr>
</tbody>
</table>